INSTITUTO FEDERAL DE SANTA CATARINA
CURSO DE ENGENHARIA ELÉTRICA

JOÃO ANTÔNIO OLDENBURG VIEIRA

APRENDIZADO POR REFORÇO PROFUNDO APLICADO À DETECÇÃO DE ANOMALIAS

TRABALHO DE CONCLUSÃO DE CURSO

ITAJAÍ
2025

JOÃO ANTÔNIO OLDENBURG VIEIRA

APRENDIZADO POR REFORÇO PROFUNDO APLICADO À DETECÇÃO DE ANOMALIAS

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia Elétrica do Instituto Federal de Santa Catarina - IFSC Campus Itajaí, como requisito parcial para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Me. Ênio dos Santos Silva
Instituto Federal de Santa Catarina

ITAJAÍ
2025

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca do IFSC.

Oldenburg Vieira, João Antonio
Aprendizado por reforço profundo aplicado à detecção de anomalias / João Antonio Oldenburg Vieira ; orientador, Ênio dos Santos Silva, 2025.
68 p.

Trabalho de Conclusão de Curso (graduação) - Instituto Federal de Santa Catarina, Campus Itajaí, Graduação em Engenharia eletrica, Itajaí, 2025.

Inclui referências.

1. Engenharia eletrica. 2. Aprendizado por reforço profundo. 3. Detecção de anomalias. 4. Inteligência artificial. 5. Redes neurais profundas. I. Silva, Ênio dos Santos. II. Instituto Federal de Santa Catarina. Graduação em Engenharia eletrica. III. Título.

Ministério da Educação
Instituto Federal de Santa Catarina
Campus Itajaí
Coordenação do Curso de Engenharia Elétrica

INSTITUTO FEDERAL
Santa Catarina
Câmpus Itajaí

TERMO DE APROVAÇÃO

Título do Trabalho de Conclusão de Curso

Aprendizado por Reforço Profundo Aplicado à Detecção de Anomalias

por

João Antônio Oldenburg Vieira

Esse Trabalho de Conclusão de Curso foi apresentado às **16h30 do dia 18 de Fevereiro de 2025** como **requisito parcial** para a obtenção do título de **Bacharel em Engenharia Elétrica**. Após deliberação da Banca Examinadora, composta pelos professores abaixo assinados, o trabalho foi considerado **APROVADO**.

gov.br Documento assinado digitalmente
FERNANDA ISABEL MARQUES ARGOUD DA SILV
Data: 31/03/2025 13:11:42-0300
Verifique em https://validar.iti.gov.br

Prof(a). Dr(a). Fernanda Isabel Marques Argoud da Silva

gov.br Documento assinado digitalmente
SERGIO AUGUSTO BITENCOURT PETROVCIC
Data: 31/03/2025 18:09:03-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Sergio Augusto Bitencourt Petrovcic

gov.br Documento assinado digitalmente
ENIO DOS SANTOS SILVA
Data: 01/04/2025 17:56:04-0300
Verifique em https://validar.iti.gov.br

Prof. Me. Ênio dos Santos Silva
Prof. Orientador
IFSC

O termo de aprovação assinado encontra-se na coordenação do curso

Itajaí, 18 de Fevereiro de 2025

AGRADECIMENTOS

À minha mãe, que tornou possível minha formação, dedicando seu tempo e esforço para que eu pudesse me dedicar ainda mais aos estudos.

À minha esposa, maior incentivadora, cujo apoio constante, paciência e motivação foram fundamentais ao longo desta jornada. Obrigado por acreditar em mim e estar sempre ao meu lado.

E ao meu orientador, que demonstra uma paciência ímpar e uma dedicação admirável à profissão. Sou grato por todo o conhecimento e apoio compartilhado.

RESUMO

A transformação digital e o aumento exponencial no volume de dados gerados diariamente têm impulsionado um cenário global orientado por dados, onde as empresas buscam cada vez mais extrair valor estratégico a partir de grandes conjuntos de informações. Nesse contexto, técnicas de aprendizado de máquina e redes neurais se tornaram ferramentas essenciais para processar e interpretar esses dados de forma eficiente. Nesse cenário, a detecção de anomalias, crucial para identificar padrões incomuns, desempenha um papel fundamental e destaca-se como uma das áreas de maior relevância. Visando a investigação e identificação de comportamentos anômalos em séries temporais, este trabalho estuda a aplicação de redes neurais profundas aliadas a técnicas de aprendizado por reforço. Entre as técnicas estudadas, destaca-se o Double Deep Q-Network (DDQN), que se mostra eficaz em cenários que demandam a generalização de grandes volumes de dados. Particularmente, utilizando o conjunto de dados “Numenta Anomaly Benchmark” (NAB), este trabalho de conclusão de curso explora dados reais e investiga o potencial do DDQN para detectar anomalias. Os resultados reforçam não apenas a contribuição acadêmica dessa abordagem, mas também seu impacto prático no desenvolvimento de soluções robustas para problemas críticos em ambientes dinâmicos e complexos.

Palavras-chave: Aprendizado por reforço profundo, detecção de anomalias, inteligência artificial, redes neurais profundas, séries temporais.

ABSTRACT

The digital transformation and the exponential growth of data generated daily have driven a global data-driven landscape, where companies increasingly seek to extract strategic value from large datasets. In this context, machine learning techniques and neural networks have become essential tools for efficiently processing and interpreting data. Anomaly detection, crucial for identifying unusual patterns, plays a fundamental role and is a highly relevant research area. To investigate and identify anomalous behaviors in time series, this work explores the application of deep neural networks combined with reinforcement learning techniques. Among the studied approaches, the Double Deep Q-Network (DDQN) stands out as an effective method in scenarios requiring the generalization of large datasets. Using the “Numenta Anomaly Benchmark” (NAB) dataset, this study investigates the effectiveness of DDQN for anomaly detection. The results highlight not only the academic contribution of this approach but also its practical impact on developing robust solutions for critical problems in dynamic and complex environments.

Keywords: Deep reinforcement learning, anomaly detection, artificial intelligence, deep neural networks, time series.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BIP-GARCH	*Bounded Innovation Propagation* GARCH
DDQN	*Double DQN*
DQN	*Deep Q-Network*
DRL	*Deep Reinforcement Learning*
FN	Falso Negativo
FP	Falso Positivo
GARCH	Generalized Autoregressive Conditional Heteroskedasticity
IA	Inteligência Artificial
MDP	*Markov decision process*
ML	*Machine Learning*
MSE	*Mean squared error*
NAB	*Numenta Anomaly Benchmark*
RL	Reinforcement Learning
SARSA	*State-Action-Reward-State-Action*
SVM	*support vector machine*
VP	Verdadeiro Positivo

LISTA DE SÍMBOLOS

t_2	Instante de tempo t_2
$\mathcal{N}$	Distribuição Normal
μ	Média
σ	Desvio padrão
$h(\theta,x)$	Função hipótese para o mapeamento da entrada x em função de parâmetros θ
σ^2	volatilidade (variância) condicional
y_t	Retorno observado no tempo t
z_t	Inovações normalmente distribuídas
a_t	Componente de salto no tempo t
N_1	Estado 1 normal
A_2	Estado 2 com anomalia
$\mathbf{o}_t$	Vetor de observação (*frame*) no tempo t
S_1	Estado 1
S_2	Estado 2
$b_{\{.\}}(\mathbf{o}_t)$	Probabilidade de emissão de $\mathbf{o}_t$ em um determinado estado $S_{\{.\}}$
k_i	Peso da i-ésima mistura de gaussina
a_{ij}	Probabilidade de transição do estado S_i para S_j
$P(\mathbf{o}_t\|N_1)$	Probabilidade do estado normal N_1 ter gerado a observação $\mathbf{o}_t$
$P(\mathbf{O}\|A_2)$	Probabilidade do estado anômalo A_2 ter gerado a observação $\mathbf{o}_t$
$\mathbf{S}_t$	Vetor que representa o Estado no tempo t
A_t	Vetor que representa o espaço de Ações no tempo t
r	Valor de recompensa
π	Política de decisão em um MDP
V^π	Valor esperado da soma de todas as recompensas obtidas pelo agente após tomar as ações “ótimas” apresentadas na Política π
$\mathbb{Z}$	Conjunto dos números inteiros
Q^*	Função Q-*Learning*
Q^π_{tar}	Função Q da Rede Neural
Q^π_{target}	Função Q da Rede Neural Alvo

SUMÁRIO

1 Introdução

Conforme revela o Relatório Global de Adoção de Inteligência Artificial (IA) da IBM, publicado em 2022, a IA tem sido cada vez mais integrada aos modelos de negócios de empresas ao redor do mundo, provocando profundas mudanças no cenário industrial global (WATSON, 2022). A incorporação dessa tecnologia abrange uma ampla gama de áreas, incluindo análise de desempenho de funcionários e perfil de contratação, análise de dados e otimização da tomada de decisões, entre outras. Essa evolução evidencia uma transformação significativa nos mercados e processos corporativos que prevaleceram até os anos mais recentes (WATSON, 2022).

Nesse contexto, até mesmo áreas bastante estudadas e fundamentadas matematicamente, como a detecção de anomalias, estão passando por uma transformação expressiva. Técnicas da estatística tradicional, algumas das mais antigas empregadas para identificar anomalias (CHANDOLA; BANERJEE; KUMAR, 2012), estão agora sendo aprimoradas e complementadas por novas abordagens que incorporam métodos de aprendizado de máquina (INJADAT et al., 2018). O desafio de reconhecer padrões que divergem do comportamento esperado, denotada detecção de anomalias, tem se tornado cada vez mais relevante no cenário industrial atual (CHANDOLA; BANERJEE; KUMAR, 2009), (INJADAT et al., 2018). Essa crescente importância impulsiona o desenvolvimento de soluções mais eficazes, abrindo caminho para avanços importantes na área. Assim, a detecção de anomalias continua a se beneficiar dessa crescente demanda e, especialmente quando combinada com técnicas de aprendizado de máquina [*Machine Learning* (ML)], desempenha um papel central no estado da arte da área.

Muitos sistemas atuais baseados em detecção de anomalias buscam identificar comportamentos fora dos padrões esperados, mas frequentemente caracterizam sinais normais como eventos anômalos, o que define um erro chamado de falso positivo (JUVONEN; SIPOLA; HäMäLäINEN, 2015). Solucionar problemas de detecção de anomalias associados com baixas ocorrências de falsos positivos vem se tornando particularmente relevante em setores como o financeiro e o de saúde. Nos Estados Unidos, por exemplo, a *National Health Care Anti-Fraud Association* estima que mais de 60 bilhões de dólares sejam perdidos anualmente devido a fraudes, enquanto, segundo a revista EXAME, fraudes envolvendo PIX resultaram em prejuízos de R$2,5 bilhões

em 2022 (NHCAA, 2023), (LOPES, 2023). Esses números evidenciam como pequenas melhorias na precisão desses sistemas podem ter um impacto significativo, prevenindo perdas substanciais.

Pensando em evitar mais perdas, algoritmos de detecção de anomalias baseados em ML vêm sendo amplamente adotados em diversas aplicações. Entre os exemplos importantes estão a detecção de invasões aplicada na área de segurança cibernética, identificação de falhas em *streaming*, monitoramento de tráfego anômalo para revelar conjuntos de dados comprometidos e identificação de fraudes no setor financeiro, especialmente em bancos (*fintechs*) (ARSHAD et al., 2022). As técnicas de detecção de anomalias também têm desempenhado um papel fundamental na melhoria da qualidade de software, dada a crescente preocupação com os custos decorrentes de *bugs* em sistemas digitais (YANAI, 2020).

Nas aplicações supracitadas diversas estratégias de aprendizado em ML vêm sendo utilizadas, dentre elas, destacam-se o aprendizado supervisionado, o não supervisionado e o aprendizado por reforço [*Reinforcement Learning* (RL)]. Particularmente, o RL é uma abordagem de ML em que um agente, ao interagir com um ambiente, busca otimizar seu comportamento para maximizar recompensas ao longo do tempo, e o aprendizado por reforço profundo [*Deep Reinforcement Learning* (DRL)] é o aprimoramento desta técnica. Atualmente, DRL representa o estado da arte em RL, pois apresenta eficácia em conjuntos de dados extensos e é aplicável em diversas áreas, abrangendo detecção de anomalias, videogames, robótica, transporte, processamento de linguagem natural, saúde, visão computacional, finanças, dentre outras (ARSHAD et al., 2022). Nos últimos anos, o uso de DRL em detecção de anomalias tem superado outras técnicas de aprendizado profundo, principalmente com o uso de técnicas de *Deep Q-Network* (DQN) (ARSHAD et al., 2022), subárea e técnica em ascensão do DRL.

No cenário atual, marcado pela geração contínua e massiva de dados, o desafio de identificar padrões anômalos com rapidez e precisão torna-se cada vez mais crítico (SILVA, 2020). Nessa circunstância, este trabalho de pesquisa objetiva aplicar técnicas avançadas de DRL para otimizar a detecção de anomalias em dados complexos e dinâmicos. Especificamente, o foco está na aplicação do DQN, avaliando sua eficácia na identificação de anomalias em grandes volumes de dados. Ao explorar o potencial dessa técnica, o estudo pretende identificar desafios e oferecer contribuições relevantes

para o avanço da detecção eficiente de padrões anômalos.

Particularmente, o DQN representa um avanço significativo no campo da detecção de anomalias devido à sua capacidade de aprender e se adaptar a padrões não convencionais através da interação contínua com o ambiente (ARSHAD et al., 2022). Ao permitir um comportamento flexível e adaptativo, o DQN oferece uma abordagem diferente das técnicas tradicionais que frequentemente se baseiam em regras estáticas e pré-definidas. A aplicação prática dessas técnicas avançadas pode resultar em economias substanciais de recursos financeiros, uma vez que a detecção precoce de anomalias permite a antecipação e mitigação de potenciais ameaças antes que causem danos significativos

Além de oferecer uma solução técnica robusta, a aplicação do DQN neste contexto contribui academicamente ao explorar novas fronteiras no aprendizado profundo. Ao preencher uma lacuna de conhecimento sobre o uso do DRL para detecção de anomalias, este estudo apresenta características importantres sobre a eficácia dessa abordagem em diferentes cenários e com dados complexos. Espera-se que a compreensão aprofundada dessas técnicas não só impulsionará avanços acadêmicos, como também servirá de base sólida para futuras implementações práticas, respondendo às crescentes demandas contemporâneas na área de detecção de anomalias.

Neste trabalho, conceitos amplamente discutidos no estado da arte da área de ML são simplificados e incorporados, com o objetivo de tornar as explicações mais acessíveis e compreensíveis. Além disso, propõe-se um *framework* que poderá ser utilizado em pesquisas futuras, viabilizando análises mais aprofundadas em tópicos como arquitetura de redes neurais, dados de entrada discriminativos e configurações otimizadas de hiperparâmetros. O caráter pioneiro da abordagem de DRL na universidade, aliado aos resultados esperados, ressalta a relevância científica e o impacto potencial deste trabalho de conclusão de curso no avanço das pesquisas em ML.

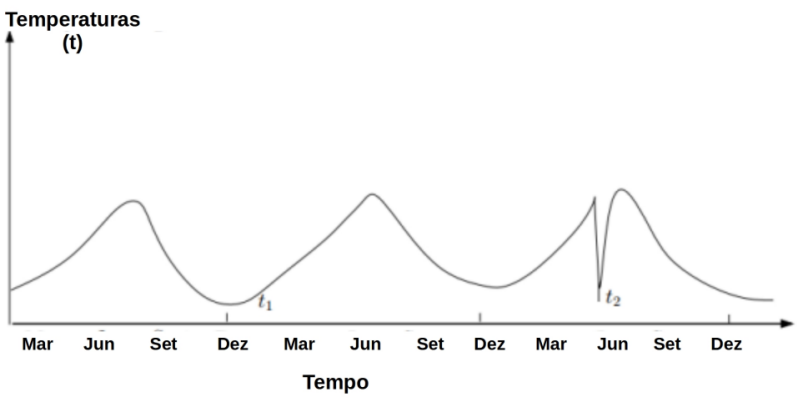
2 Referencial Teórico

2.1 Detecção de anomalias em séries temporais

As anomalias em séries temporais correspondem a registros que se desviam de forma significativa do comportamento esperado dos dados ao longo do tempo. A identificação dessas ocorrências é essencial para a análise de séries temporais, pois elas podem sinalizar eventos raros e relevantes ou indicar problemas na coleta de dados. Dessa forma, a detecção de anomalias desempenha um papel crucial em diversas aplicações, destacando-se como uma área de grande relevância em contextos como segurança, saúde e análise de dados (MOLINA, 2022). Essa relevância é ainda maior para as várias áreas onde é essencial identificar comportamentos inesperados e discrepantes em relação ao comportamento padrão (BOZZETTO, 2023). Assim, a detecção de anomalia em séries temporais encontra aplicações em diversas situações como a detecção de invasões em redes de comunicação, identificação de fraudes em diferentes contextos, monitoramento de processos para prevenção de falhas, suporte a diagnósticos médicos, aplicações em ciências gerais e processamento de sinais, entre outros (RUFF et al., 2021).

De acordo com a literatura, as anomalias em séries temporais são classificadas como: pontuais, contextuais ou coletivas (CHANDOLA; BANERJEE; KUMAR, 2009). As séries temporais pontuais são pontos de dados isolados que se desviam do padrão geral. Por exemplo, uma queda abrupta e isolada na temperatura registrada por um determinado sensor, como a observada no instante de tempo t_2 da série apresentada na Figura 1.

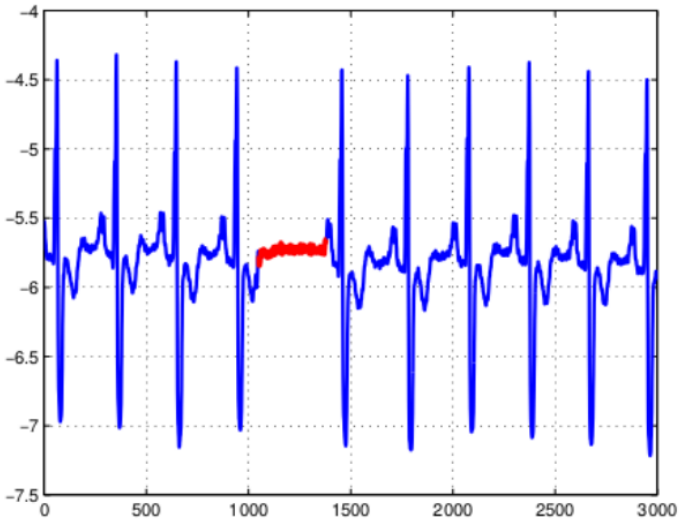
Figura 1 – Anomalia pontual.



Fonte: (BOZZETTO, 2023).

As séries temporais contextuais são observações que parecem normais em um contexto, mas não em outro, como, por exemplo, registros de altas temperaturas em meses frios. Já as anomalias coletivas envolvem um conjunto de dados que juntos desviam do padrão estabelecido, a Figura 2 ilustra um exemplo de série temporal com anomalia coletiva. Essas classificações de anomalias ajudam a discriminar e entender variações significativas em dados ao longo do tempo.

Figura 2 – Anomalia coletiva.



Fonte: (BOZZETTO, 2023).

Conhecer a teoria sobre anomalias em séries temporais é importante para o desenvolvimento de tecnologias de detecção de anomalias em tempo real. Nesse contexto, detectar anomalias em tempo real pode antecipar eventos que necessitem de intervenção ou controle rápido como em aplicações no mercado financeiro, diagnósticos de sistemas e controle de ações de usuários. No mercado financeiro, detectar rapidamente eventos como, por exemplo, a queda de 6 de maio de 2010, é fundamental para evitar grandes perturbações devido a fraquezas nos sistemas de negociação (GUPTA et al., 2014). Na área de diagnósticos de sistemas, aplicações destinadas à supervisão de condições de funcionamento de aeronaves ou sistemas de detecção de intrusões baseados em *hosts* revelam eventos de falha ou ameaças. Por último, sequências de ações de usuários, como padrões de navegação na web ou transações de clientes, podem indicar comportamentos anômalos, como tentativas de quebra de senha (GUPTA et al., 2014).

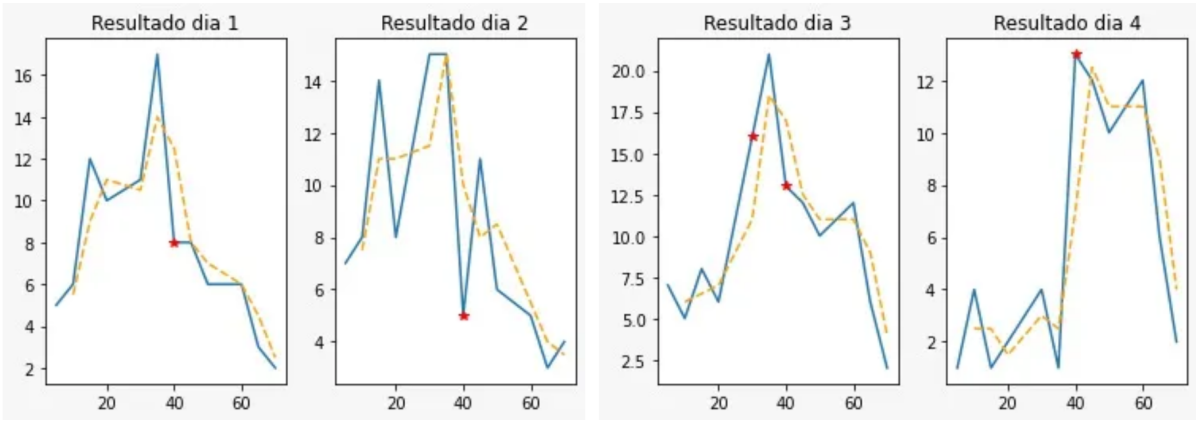
2.1.1 Detecção de anomalias utilizando métodos estatísticos

Os métodos estatísticos desempenham um papel fundamental na detecção de anomalias, identificando pontos que se desviam das propriedades estatísticas particulares de um conjunto de dados, como média, mediana, moda e quantis. Essas técnicas podem ser classificadas em dois tipos principais: paramétricas e não paramétricas (HAN; PEI; KAMBER, 2011).

Os métodos paramétricos avaliam hipóteses sobre parâmetros específicos, como média e desvio-padrão, assumindo que os dados seguem uma distribuição pré-definida. Por outro lado, os métodos não paramétricos não consideram suposições estritas sobre a distribuição dos dados. Particularmente, tais métodos avaliam hipóteses sobre a forma da distribuição dos dados [por exemplo, $x \sim \mathcal{N}(\mu, \sigma)$] e sobre a relação [mapeamento $y = h(\theta,x)$] entre os dados e suas correspondentes classes (y) (HAN; PEI; KAMBER, 2011).

Um exemplo clássico de método estatístico é a média móvel simples, que pode ser definida matematicamente como um filtro passa baixa. Este método suaviza os valores de uma série temporal, filtrando o ruído causado por flutuações de curto prazo e destacando tendências ou ciclos de longo prazo. A operação de um filtro passa baixa é ilustrada na Figura 3, onde a linha laranja (- -) representa a média móvel calculada e os pontos em vermelho ($\star$) representam as anomalias identificadas. Embora eficaz em alguns casos, essa técnica pode falhar em identificar anomalias quando os dados forem corrompidos por ruído cuja distribuição seja semelhante a um determinado comportamento anômalo.

Figura 3 – Modelo estatístico: média móvel.



Fonte: (GALVãO, 2024).

Com base em modelos matemáticos consagrados e estabelecidos na literatura, o uso de métodos estatísticos apresenta alta confiabilidade em cenários bem comportados (TAN; STEINBACH; KUMAR, 2009). No entanto, em séries temporais com padrões complexos de sazonalidade, torna-se necessário utilizar métodos mais sofisticados que decomponham os dados (sinais analisados) em várias tendências para melhor identificar mudanças sazonais e detectar anomalias de forma mais eficaz.

2.1.2 Volatilidade condicional e desvio padrão condicional

A volatilidade condicional (σ^2) é um conceito estatístico utilizado para medir a incerteza futura sobre variáveis que dependem de dados históricos e de outras variáveis observadas (BOUDT; DANíELSSONB; LAURENT, 2013). Esse conceito é amplamente utilizado na área de finanças, sendo fundamental para a análise de séries temporais financeiras, pois captura a natureza dinâmica e variável dos retornos de ativos. Sua importância está relacionada à modelagem e previsão de riscos associados aos investimentos. A volatilidade condicional é especialmente relevante quando a variância dos retornos não é constante ao longo do tempo (comportamento heterocedástico) (BOUDT; DANíELSSONB; LAURENT, 2013).

Um dos modelos mais utilizados para estimar a volatilidade condicional é o modelo Heterocedasticidade Condicional Autorregressiva Generalizada [*Generalized Autoregressive Conditional Heteroskedasticity* (GARCH)]. O modelo GARCH, introduzido por (BOLLERSLEV, 1986), é amplamente adotado devido à sua capacidade de capturar a persistência da volatilidade e pela sua simplicidade computacional. A realização básica do GARCH é dada por:

$$\sigma_t^2 = \omega + \alpha_1 y_{t-1}^2 + \beta_1 \sigma_{t-1}^2, \tag{1}$$

onde σ_t^2 é a variância condicional no tempo t; ω, α_1 e β_1 são parâmetros do modelo; e y_{t-1} é o retorno no período anterior.

Dessa forma, o desvio padrão condicional (σ) é diretamente obtido da volatilidade condicional (σ^2): $\sigma = \sqrt{\sigma^2}$. O desvio padrão condicional fornece uma medida do risco em termos absolutos, sendo utilizado para calcular intervalos de confiança e identificar eventos extremos nos retornos de ativos (BOUDT; DANíELSSONB; LAURENT, 2013).

A volatilidade condicional é uma ferramenta eficaz para a detecção de eventos inesperados ou mudanças estruturais em séries temporais financeiras, grandes saltos ou quedas de magnitudes, chamados *outliers*, um tipo de anomalia no contexto desse trabalho. O modelo GARCH pode ser construído para identificar essas alterações drásticas de valores, por desviarem significativamente da volatilidade esperada.

Dada a importância do modelo GARCH, diversas extensões têm sido investigadas na literatura, como por exemplo, o modelo Propagação de Inovação Limitada GARCH [*Bounded Innovation Propagation* GARCH (BIP-GARCH)] proposto para melhorar a robustez da previsão de volatilidade, limitando o impacto de inovações extremas (BOUDT; DANíELSSONB; LAURENT, 2013). Este modelo considera a presença de saltos que são distribuídos de acordo com uma distribuição de Poisson, o que permite a detecção de anomalias através da identificação de componentes de salto nos retornos:

$$y_t = \sqrt{\sigma^2} z_t + a_t, \tag{2}$$

onde y_t é o retorno observado no tempo t, z_t são inovações normalmente distribuídas, e a_t representa o componente de salto.

Ao fornecer uma medida dinâmica do risco, essas variáveis possibilitam que os modelos de ML ajustem suas previsões de acordo com as mudanças na incerteza associada ao objeto previsto. Isso é importante para a construção de modelos preditivos robustos que possam responder eficientemente a eventos extremos e mudanças de regime nos dados.

Os modelos de volatilidade condicional, como o GARCH, e suas extensões, oferecem uma base sólida para a detecção de *outliers* e a previsão de risco, servindo como uma etapa preliminar (processamento dos dados de entrada e geração de novos dados discriminativos) essencial nas aplicações de ML definidas no escopo desse trabalho.

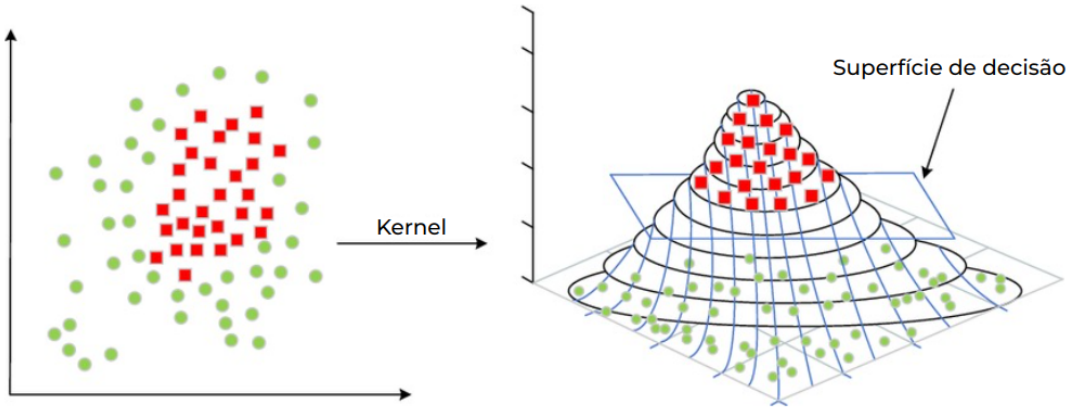
2.1.3 Detecção de anomalias usando máquina de vetores de suporte

A Máquina de Vetores de Suporte [*support vector machine* (SVM)] é um método de aprendizado amplamente utilizado em problemas de classificação e regressão (CHERKASSKY, 1997). O objetivo principal da SVM é encontrar um hiperplano que melhor separe os dados em diferentes classes, maximizando a margem entre os exemplos mais próximos de cada classe (CHERKASSKY, 1997). Esse princípio permite

ao algoritmo criar modelos robustos para tarefas em que a separação entre as classes é claramente definida.

Nos casos em que os dados não são linearmente separáveis no espaço original, a SVM utiliza técnicas baseadas em funções *kernel*. Essas funções projetam os dados em espaços de maior dimensão, onde se torna possível encontrar um hiperplano separador, conforme ilustrado na Figura 4. Dentre os kernels mais utilizados, destacam-se o linear, polinomial, radial e sigmoid, cada um adequado para diferentes tipos de distribuição de dados. Essa flexibilidade faz da SVM uma ferramenta poderosa para lidar com dados complexos e com múltiplas variáveis.

Figura 4 – Modelo estatístico: Suport Vector Machine.



Fonte: Adaptado de (YU; ZHANG; GAO, 2023).

Embora a SVM seja geralmente associada ao aprendizado supervisionado, existem extensões, como o One-Class SVM, que podem ser aplicadas em problemas não supervisionados para identificar anomalias, utilizando dados de treinamento não rotulados. Esse algoritmo aprende a delinear uma fronteira flexível para agrupar instâncias de dados não anômalos a partir do conjunto de treinamento e, posteriormente, ajusta-se durante a fase de teste para detectar anomalias que estão fora da região aprendida (MARTINS, 2020).

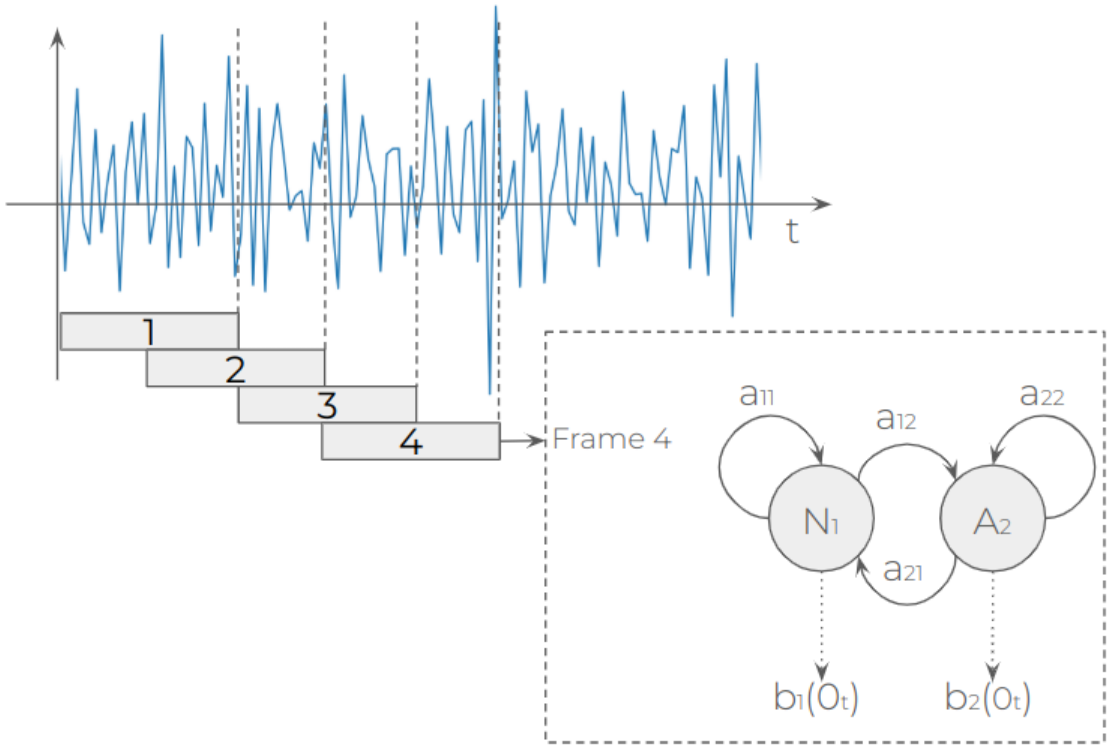
2.1.4 Detecção de anomalias usando autômatos

Autômatos podem ser considerados modelos matemáticos que representam sistemas discretos. Na detecção de anomalia, os autômatos podem ser configurados

para modelar (ou identificar) o comportamento normal e/ou anormal de um determinado sistema.

Visando exemplificar (de forma resumida) os conceitos envolvidos sobre modelos autômatos, a Figura 5 ilustra as operações realizadas em um sistema de detecção de anomalias usando autômatos.

Figura 5 – Segmentação de sinal (*frame*) e autômato de representação de estados e transições.



Fonte: Elaboração própria.

Nesse cenário, o sistema é modelado com dois estados: N_1 correspondendo ao estado normal e A_2 correspondendo ao estado com anomalia. Além disso, considerando o sinal de entrada $x(t)$, a detecção de anomalias ocorre a partir da análise de segmentos (*frame*) $\mathbf{o}_t$ de $x(t)$. Dessa forma, considera-se que cada segmento $\mathbf{o}_t$ contém propriedades estatísticas suficientes sobre a distribuição de $x(t)$. Então, uma rede de autômatos pode ser projetada para calcular (estimar) se o *frame* $\mathbf{o}_t$ foi gerado por um estado normal N_1 (S_1) ou por um estado anômalo A_2 (S_2) (GOERNITZ et al., 2015). Assim, cada estado apresenta uma probabilidade de emissão $b_{\{.\}}(\mathbf{o}_t)$ do *frame* observado $\mathbf{o}_t$ (probabilidade do estado $S_{\{.\}}$ ter gerado o vetor de observação $\mathbf{o}_t$). Essa probabilidade de emissão pode ser calculada, por exemplo, por um modelo de misturas

de gaussianas (DUDA, 2001):

$$b_{\{.\}}(\mathbf{o}_t) = \sum_i k_i \mathcal{N}(\mathbf{o}_t; \mu_i, \sigma_i^2) \tag{3}$$

Onde k_i são os pesos para cada mistura de gaussiana e $\mathcal{N}(.; \mu_i, \sigma_i^2)$ é a função gaussiana com média μ_i e variância σ_i^2 expressa por:

$$\mathcal{N}(\mathbf{o}; \mu, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)}} e^{-\frac{(\mathbf{o}-\mu)^2}{2\sigma^2}}. \tag{4}$$

Finalmente, considerando os coeficientes de transição a_{ij} entre estados, a probabilidade $P(\mathbf{o}_t|N_1)$ do estado normal N_1 ter gerado a observação $\mathbf{o}_t$ (ou a probabilidade da observação ser normal) pode ser computada por:

$$P(\mathbf{o}_t|N_1) = \pi_1 a_{11} b_1(\mathbf{o}_t) + \pi_2 a_{21} b_1(\mathbf{o}_t) \tag{5}$$

Da mesma forma, a probabilidade $P(\mathbf{O}|A_2)$ do estado anômalo A_2 ter gerado a observação $\mathbf{o}_t$ (ou a probabilidade da observação ser anômala) pode ser computada por:

$$P(\mathbf{O}|A_2) = \pi_1 a_{12} b_2(\mathbf{o}_t) + \pi_2 a_{22} b_2(\mathbf{o}_t) \tag{6}$$

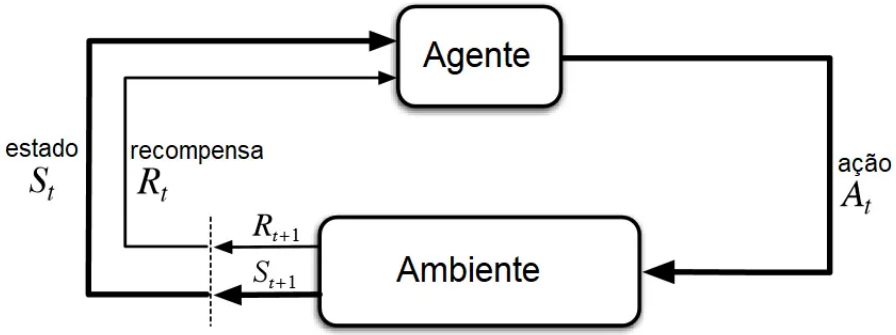
Onde, π_1 e π_2 representam as probabilidades iniciais da observação (quadro) $\mathbf{o}_{t-1}$ pertencer ao estado N_1 [$P(\mathbf{o}_{t-1} \in N_1)$] e ao estado A_2 [$P(\mathbf{o}_{t-1} \in A_2)$], respectivamente. Nesse técnica de aprendizado, os parâmetros do modelo (k, μ, σ^2, a e π) são estimados (treinados) através de algorítimos do arcabouço estatístico (como o algorítimo EM, *Baum-Welch*, *Viterbi*, entre outros) (DUDA, 2001).

2.1.5 Detecção de anomalias usando aprendizado por reforço

Esta seção apresenta uma breve introdução sobre a aplicação de RL no contexto de detecção de anomalias. Nas seções seguintes, os conceitos fundamentais e as técnicas associadas à RL são apresentados de forma mais detalhada. Para ilustrar o funcionamento do modelo de RL, a Figura 6 é apresentada. Em um modelo de RL, um agente interage com um determinado ambiente, observando um determinado estado S_t e recebendo uma recompensa ou penalidade R_t a cada ação A_t executada. Assim, o agente aprende com suas experiências e aprimora suas decisões ao longo do tempo.

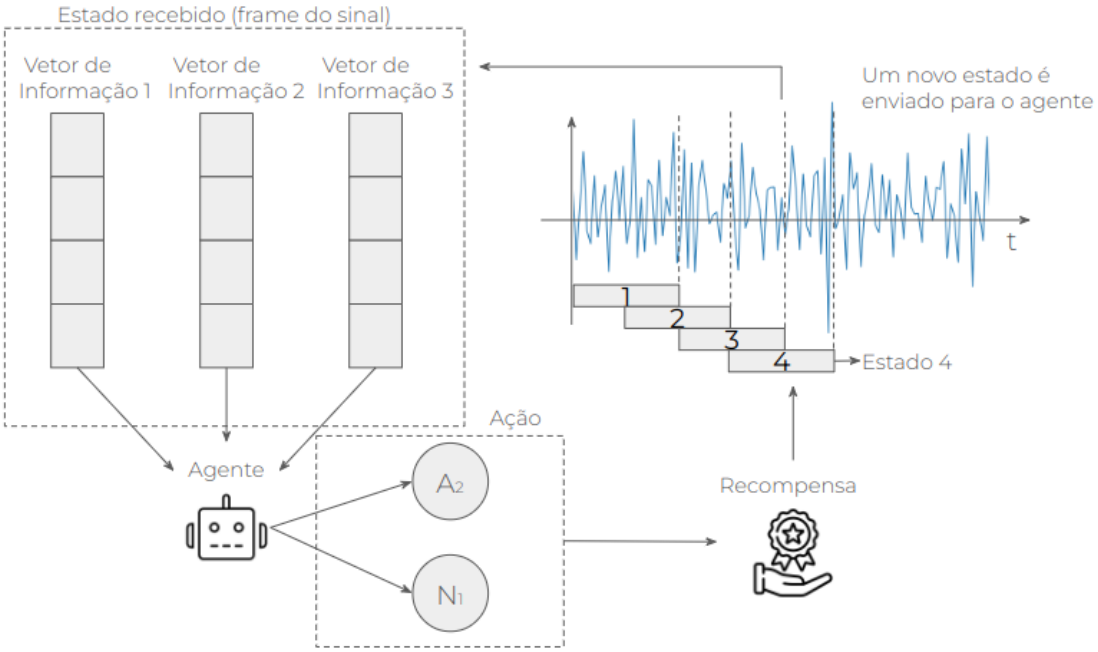
Visando ilustrar a aplicação de modelos de RL em sistemas de detecção de anomalias, a Figura 7 é mostrada.

Figura 6 – Ilustração de um modelo de RL.



Fonte: Adaptado de (SUTTON; BARTO, 2018).

Figura 7 – Diagrama de funcionamento do aprendizado por reforço.



Fonte: Elaboração própria.

Nesse cenário, o sistema é configurado com duas ações possíveis: N_1, que representa a classificação como normal, e A_2, que corresponde a classificação como anomalia. A abordagem de duas ações possíveis é comumente utilizada devido a sua simplicidade (LI; WU, 2022). O processo de detecção de anomalias se inicia com o sinal de entrada $x(t)$, que é analisado por meio de segmentos (*frames*) $\mathbf{o}_t$. Visando a obtenção de dados discriminativos (que facilitem a tomada de decisão sobre $\mathbf{o}_t \in N_1$ ou $\mathbf{o}_t \in A_2$), tais segmentos podem ser pré-processados para extrair características adicionais, como a variação temporal (derivada) sinal $\partial \mathbf{o}_t / \partial t$ e o desvio padrão condicional σ_t. Em seguida, essas informações são vetorizadas e constituem

o estado S_t do ambiente no instante t. Assim, o vetor de estado $\mathbf{S}_t = [\mathbf{o}_t, \partial\mathbf{o}_t/\partial t, \sigma_t]$ (contendo informações relevantes sobre o ambiente atual) é fornecido ao agente. Em resumo, consideram-se as seguintes premissas:

1. Ambiente: Processo de detecção de anomalias;
2. Espaço de estados $\mathbf{S}_t$: Vetor de estado $\mathbf{S}_t = [\mathbf{o}_t, \partial\mathbf{o}_t/\partial t, \sigma_t]$;
3. Espaço de ações A_t: Classificar o segmento $\mathbf{o}_t$ como normal ou anomalia, $A_t = [N_1, A_2]$.

Particularmente, considera-se que cada vetor de estado $\mathbf{S}_t$ contem propriedades estatísticas suficientes para descrever as características de $x(t)$ em um dado segmento $\mathbf{o}_t$. Dessa forma, o RL é projetado para classificar se o segmento $\mathbf{o}_t$ resulta na ação N_1 (classificação de $\mathbf{o}_t$ como normal) ou na ação A_2 (classificação de $\mathbf{o}_t$ como anomalia). Assim, cada estado fornece informações suficientes para a tomada de decisão, permitindo que o agente determine se uma anomalia está presente.

O mecanismo de recompensas funciona como balizador da decisão tomada, guiando as decisões do agente (YU; SUN, 2020). A recompensa reflete a qualidade da ação tomada, penalizando erros com valores negativos ou recompensas reduzidas, enquanto reforça decisões corretas com ganhos positivos. Esse *feedback* torna o RL uma técnica de aprendizado semi-supervisionado, pois depende tanto de explorações iniciais quanto do aprimoramento baseado em experiência (YU; SUN, 2020).

Ao buscar maximizar a soma das recompensas ao longo do tempo, o agente ajusta suas estratégias para tomar ações que conduzam à maior recompensa possível. No contexto de detecção de anomalias, isso significa classificar corretamente os segmentos $\mathbf{o}_t$ como normais ou anômalos, otimizando continuamente sua precisão ao longo dos estados observados.

De maneira geral, os processos de aprendizado de máquina têm como objetivo realizar a classificação a partir de sequências temporais de dados. Tais sequências podem ser modeladas, por exemplo, por autômatos ou por RL. No contexto de RL, o processo de classificação é associado a um mecanismo de sequência temporal guiado por um sistema de recompensas. A Tabela 1 ilustra as principais diferenças entre os processos de aprendizado baseados em autômatos e os sistemas que utilizam RL.

Tabela 1 – Considerações sobre detecção de anomalias usando autômatos e RL

Considerações	**Detecção de Anomalias usando autômatos**	**Detecção de Anomalias usando aprendizado por reforço**
Modelo de Abordagem	Baseado em probabilidades e transições de estados ocultos.	Baseado em um agente que aprende interagindo com o ambiente através de recompensas.
Representação dos Dados	Representa dados como sequências de observações em estados ocultos.	Modela a sequência temporal e a dinâmica do ambiente com um agente interativo.
Treinamento	Treinamento supervisionado (ou não supervisionado) usando dados históricos.	Treinamento baseado em interações com o ambiente e feedback de recompensas.
Capacidade de Generalização	Limitado a sequências que seguem a mesma estrutura dos dados de treinamento.	Capacidade de generalizar em ambientes dinâmicos com diversas recompensas e penalidades.
Complexidade Computacional	Relativamente baixa, dependente de número de estados e observações.	Alta, requer recursos computacionais consideráveis para otimizar redes neurais profundas.
Adaptabilidade	Adaptável a novos dados, mas pode ter dificuldades em ambientes dinâmicos.	Alta adaptabilidade a ambientes em constante mudança.
Exigência de Dados Rotulados	Requer grandes quantidades de dados rotulados ou sequências para treinamento.	Pode funcionar de forma não supervisionada ou com dados rotulados limitados.
Recompensas e Penalidades	Não utiliza recompensas explícitas; depende das transições de estados.	Recompensas explícitas são usadas para orientar o aprendizado e otimizar a política de decisão.
Interpretação dos Resultados	Fácil interpretação das transições entre estados e probabilidades.	Resultados mais difíceis de interpretar devido à complexidade do modelo e rede neural.

Fonte: Elaboração própria.

3 Aprendizado por Reforço Profundo

Uma abordagem moderna para a detecção de anomalias em séries temporais é o uso de aprendizado por reforço profundo (DRL). O DRL combina *deep learning* com RL para criar modelos que podem aprender a identificar padrões complexos em grandes volumes de dados e adaptar-se a mudanças ao longo do tempo (GRAESSER; KENG, 2020). A aplicação de DRL na detecção de anomalias é promissora devido à sua capacidade de lidar com a alta dimensionalidade e a variabilidade dos dados temporais, além de realizar a detecção em tempo real. O DRL permite que o sistema aprenda e melhore continuamente, ajustando-se automaticamente a novos padrões de anomalias conforme eles surgem. Essa adaptabilidade é fundamental para manter a eficácia da detecção de anomalias em ambientes dinâmicos (ARSHAD et al., 2022).

Dependendo do caso de uso, a saída de um detector de anomalias pode ser representada por valores escalares numéricos, usados para delimitar valores máximos e mínimos dentro do sinal, ou por rótulos textuais (como binário ou multiclasse) (ARSHAD et al., 2022). O DRL vem como solução para limitações encontradas em outras técnicas eficazes apenas para identificar anomalias em situações simples que apresentam limitações em certos cenários, como por exemplo, quando o ruído nos dados pode se assemelhar ao comportamento anômalo, dificultando a distinção clara entre comportamento normal e anormal, também quando a definição de comportamento normal ou anômalo pode mudar com o tempo (ARSHAD et al., 2022), especialmente em contextos onde, por exemplo, fraudes se adaptam constantemente, tornando o uso de limites fixos, como a média móvel, insuficiente. Além das situações supramencionadas, para dados com sazonalidade, é necessário utilizar métodos mais sofisticados, que decompõem os dados em várias tendências para identificar mudanças na sazonalidade.

3.1 Aprendizado por reforço com Q-*learning*

Particularmente, a técnica de RL é formalmente descrita por um processo de decisão de Markov [*Markov decision process* (MDP)]. Tipicamente, sistemas usando RL são compostos por agentes que atuam de forma dinâmica na exploração de determinados ambientes. Especificamente, a cada iteração (passo) realizada pelo agente,

um novo estado s é assumido. Nesse estado s, o agente executa apenas as ações a permitidas no ambiente.

Neste trabalho, o ambiente proposto é uma série temporal que permite as ações correspondentes à detecção de anomalias. Considerando um processo de decisão de Markov de 1ª ordem, a ação a escolhida pelo agente depende exclusivamente do estado atual s_t, já que o processo de decisão considera que toda a informação relevante do passado ($t-1, t-2, t-3, \ldots$) está contida no estado atual s_t (SUTTON; BARTO, 2018).

Para a detecção de anomalias em séries temporais, cada quadro (*frame*) da série é considerado como um estado s. Em cada estado s é associado um valor de recompensa r que é diretamente relacionado com a precisão da detecção de anomalias. Portanto, um MDP é um processo de recompensa de Markov dependente de decisões. Assim, todos os estados são Markovianos e a detecção de anomalias é dividida em iterações (ou passos). A cada passo, um estado s é assumido e uma nova ação a é tomada considerando apenas o estado atual s_t. O aprendizado por reforço pode ser representado pelo MDP ilustrado na Figura 6. Matematicamente, um MDP é definido como uma tupla (S,A,T,R,γ). Assim como em (SILVA; HEMKEMAIER, 2017), aqui S,A,T,R e γ são definidos como:

- $\mathbf{S}$ é o conjunto de estados finitos. Cada *frame* na série temporal representa um estado diferente;
- $\mathbf{A}$ é o conjunto de ações finitas. Para ocorrer a mudança de estado, o agente deve realizar uma ação de detecção de anomalia ou não;
- $\mathbf{T}$ é a matriz de probabilidades de transição de estados $T(s_{t+1}|s_t,a_t)$. Probabilidade de ir a um estado futuro s_{t+1}, dado que no estado atual s_t será realizada a ação a_t;
- $\mathbf{R}$ é a função de recompensas que atribui o valor da recompensa fornecida ao agente quando o estado s é alcançado, $R(s,a) = E[r|s,a]$;
- γ é um fator de desconto ($\gamma \in [0,1]$) que indica a importância das recompensas em cada estado ao longo de uma sequência.

Então, para a detecção de anomalias em séries temporais contendo vetores de estados finitos (correspondentes a uma determinada quantidade de *frames* obtidos da série temporal) e múltiplas ações, o MDP visa aprender uma política π de decisão que eleve ao máximo o desempenho de uma dada tarefa ao longo do tempo (PORSCH et

al., 2023). Nesse contexto, o objetivo do agente é percorrer a série temporal e identificar o máximo de anomalias (e/ou o mínimo de falsos positivos) possíveis. Para isso, tal agente deve aprender uma política π que determine a ação a "ótima" a ser tomada em cada estado s_t que o levará ao seu estado futuro s_{t+1} (ponto futuro na série), sendo o estado atual s_t estatisticamente suficiente para a estimação (previsão) do estado futuro s_{t+1} (SUTTON; BARTO, 2018). Uma política π pode ser avaliada pela soma total das recompensas de um estado inicial s_0 até o estado atual s_t, como mostrado por V^{π}(s) em (7) (onde γ representa um fator de ponderação aplicado nas recompensas r_{t-i} correspondentes a cada estado s_{t-i}, $\forall i \in \mathbb{Z}$).

$$\begin{aligned} V^{\pi}(s) &= r_t + \gamma r_{t-1} + ... = E\left\{\sum_{i=0}^{\inf} \gamma^i r_{t-i} | s_0 = s, \pi\right\} \\ V^{\pi}(s) &= \sum_{a_t} \pi(s_t, a_t) \sum_{s_t, a_t, s_{t+1}} [R(s_t, a_t, s_{t+1}) + \gamma V^{\pi}(s_{t+1})] \end{aligned} \tag{7}$$

Adicionalmente, a função

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s_{t+1} \in S} T(s_t, a_t, s_{t+1}) V^*(s_{t+1}),$$

é definida para obter o valor atribuído a escolha da ação a no estado s, seguindo a política ótima π^*. Assim, em RL, a estimação da função Q^* consiste no aprendizado de uma solução de um MDP (SUTTON; BARTO, 2018).

3.2 Aprendizado por reforço profundo

O RL tradicional enfrenta limitações significativas quando a quantidade de estados no ambiente é muito grande. Em um processo de MDP, cada estado do ambiente precisa ser explicitamente representado, o que pode levar a uma explosão combinatória à medida que o número de estados aumenta (MNIH et al., 2015). Isso torna o armazenamento e a computação inviáveis, especialmente em ambientes complexos ou de alta dimensionalidade, como séries temporais com múltiplas variáveis ou dados de alta frequência (MNIH et al., 2015).

Nesses casos, o agente de RL pode ter dificuldade em generalizar seu aprendizado para novos estados, pois a política π e a função de valor $V^{\pi}(s)$ precisam ser definidas para cada estado específico. Isso resulta em uma necessidade de grandes quantidades de dados e um tempo de treinamento substancialmente longo para cobrir todos os possíveis estados do ambiente. Além disso, o RL tradicional pode falhar em

capturar a complexidade e a interdependência entre os estados de forma eficiente, levando a um desempenho subótimo.

Para superar essas limitações, o DRL introduz o uso de redes neurais profundas para abstrair e generalizar os estados do ambiente. Em vez de representar explicitamente cada estado, o DRL utiliza uma rede neural para aproximar as funções de valor $V(s)$, de política $\pi(a|s)$ e de ação-valor $Q(s,a)$ (GRAESSER; KENG, 2020). Essas redes neurais são capazes de extrair características relevantes dos estados e ações, permitindo que o agente aprenda representações mais compactas e abstratas dos estados (GRAESSER; KENG, 2020).

A arquitetura de uma rede neural profunda permite a modelagem de relações complexas entre os estados, fornecendo uma capacidade de generalização que é crítica em ambientes de alta dimensionalidade (MNIH et al., 2015). Isso é alcançado através do uso de múltiplas camadas de neurônios que aprendem diferentes níveis de abstração a partir dos dados de entrada. O uso de DRL também facilita a exploração de espaços de estado e ação muito maiores do que seria possível com RL tradicional. A rede neural pode ser treinada com técnicas avançadas como o DQN, que combinam a eficiência computacional com a capacidade de generalização das redes neurais e é foco neste trabalho.

Em suma, o DRL mitiga as limitações do RL tradicional ao abstrair a complexidade dos estados através de redes neurais profundas, permitindo a modelagem e generalização em ambientes de alta dimensionalidade (MNIH et al., 2015). Isso resulta em um aprendizado mais eficiente e eficaz, capaz de lidar com a complexidade intrínseca de problemas reais, como a detecção de anomalias em séries temporais.

3.3 Algoritmo DQN - Deep Q-Network

O DQN é um algoritmo de aprendizado por reforço que combina *Q-Learning* com redes neurais profundas para aprender a função de valor Q (MNIH et al., 2015). A função Q estima a recompensa esperada para cada par estado-ação (s, a), utilizando uma rede neural para aproximar a função Q tradicional. O DQN introduz duas técnicas principais para estabilizar o treinamento: a rede alvo, que fornece uma referência estável para as atualizações, e a memória de replay, que armazena e reutiliza experiências passadas para reduzir correlações temporais. Essas melhorias permitem que o DQN

aprenda políticas eficazes em ambientes complexos com grandes espaços de estados e ações (MNIH et al., 2015).

3.3.1 Considerações sobre a aprendizagem do valor Q

O RL envolve a modelagem de agentes que tomam decisões em um ambiente visando maximizar recompensas cumulativas. Dois algoritmos populares de RL que aprendem a função Q usando aprendizado por diferença temporal (TD) são o Deep Q-Network (DQN) e o SARSA [*State-Action-Reward-State-Action* (SARSA)]. Ambos utilizam a equação de Bellman para atualizar as estimativas do valor Q, mas diferem na forma como constroem a função $Q^{\pi}_{tar}(s, a)$, o valor Q alvo (GRAESSER; KENG, 2020).

3.3.2 Equações de Bellman para DQN e SARSA

As equações de Bellman para DQN e SARSA são mostradas a seguir:

$$Q^{\pi}_{DQN}(s, a) \approx r + \gamma \max_{a'} Q^{\pi}(s', a') \tag{8}$$

$$Q^{\pi}_{SARSA}(s, a) \approx r + \gamma Q^{\pi}(s', a') \tag{9}$$

A diferença fundamental está no cálculo do valor Q para o próximo estado s'. No DQN, o valor Q é atualizado considerando o valor máximo entre todas as ações possíveis no próximo estado, enquanto no SARSA, o valor Q é atualizado com base na ação específica realmente tomada pela política atual (GRAESSER; KENG, 2020).

3.3.3 Construção do valor alvo (*tar*) calculado Q^{π}_{tar}

As fórmulas para construir $Q^{\pi}_{tar}(s, a)$ são:

$$Q^{\pi}_{tar:DQN}(s, a) = r + \gamma \max_{a'} Q^{\pi}(s', a') \tag{10}$$

$$Q^{\pi}_{tar:SARSA}(s, a) = r + \gamma Q^{\pi}(s', a') \tag{11}$$

No DQN, o valor de Q^{π}_{tar} independe da política utilizada para coletar as experiências, tornando-o um algoritmo *off-policy*. Isso ocorre porque $Q^{\pi}_{tar:DQN}(s, a)$ é calculado utilizando o máximo valor Q possível no próximo estado, sem considerar a política que escolheu a ação a'. Portanto, as experiências geradas sob diferentes políticas ainda podem ser utilizadas para atualizar o valor Q de forma consistente (GRAESSER; KENG, 2020).

Por outro lado, o SARSA é um algoritmo *on-policy*, pois a construção de $Q^{\pi}_{tar:SARSA}(s,a)$ depende da ação a' realmente tomada pela política atual. Isso significa que as atualizações do valor Q são diretamente influenciadas pela política que gera as experiências (GRAESSER; KENG, 2020).

3.3.4 Implicações para o treinamento

Considerando duas políticas π_1 e π_2 e suas respectivas funções Q $\hat{Q}^{\pi}_1(s,a)$ e $\hat{Q}^{\pi}_2(s,a)$, se π_1 é a política atual e π_2 é uma política mais antiga, os dados gerados por estas políticas [por exemplo, (s,a,r,s',a'_1) e (s,a,r,s',a'_2)] podem resultar em diferentes atualizações para o valor Q no SARSA. Isso ocorre porque a ação a' é específica da política utilizada para coletar a experiência. No entanto, no DQN, como a atualização usa o valor máximo entre todas as ações possíveis no próximo estado, a diferença entre as políticas não afeta a construção de Q^{π}_{tar} (GRAESSER; KENG, 2020).

3.3.5 Robustez do DQN em ambientes estocásticos

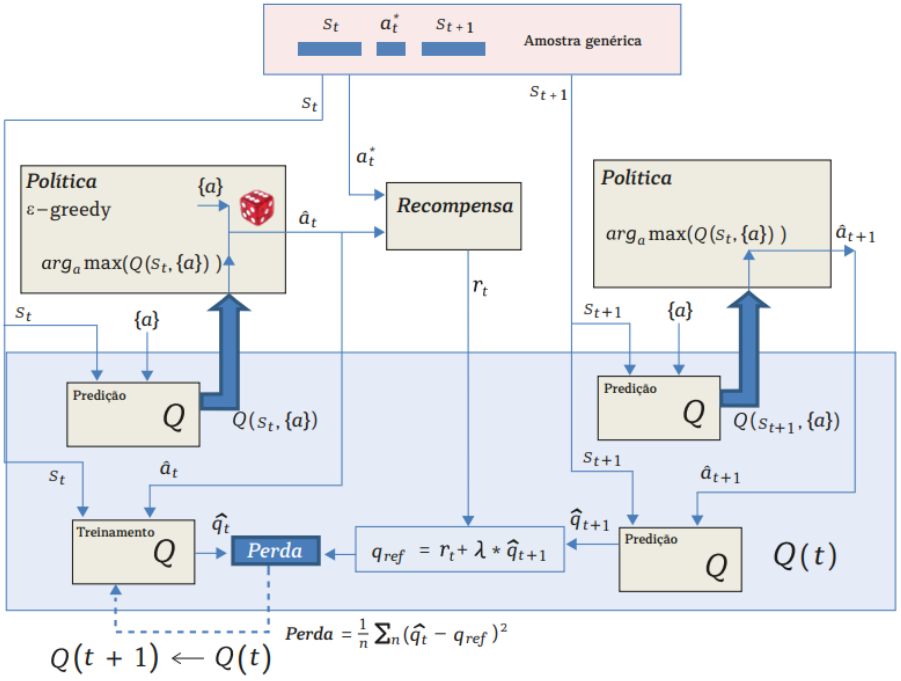
Mesmo em ambientes onde as funções de transição e recompensa são estocásticas, o DQN mantém sua validade. Isso porque a função Q é definida como o retorno futuro esperado, considerando todas as incertezas nas transições e recompensas (GRAESSER; KENG, 2020). A construção de $Q^{\pi}_{tar:DQN}(s,a)$ continua a ser independente da política usada para coleta de experiências, desde que as transições e recompensas sejam determinadas pelo ambiente, o fluxograma de funcionamento de um DQN pode ser descrito através da Figura 8.

3.3.6 Otimização rede DQN

O Deep Q-Network (DQN) enfrenta alguns problemas, como *overfitting* no espaço de tempo e instabilidade durante o treinamento. O overfitting temporal ocorre quando as atualizações dos valores Q são altamente correlacionadas. Isso acontece porque, ao treinar a rede Q, as atualizações consecutivas são realizadas com base em transições sequenciais (s_t, a_t, r_t, s_{t+1}), $(s_{t+1}, a_{t+1}, r_{t+1}, s_{t+2})$, e assim por diante (GRAESSER; KENG, 2020). Como essas transições estão temporalmente próximas, elas compartilham muitas características comuns, o que pode levar a uma convergência

subótima, onde a rede Q ajusta-se excessivamente a padrões específicos dessas sequências (GRAESSER; KENG, 2020).

Figura 8 – Rede DQN sem otimizações.



Fonte: Adaptado de (LOPEZ-MARTIN; CARRO; SANCHEZ-ESGUEVILLAS, 2020).

Além disso, o DQN pode sofrer com a superestimação dos valores Q, onde o valor máximo escolhido entre as ações possíveis é, em média, maior do que o valor real, resultando em políticas menos eficientes. Esse problema surge porque o algoritmo tende a selecionar ações que parecem ter alto valor esperado, mas que, na realidade, são superestimadas devido à variabilidade na estimativa dos valores Q(GRAESSER; KENG, 2020).

Para mitigar esses problemas, várias otimizações foram propostas, incluindo a rede alvo, Double DQN e a memória de *replay*. Cada uma dessas técnicas contribui para estabilizar e melhorar o desempenho do DQN de maneiras distintas (GRAESSER; KENG, 2020).

3.3.7 Rede alvo

A rede alvo é uma técnica introduzida para reduzir a instabilidade no treinamento do DQN. Em vez de usar a mesma rede neural para selecionar e avaliar ações, uma cópia da rede Q, chamada de rede alvo, é utilizada para calcular os valores alvo (GRAESSER; KENG, 2020). Esta rede alvo é atualizada periodicamente com os pesos

da rede principal, em vez de ser atualizada em todas as iterações. Isso ajuda a suavizar as atualizações e reduz as oscilações no aprendizado, pois desacopla os valores alvo das rápidas mudanças na rede Q principal, proporcionando uma referência mais estável (GRAESSER; KENG, 2020).

Matematicamente, a rede alvo modifica a construção do valor Q^{π}_{tar} para:

$$Q^{\pi}_{tar}(s,a) = r + \gamma \max_{a'} Q^{\pi}_{target}(s',a'), \tag{12}$$

onde Q^{π}_{target} é a função Q da rede alvo.

3.3.8 Double DQN

A superestimação dos valores Q é outro problema significativo no DQN (GRAESSER; KENG, 2020). O *Double DQN* (DDQN) foi introduzido para abordar esse problema, separando a seleção e a avaliação das ações(GRAESSER; KENG, 2020). No DDQN, a rede principal seleciona a ação que maximiza o valor Q, mas a rede alvo avalia o valor do Q correspondente. Isso reduz o viés de superestimação e resulta em valores Q mais precisos. Ao utilizar duas redes para separar esses processos, a avaliação se torna mais realista e menos tendenciosa (GRAESSER; KENG, 2020).

A modificação no valor alvo do DDQN é dada por:

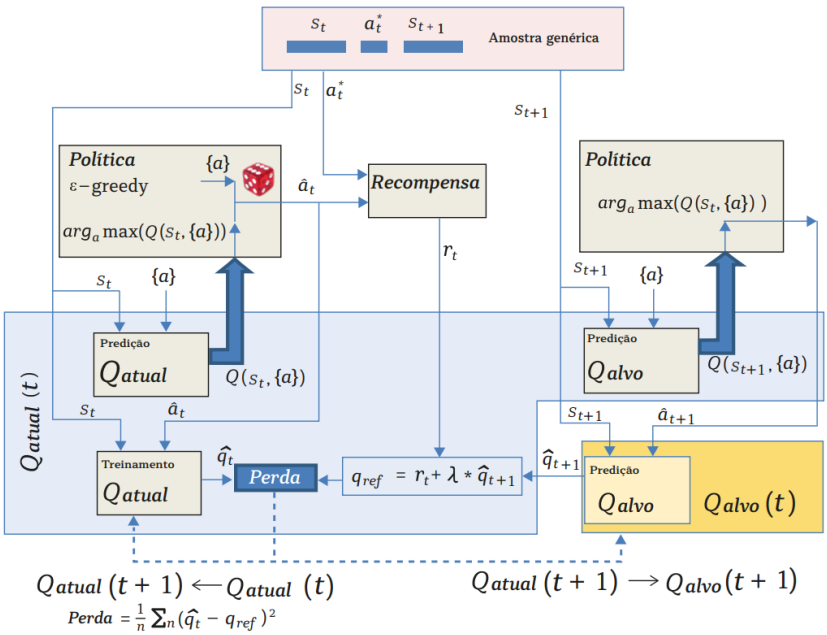
$$Q^{\pi}_{tar}(s,a) = r + \gamma Q^{\pi}_{target}(s', \arg\max_{a'} Q^{\pi}(s',a')) \tag{13}$$

Essa abordagem garante que a escolha da ação seja baseada na rede principal, enquanto a avaliação do valor dessa ação seja mais precisa, feita pela rede alvo, após sua adição a rede pode ser vista na Figura 9, abaixo.

3.3.9 Memória de replay

A memória de *replay* é usada para desacoplar a correlação entre as atualizações, armazenando transições observadas (s,a,r,s') em um *buffer* de *replay*. Durante o treinamento, amostras aleatórias são extraídas deste *buffer* para atualizar a rede Q. Isso não só ajuda a mitigar o *overfitting* temporal, mas também melhora a eficiência de uso das experiências coletadas, permitindo um aprendizado mais robusto e estável. Ao amostrar aleatoriamente, as correlações temporais são quebradas, o que resulta em uma atualização mais independente e uma melhor generalização do modelo (GRAESSER; KENG, 2020).

Figura 9 – Rede otimizada: DDQN.



Fonte: Adaptado de (LOPEZ-MARTIN; CARRO; SANCHEZ-ESGUEVILLAS, 2020).

A memória de *replay* de experiências armazena as k experiências mais recentes que um agente coletou. Quando a memória está cheia, a experiência mais antiga é descartada para abrir espaço para a nova. Cada vez que o agente treina, um ou mais lotes de dados são amostrados de forma aleatória e uniforme da memória de *replay* de experiências. Cada um desses lotes é usado sequencialmente para atualizar os parâmetros da rede de função Q. O valor de k é tipicamente bastante grande, variando entre 10.000 e 1.000.000, enquanto o número de elementos em um lote é bem menor, geralmente entre 32 e 2.048 (GRAESSER; KENG, 2020).

O processo de atualização com memória de *replay* pode ser descrito como:

1. Armazene cada transição (s, a, r, s') na memória de *replay*.
2. Amostre aleatoriamente um *minibatch* de transições da memória de *replay*.
3. Calcule os valores Q alvo para cada transição no *minibatch*.
4. Atualize a rede Q usando os valores Q alvo calculados.

Dessa forma, a memória de *replay* permite que o agente aprenda de um conjunto mais diversificado de experiências, reduzindo a correlação entre as atualizações e promovendo um treinamento mais eficaz.

Estas três técnicas — rede alvo, DDQN e memória de *replay* — são essenciais para otimizar o desempenho do DQN, tornando o aprendizado mais estável e eficaz em diversos ambientes de RL (GRAESSER; KENG, 2020).

4 Metodologia

Neste capítulo, a metodologia empregada neste trabalho é apresentada e detalhada, com uma abordagem sistemática que abrange cada etapa dos processos relacionados à detecção de anomalias utilizando DRL. Primeiramente, é apresentada uma visão abrangente do conjunto de dados utilizado, o *Numenta Anomaly Benchmark* (NAB), incluindo uma explicação sobre seu funcionamento e a metodologia adotada para avaliação de desempenho, com destaque para a métrica F1-score. Em seguida, o processo de pré-processamento dos dados é descrito em detalhes, abordando etapas como normalização com o `MinMaxScaler`, criação de janelas temporais, cálculo do desvio padrão condicional e a computação da derivada do sinal original. Tais procedimentos são essenciais para garantir que os sinais analisados estejam em um formato discriminativo e adequado para o treinamento do modelo. A definição do modelo é explorada na sequência, com uma descrição completa da arquitetura do DDQN, incluindo a configuração da rede neural, funções de ativação e outros detalhes técnicos relevantes. A definição do espaço de estados e ações também é discutida, explicando como esses elementos são estruturados no contexto do problema abordado. A função de recompensa recebe uma atenção específica, com a explicação de como ela é calculada com base nas decisões de detecção de anomalias. A estrutura do código é descrita, destacando a implementação da atualização do modelo alvo e o processo de treinamento. Por fim, o treinamento do modelo é detalhado, incluindo a configuração dos hiperparâmetros, como taxa de aprendizado, fator de desconto, tamanho do lote, entre outros. Também são discutidos o número de iterações e as estratégias adotadas para evitar *overfitting*, assegurando, assim, a robustez e a eficácia do modelo desenvolvido.

4.1 Recursos computacionais utilizados

Para a implementação dos sistemas desenvolvidos neste trabalho de conclusão de curso, foi configurado um ambiente virtual Python, permitindo a execução eficiente das técnicas de DRL consideradas e os seus treinamentos em uma GPU NVIDIA GeForce RTX 3060, integrada a uma máquina com 64 GB de RAM. A linguagem de programação utilizada foi o Python 3, juntamente com a biblioteca de aprendizado

profundo TensorFlow, por meio da API Keras. Adicionalmente, bibliotecas como NumPy, Pandas, Plotly e Scikit-learn foram empregadas para suporte ao desenvolvimento e análise dos resultados obtidos. O processo de desenvolvimento foi conduzido na IDE Visual Studio, que oferece ferramentas robustas para análise de dados e gerenciamento de ambientes Python, facilitando tanto a implementação quanto a validação do modelo.

4.2 Conjunto de dados utilizados

O conjunto utilizado foi o *Numenta Anomaly Benchmark* (NAB), o qual é amplamente reconhecido como uma ferramenta robusta para a análise de desempenho de implementações de detecção de anomalias. O NAB facilita a aquisição de dados, disponibilizando um repositório público online que elimina a necessidade de coleta e pré-processamentos sofisticados, permitindo que os pesquisadores se concentrem no desenvolvimento e avaliação de suas implementações, otimizando tempo e esforço. Além disso, oferece uma diversidade de conjuntos de treinamento e validação já separados, trazendo reconhecimento e confiabilidade na comunidade de pesquisa em detecção de anomalias, com possibilidade de ampla utilização para validação dos resultados e comparação com outros trabalhos da área (LAVIN; ALEXANDER; SUBUTAI, 2015). Por fim, o uso do NAB contribui para a possibilidade de construção de uma base sólida para pesquisas futuras, permitindo a comparação de resultado com demais grupos de pesquisa. Todos os aspectos supracitados aliados à escassez de outros conjuntos de dados com a mesma robustez e fundamentação que o NAB justificam sua escolha para a avaliação de desempenho dos sistemas de detecção de anomalias desenvolvidos neste trabalho de pesquisa.

O conjunto de dados NAB foi introduzido por (LAVIN; ALEXANDER; SUBUTAI, 2015), juntamente com uma metodologia de avaliação. Os dados disponíveis são organizados em séries temporais, compreendendo originalmente 58 diferentes sinais de diversas aplicações, cada um contendo entre 1000 e 22000 amostras, somando aproximadamente 365.550 amostras em todos os subconjuntos de dados. Os autores agruparam esses dados nos seguintes subconjuntos:

- **Artificial com anomalias**: Seis séries temporais geradas artificialmente, contendo anomalias.
- **Artificial sem anomalias**: Cinco séries temporais geradas artificialmente, sem

anomalias.

- **Real AWS CloudWatch**: Dezessete séries coletadas pelo serviço Amazon Cloud Watch, com medições de uso de CPU, entrada de bytes e leitura de bytes.
- **Real causa conhecida**: Sete séries temporais de processos reais com anomalias de causa conhecida.
- **Real tráfego**: Sete séries temporais com dados de tráfego reais coletados pelo departamento de transporte de Minnesota, incluindo métricas como ocupação, velocidade e tempo de viagem de veículos de transporte público.
- **Real Tweets**: Dez séries temporais com contagens de menções no Twitter sobre dez diferentes empresas.
- **Real AdExchange**: Cinco séries com métricas de custo de serviços de publicidade online, como CPC (custo por clique) e CPM (custo por mil impressões).

Em (LAVIN; ALEXANDER; SUBUTAI, 2015) a metodologia para a obtenção dos rótulos dos dados anômalos são descritas. Especificamente, inspeções visuais e marcações de regiões visualmente anômalas são realizadas manualmente. Alguns aspectos desse procedimento são destacados, devido ao impacto direto na avaliação considerada aqui neste trabalho:

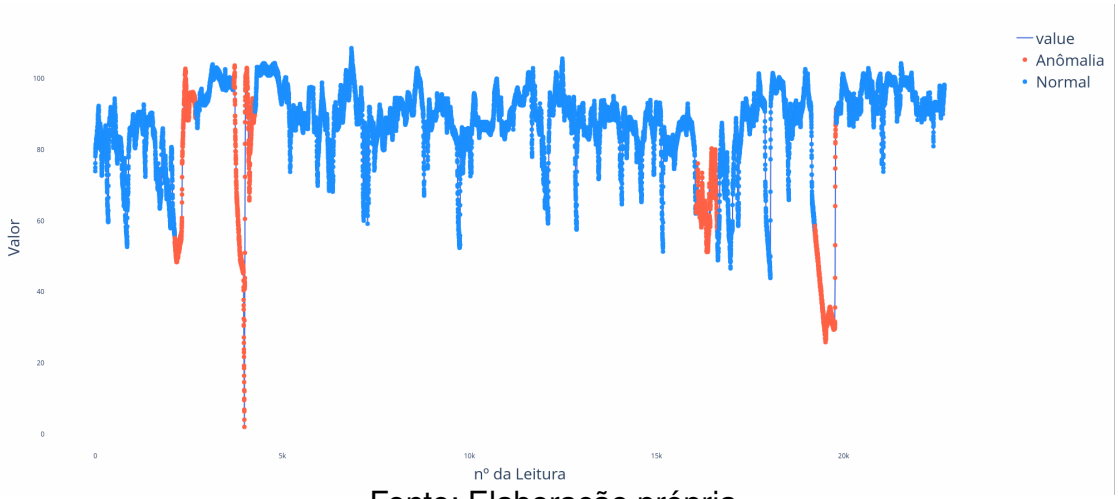
- Nos primeiros 15% do sinal não são marcadas quaisquer anomalias, por decisão do NAB. Essas amostras são utilizadas para caracterizar a classe de comportamento normal N_1.
- Se um padrão de comportamento anômalo se mantém por mais de 10% da duração do sinal, ele deixa de ser computado como anômalo após esse período.
- A partir de anomalias pontuais identificadas, são criadas janelas de anomalias ao redor do ponto identificado como anômalo. Então, os comprimentos das janelas de anomalias são computados pela Equação (14):

$$\text{Tamanho da Janela} = 0.1 \times \frac{\text{Tamanho do Sinal}}{\text{Número de Anomalias}} \tag{14}$$

A avaliação do desempenho amostra-a-amostra pode ser prejudicada pela criação de janelas (de anomalias) de tamanho fixo ao redor das amostras identificadas como anômalas. Tal prejuízo pode ocorrer porque os exemplos marcados como anômalos dentro de uma janela (de anomalia) podem não conter anomalias verdadeiras (BOZZETTO, 2023). A fim de exemplificar esse aspecto, a Figura 10 mostra a sequência

temporal de um dos subconjuntos de dados disponíveis no NAB, correspondendo ao monitoramento de temperatura de um sistema de máquinas. Esse subconjunto de dados apresenta quatro janelas de anomalias. Ainda na Figura 10, os pontos vermelhos indicam amostras nas janelas anômalas e os pontos azuis indicam as amostras classificadas como normais. Conforme mostrado na Figura 10 é possível perceber amostras normais (que seguem uma determinada tendência/padrão) marcados como anômalas.

Figura 10 – Janelas anômalas NAB.



Fonte: Elaboração própria.

4.2.1 Considerações sobre a utilização dos subconjuntos de dados para as etapas de treinamento e teste

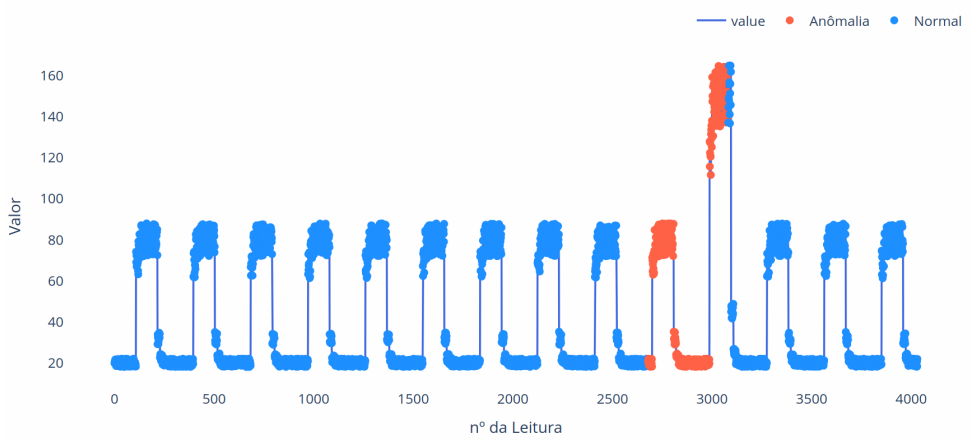
Os dados utilizados foram aqueles disponíveis no conjunto de dados (NAB). A Tabela 2 mostra todos os subconjuntos de dados que foram utilizados durante as etapas de treinamento e teste. Em geral, os subconjunto de dados são divididos entre dados artificiais e dados reais. Para os dados artificiais, identificados pelo prefixo ‘art’, o sinal *art_daily_jumpsup* foi utilizado para treinamento, enquanto os demais sinais artificiais foram reservados para teste. No caso dos dados reais, ou seja, aqueles que não foram gerados artificialmente e possuem maior complexidade de distribuição dos dados, a divisão entre treino e teste foi manualmente realizada em cada sinal. Nesses casos, os primeiros 50% de cada série são usados para treinamento e os 50% finais são utilizados para teste.

Tabela 2 – Dados utilizados

Nome do Sinal	N. Pontos	N. anomalias	N. Janelas anômalas
art_daily_flatmiddle	4032	402	1
art_daily_jumpsdown	4032	402	1
art_daily_jumpsup	4032	402	1
art_daily_nojump	4032	402	1
Twitter_volume_AMZN	15831	1580	4
RealAdExchange_4_cpc_results	1643	164	3
RealAWSCloudWatch_c06d44	15831	1580	3
RealTraffic_speed_7578	1127	116	4

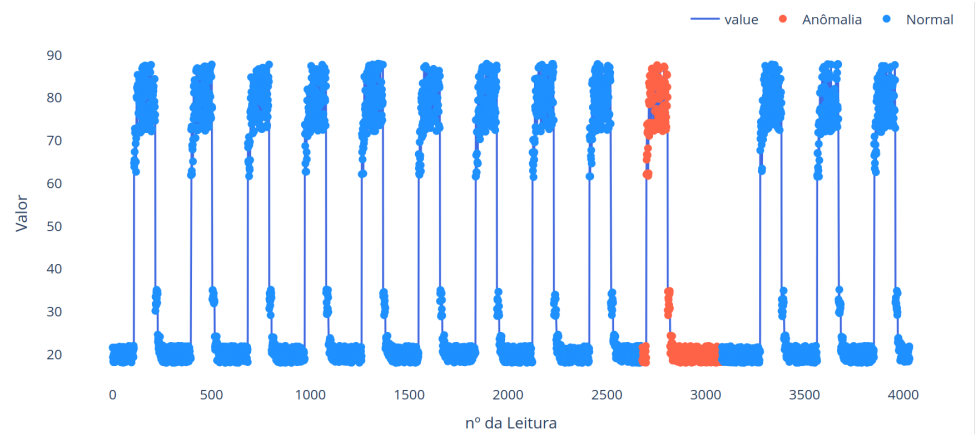
No contexto dos dados artificiais, na Figura 11 é possível visualizar o padrão “Artificial com anomalia”, como também sua janela anômala, descrita pelos pontos em vermelho. Os dados apresentados na Figura 11 são utilizado para treinamento. Já na Figura 12, o conjunto de dados *art_daily_nojump* é apresentado e utilizado para a etapa de teste dos sistemas implementados.

Figura 11 – Sinal utilizado para treinamento - dados artificiais.



Fonte: Elaboração própria.

Figura 12 – Sinal utilizado para teste - dados artificiais.



Fonte: Elaboração própria.

4.2.2 Considerações sobre o processo de avaliação do modelo

O processo de avaliação é descrito a seguir, nele é utilizada a técnica denominada F1-*score* (F1). A estratégia de avaliação proposta em (HUNDMAN et al., 2018) é também considerada aqui neste trabalho. Tal estratégia vem sendo utilizada em diversos trabalhos e *benchmarks* bem conceituados da literatura da área, como por exemplo em (GEIGER; ALNEGHEIMISH; CUESTA-INFANTE, 2020). A estratégia de avaliação consiste na detecção de anomalias em quadros (janelas) do sinal analisado. Dessa forma, a partir de rótulos anotados manualmente por especialistas, a classificação desses quadros como normais ou anômalos é comparada com a classificação anotada. Assim, a classe estimada ($\hat{N}_1$ ou $\hat{A}_2$) pelo sistema (modelo) é caracterizada como:

- Um Verdadeiro Positivo (VP), quando uma janela detectada (classificada) como anomalia pelo modelo se sobrepõe a uma janela rotulada (previamente anotada por um especialista);
- Um Falso Positivo (FP), quando uma janela detectada (classificada) como anomalia pelo modelo não se sobrepõe a nenhuma janela rotulada (previamente anotada por um especialista);
- Um Falso Negativo (FN), quando janelas rotuladas (previamente anotadas por um especialista) não são detectadas por nenhuma janela detectada (classificada).

A Figura 13 ilustra as considerações supracitadas.

Com base nos valores obtidos de verdadeiros positivos, falsos positivos e falsos negativos, através da detecção de anomalia, a métrica F1-*score* é considerada e computada pelas Equações (17), (15) e (16) (Obs.: da mesma forma que vem sendo considerada pelos processos tradicionais disponíveis na literatura (GEIGER; ALNEGHEIMISH; CUESTA-INFANTE, 2020)):

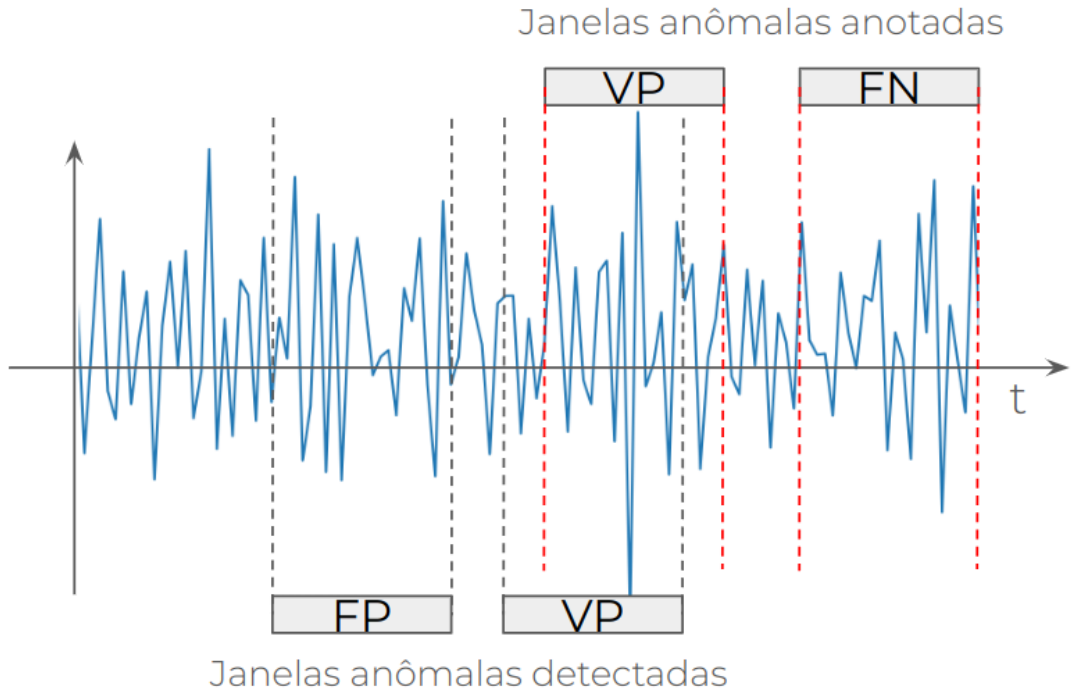
$$\text{Precision} = \frac{\text{Verdadeiros Positivos}}{\text{Total Detectado como Positivos}} = \frac{VP}{VP + FP} \tag{15}$$

$$\text{Recall} = \frac{\text{Verdadeiros Positivos}}{\text{Total Positivos}} = \frac{VP}{VP + FN} \tag{16}$$

Assim, o F1 é então calculado como a média harmônica entre Precision (15) e Recall (16), conforme a Equação (17),

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

Figura 13 – Ilustração das características de um VP, FP e FN.



Fonte: Elaboração própria.

É importante salientar que neste trabalho, assim como em (SINGH; OLINSKY, 2017), o processo de avaliação considerado equivale a uma adaptação do F1-*score* considerado no *benchmark* NAB. De acordo com (SINGH; OLINSKY, 2017), tais adaptações do processo de avaliação vêm sendo realizadas na literatura. Aqui, a adaptação se deu pela necessidade de diminuição de complexidade na avaliação dos processos. Em (SINGH; OLINSKY, 2017), sugere-se que o processo de avaliação descrito na documentação do NAB não é clara e deixa lacunas para um pleno entendimento do processo (mais justificativas e alternativas do referido processo podem ser encontradas nas referências citadas nesta seção). Vale ressaltar que mesmo com essas dificuldades, conforme já discutido nesta seção, o conjunto de dados NAB vem sendo o principal conjunto de dados para avaliações de sistemas de detecção de anomalias disponível na literatura da área.

4.3 Pré-processamento dos dados

O pré-processamento dos dados é uma etapa fundamental para assegurar que as informações estejam em um formato adequado e discriminativo para o treinamento

do modelo DDQN. Além disso, busca-se aplicar técnicas que extraiam informações relevantes do sinal, considerando o custo computacional envolvido nas transformações, dado que o DDQN, por si só, é uma abordagem computacionalmente custosa.

Nesta seção, são descritas as principais etapas de preparação dos dados, incluindo a normalização, a criação de janelas temporais e o cálculo de atributos, como o desvio padrão condicional e a derivada (do sinal original de entrada) por meio do gradiente. Esses procedimentos são essenciais para otimizar o desempenho do modelo e garantir que ele capture adequadamente os padrões presentes nos dados.

4.3.1 Criação de janelas de tempo

Para capturar a dinâmica temporal dos dados, é necessário criar janelas de tempo. Neste trabalho, a quantidade de amostras correspondente a 10% do sinal total (de cada série temporal descrita na Seção 4.2.1) foi considerada para determinar o comprimento das janelas de tempo. Dessa forma, tais janelas contêm o ponto atual, o qual é predito como anomalia ou não, e os pontos imediatamente anteriores, garantindo uma visão suficiente e representativa dos dados ao longo do tempo. Esse método envolve segmentar a série temporal em subsequências de tamanho fixo, que são então utilizadas como entradas para o modelo de aprendizado. A criação de janelas de tempo é essencial para capturar as dependências temporais e os padrões recorrentes nos dados.

4.3.2 Desvio padrão condicional

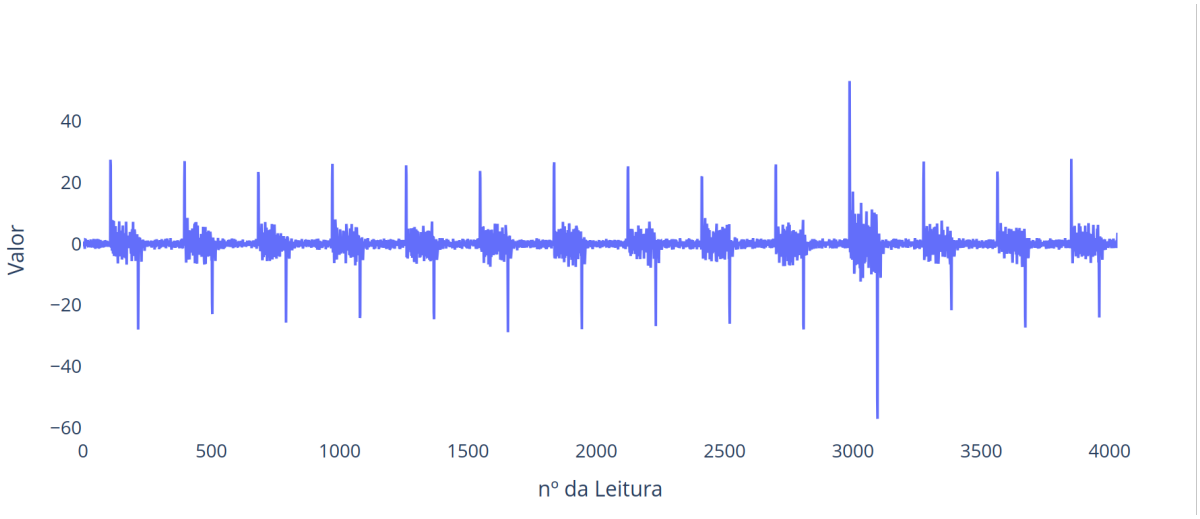
Conforme discutido na Seção 2.1.2, o desvio padrão condicional é uma medida importante para entender a variabilidade dos dados ao longo do tempo. Utilizamos a biblioteca `ARCH` para calcular o desvio padrão condicional. O modelo utilizado pela bilbioteca é o GARCH (*Generalized Autoregressive Conditional Heteroskedasticity*) é adequado para capturar a heterocedasticidade presente em séries temporais financeiras e outras aplicações onde as propriedades estatísticas dos dados variam ao longo do tempo.

4.3.3 Considerações sobre o uso da derivada

Para capturar as nuances de mudanças abruptas no sinal e identificar variações significativas na série temporal, aplicamos o cálculo da derivada utilizando a função

`gradient` da biblioteca `numpy`. A Figura 14 apresenta o sinal da derivada do subconjunto de dados utilizados para treinamento mostrado na Figura 11. A diferenciação de uma função reflete a taxa de variação do valor da função em relação a uma de suas variáveis. No contexto do pré-processamento de dados, a derivada desempenha um papel essencial ao destacar alterações no comportamento do sinal, fornecendo uma representação mais sensível às mudanças dinâmicas do sinal analisado. Nesse contexto, o uso da derivada possibilita a identificação de ruídos e variações abruptas, que poderiam confundir o modelo durante o treinamento, garantindo uma representação mais informativa e discriminativa do sinal.

Figura 14 – Derivada do subconjunto de dados art daily jumpsup.



Fonte: Elaboração própria.

4.3.4 Normalização

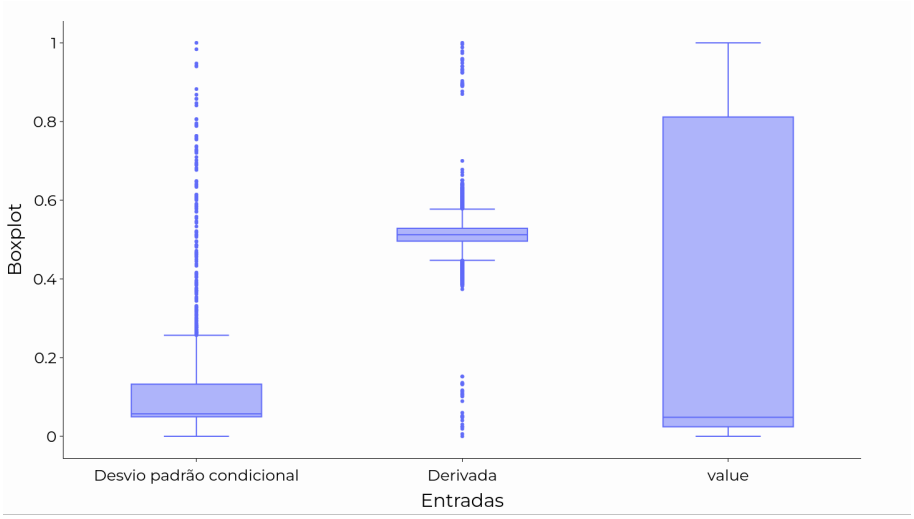
A normalização dos dados é fundamental para assegurar que todas as características dos dados estejam na mesma escala, o que facilita o processo de aprendizado do modelo. Para isso, utilizamos a função de normalização `MinMaxScaler`, disponível na biblioteca `sklearn`. Essa técnica transforma os valores dos dados para um intervalo entre 0 e 1, conforme a Equação (18):

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{18}$$

Essa transformação ajuda a prevenir que características com maior escala dominem o processo de aprendizado, permitindo que o modelo trate todas as entradas de forma equilibrada. Na Figura 15 é possível observar a dispersão dos dados através de um

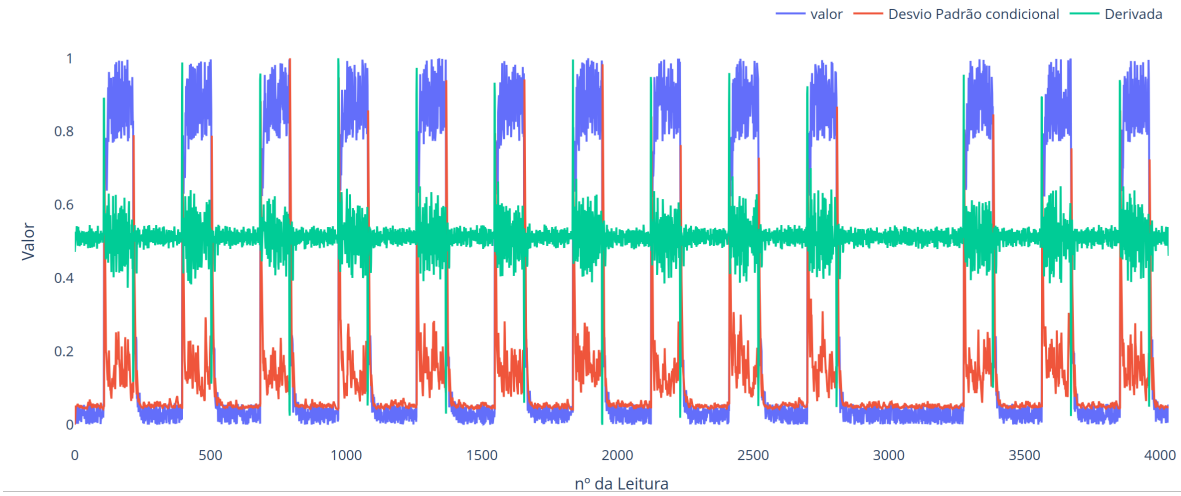
diagrama de caixa (*boxplot*). Já na Figura 16 são mostrados os sinais que compõem um vetor de estado $\mathbf{S}_t$, a saber: desvio padrão condicional σ_t, derivada $\partial x_t/\partial t$ e as amostras da série pós normalização $\bar{x}_t$.

Figura 15 – Diagrama de caixa correspondente ao sinal de entrada.



Fonte: Elaboração própria.

Figura 16 – Sinais temporais normalizados.



Fonte: Elaboração própria.

Com essas etapas de pre-processamento dos dados, garantimos que os dados de entrada estejam em um formato adequado e otimizado para o treinamento do modelo de *Deep Reinforcement Learning*, melhorando a capacidade do modelo de detectar padrões e anomalias de forma eficaz.

4.4 Características da rede

4.4.1 Espaço de estados e ações

No contexto da detecção de anomalias utilizando DRL, a definição do espaço de estados e ações é fundamental para a eficácia do modelo. O espaço de estados é composto por características extraídas das séries temporais que capturam a informação relevante necessária para a tomada de decisões pelo agente. Especificamente, os estados são definidos utilizando uma janela de tempo que abrange múltiplos pontos de dados consecutivos $[\mathbf{o}_{t-2}, \mathbf{o}_{t-1}, \mathbf{o}_t]$. Cada estado é representado por um vetor $\mathbf{S}_t$ que inclui os seguintes sinais normalizados: desvio padrão condicional σ_t, derivada do sinal $\partial \mathbf{o}_t / \partial t$ e o valor original do sinal $\mathbf{o}_t$. Assim, o vetor de estado $\mathbf{S}_t = [\sigma_t, \partial \mathbf{o}_t / \partial t, \mathbf{o}_t]$ é definido (e redimensionado) para formar uma entrada (de dimensão adequada) para o modelo DRL de aprendizado.

As ações disponíveis para o agente são binárias e consistem em detectar ou não uma anomalia no ponto de dados atual, $A_t = [N_1, A_2]$. A seleção da ação é realizada pelo agente com base no estado atual S_t, utilizando uma rede neural que estima a política π para tomada de decisão. A relação de compromisso entre explorar novos estados e aproveitar o conhecimento adquirido de estados previamente processados (conhecidos) é ajustada por meio da configuração de hiperparâmetros. À medida que as épocas avançam, esses hiperparâmetros gradualmente transferem o poder de decisão para a rede neural (podendo reduzir a exploração e favorecer a exploração do aprendizado acumulado). Esse equilíbrio é crucial para o aprendizado eficiente dos agentes, permitindo a identificação de anomalias de forma eficaz e consistente.

4.4.2 Função de recompensa

A função de recompensa é projetada para orientar o agente na tomada de decisões corretas ao detectar anomalias. A recompensa é calculada com base na precisão das decisões do agente, incentivando detecções corretas e penalizando erros. Especificamente, a função de recompensa concede uma recompensa positiva quando o agente detecta corretamente uma anomalia e aplica penalidades quando ocorrem detecções incorretas.

Quando o agente detecta uma anomalia e a detecção está correta, uma recompensa positiva é concedida. Isso reforça a decisão correta e incentiva o agente a

continuar detectando anomalias com precisão. Em contrapartida, se o agente detecta uma anomalia e esta detecção está incorreta, uma penalidade é aplicada para desincentivar falsas detecções de anomalias. Da mesma forma, se o agente não detecta uma anomalia que de fato existe, uma penalidade significativa é imposta para corrigir essa falha no futuro. Por fim, uma pequena recompensa positiva é dada quando o agente corretamente não detecta uma anomalia, incentivando a precisão global do modelo. Os detalhes dos valores de recompensas imediatos são descritos, abaixo:

- Se a ação selecionada pelo agente no estado caracterizar anomalia e a série temporal indicar uma anomalia no índice atual, a recompensa é definida como 10.
- Se a ação selecionada pelo agente no estado caracterizar normalidade e a série temporal indicar uma anomalia no índice atual, a recompensa é definida como -10.
- Se a ação selecionada pelo agente no estado caracterizar anomalia e a série temporal indicar uma normalidade no índice atual, a recompensa é definida como -2.
- Se a ação selecionada pelo agente no estado caracterizar normalidade e a série temporal indicar normalidade no índice atual, a recompensa é definida como χ, esse sinal de recompensa será parâmetro de ajuste da rede, podendo variar a depender do sinal aplicado.

As recompensas e penalidades imediatas obtidas pelo agente são processadas pela equação de Bellman, descrita nas Equações (8) e (9). No entanto, para a implementação do DDQN, utiliza-se uma versão aprimorada, apresentada na Equação (13). Essas recompensas, juntamente com os estados e ações correspondentes, são armazenadas em uma memória de repetição (*replay memory*). Essa estratégia permite que o agente revisite e aprenda com experiências passadas, refinando continuamente sua capacidade de detectar anomalias ao longo do tempo.

4.4.3 Definição do modelo: Double Deep Q-Network (DDQN)

O modelo utilizado para detecção de anomalias é baseado na arquitetura DDQN. Este modelo aprimora o algoritmo DQN ao usar duas redes neurais idênticas: a rede principal e a rede alvo(GRAESSER; KENG, 2020). A rede principal é utilizada para selecionar as ações, enquanto a rede alvo é empregada para calcular os valores

Q esperados, o que ajuda a reduzir o viés nas estimativas de valor e a melhorar a estabilidade do treinamento. O algoritmo de treinamento utilizando memória de *replay* pode ser visto no quadro abaixo:

Algoritmo 1: : Algoritmo de treinamento

for (s_t, a_t, r_t, s_{t+1}) *até* $N_{minibatchs}$ **do**
  A rede principal prevê os valores dos alvos
  O próximo estado é previsto como o argumento de maior valor do próximo estado
  A rede Alvo avalia o valor da ação de maior valor no próximo estado escolhido pela rede principal
  A ação prevista pela rede principal recebe o valor referente a $r + \gamma Q^{\pi}_{target}(s', \arg\max_{a'} Q^{\pi}(s', a'))$
  A rede neural treina
end

Além disso, o modelo incorpora uma memória de *replay*, que armazena experiências passadas sob a forma de transições (estado, ação, recompensa, próximo estado). Durante o treinamento, mini-lotes são amostrados aleatoriamente dessa memória para diminuir a correlação temporal das experiências consecutivas, garantindo diferentes dinâmicas nas sequências de dados analisados e possibilitando um treinamento mais eficiente.

Para cada mini-lote, o modelo realiza múltiplas iterações de treinamento, o que permite ao agente aprender de maneira mais robusta a partir das experiências passadas. A rede é treinada utilizando a função de perda de erro quadrático médio [*Mean squared error* (MSE)] e o otimizador Adam com uma taxa de aprendizado ajustável. A arquitetura da rede neural usada no DDQN é simples, com duas camadas densas de 64 neurônios cada, usando a função de ativação ReLU, seguidas por uma camada de saída linear correspondente ao número de ações possíveis.

4.4.4 Rede neural

A rede neural utilizada no modelo de *Deep Reinforcement Learning* (DRL) é relativamente simples, refletindo o foco em outras partes da técnica. A arquitetura da rede consiste em uma sequência de camadas densas. Primeiramente, uma camada densa com 64 neurônios e função de ativação ReLU processa a entrada. Em seguida, uma segunda camada densa também com 64 neurônios e função de ativação ReLU é aplicada. Finalmente, a camada de saída possui um número de neurônios igual ao

número de ações possíveis, com ativação linear. A rede é compilada com a função de perda de erro quadrático médio (MSE) e o otimizador Adam, utilizando uma taxa de aprendizado de $0{,}001$. A simplicidade desta rede facilita a integração e treinamento rápidos, permitindo que o foco principal do estudo permaneça nas técnica da rede de aprendizado por reforço.

4.4.5 Algoritmo completo

O algoritmo utilizado para o treinamento do modelo na detecção de anomalias é apresentado a seguir. Ele oferece uma visão geral consolidada de todos os temas abordados nesta seção, sem se aprofundar nos detalhes de cada etapa, como o processo de obtenção e transformação dos dados de entrada do modelo. Os principais hiperparâmetros necessários para o treinamento estão resumidamente listados na Tabela 3.

Tabela 3 – Hiperparâmetros

Hiperparametro	Valor
Janelamento	10%
Tamanho de estado	3 x Tamanho da janela
Iterações	150 (quando não espeficicado)
Fator de desconto	0.9
Taxa de exploração inicial	1
Taxa de decaimento	0.8
Memória de *replay*	10000
Lotes por treinamento	2
Treinamentos por lote	3
Frequência de atualização rede alvo	20

Algoritmo 2: : Algoritmo *Double Deep Q-Learning*

for *época* $= 1$ *até* N_{pocas} **do**
  for *estado até* $N_{estados}$ **do**
    Estado atual $s_t \leftarrow$ index série temporal
    Próximo estado $s_{t+1} \leftarrow$ index$_{t+1}$ série temporal
    $a_t \leftarrow$ a ação tomada $\arg\max_{a_t}\{Q_t(s_t,a_t)\}$
    if *número aleatório* $\leq$ *taxa de exploração* **then**
      $a_t \leftarrow$ ação aleatória
    end
    if *Previu corretamente* **then**
      Obtém a recompensa imediata positiva
      else
        Obtém a recompensa imediata negativa
      end
    end
    if *tamanho esperado memória de replay* $\leq$ *memória de replay* **then**
      delete primeira posição
    end
    Salva tupla (s_t, a_t, r_t, s_{t+1})
  end
  Batches aleatórios
  for *batche até* $N_{Batches}$ **do**
    for *atualizações até* $N_{atualizaes}$ **do**
      Training()
      Atualiza Q: $Q^{\pi}_{tar}(s,a) = r + \gamma Q^{\pi}_{target}(s', \arg\max_{a'} Q^{\pi}(s', a'))$
    end
  end
  if *época até Atualização de Alvo* **then**
    Atualiza rede alvo
  end
  if *taxa de exploração* $>$ *taxa de exploração mínima* **then**
    Decaia taxa de exploração
  end
end

5 Análise e Discussão de Resultados

Os subconjuntos de dados considerados neste trabalho, apresentados na Seção 4.2, são usados para avaliar o desempenho do algoritmo DDQN, conforme o protocolo descrito na seção anterior. Nesse contexto, é importante destacar que os 10% iniciais das amostras de dados no sinal analisado são excluídos da análise final de desempenho. Essa exclusão é necessária porque o modelo utiliza uma janela móvel de 10% para o processamento das entradas, resultando na ausência de detecção de anomalias durante esse período inicial. Esse procedimento é fundamental para garantir a confiabilidade e a precisão dos resultados, que são discutidos e ilustrados nas figuras apresentadas nesta seção.

Para fins de comparação, é utilizado o trabalho realizado por (GEIGER; ALNEGHEIMISH; CUESTA-INFANTE, 2020), que referencia diversos estudos que empregaram o conjunto de dados NAB na avaliação de modelos de detecção de anomalias. Esses trabalhos são destacados pelo autor não apenas por representarem o estado da arte na área, mas também pela relevância das instituições responsáveis por suas pesquisas e publicações.

Os resultados apresentados na Tabela 4 correspondem aos desempenhos de sistemas de detecção de anomalias baseados em diferentes modelos de referência do estado da arte disponíveis na literatura, tais como TadGAN, LSTM, ARIMA, DeepAR, LSTM AE, HTM, Dense AE, MAD-GAN e MS Azure, todos implementados conforme descrito em (GEIGER; ALNEGHEIMISH; CUESTA-INFANTE, 2020). Ainda na Tabela 4, destaca-se o sistema de detecção de anomalias desenvolvido neste trabalho, fundamentado no modelo DDQN. Observa-se que o desempenho alcançado pela abordagem proposta é competitivo em relação aos modelos do estado da arte, evidenciando a eficácia do sistema de detecção de anomalias baseado em DDQN apresentado neste trabalho de pesquisa.

A Tabela 4 apresenta os resultados gerais, expressos pelos valores da métrica F1-*score*, obtidos nos subconjuntos de dados analisados neste trabalho (NAB-Art, NAB-AdEx, NAB-AWS, NAB-Traf e NAB-Tweets). Conforme discutido na Seção 4.2.2 e calculado pela Equação (17), o F1-*score* varia no intervalo de 0 a 1 ($F1 \in [0,1]$), onde valores próximos de 0 indicam um desempenho insatisfatório, enquanto valores

próximos de 1 refletem um desempenho ideal.

Um aspecto relevante observado no uso do modelo foi o valor atribuído às recompensas para verdadeiros negativos. Nessas situações, adotou-se um valor dinâmico, considerando as características empíricas inerentes às recompensas no contexto da aplicação. Observou-se que a variabilidade das recompensas é inversamente proporcional ao tamanho do sinal analisado. Verificou-se que, para sinais com um maior número de estados, o valor da recompensa para verdadeiros negativos deve ser reduzido. Isso ocorre porque, em sinais mais extensos, a frequência de verdadeiros negativos é mais elevada, contribuindo de maneira significativa para a maximização da soma total de recompensas, o que é coerente, dado o impacto acumulado desses acertos no desempenho global do modelo.

Tabela 4 – Comparação de desempenho entre diferentes modelos no conjunto NAB (adaptado de (GEIGER; ALNEGHEIMISH; CUESTA-INFANTE, 2020))

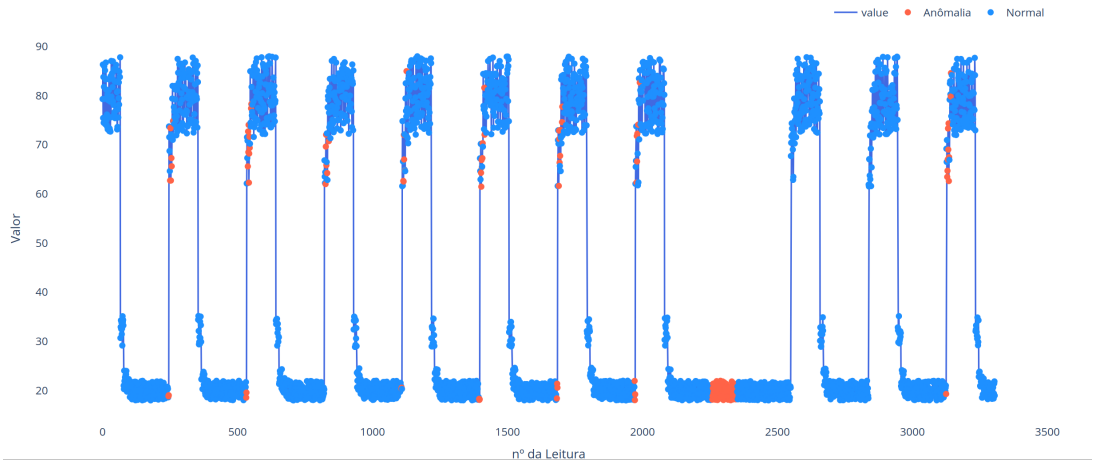
	NAB				
Modelos	**Art**	**AdEx**	**AWS**	**Traf**	**Tweets**
DDQN	**1**	**0,516**	**0,423**	**0,471**	**0,641** $\pm 0{,}005$
TadGAN	0,8	0,8	0,644	0,486	0,609
LSTM	0,375	0,538	0,474	0,634	0,543
Arima	0,353	0,583	0,518	0,571	0,567
DeepAR	0,545	0,615	0,39	0,542	0,542
LSTM AE	0,545	0,571	0,764	0,552	0,542
HTM	0,455	0,519	0,571	0,474	0,526
Dense AE	0,444	0,267	0,273	0,412	0,444
MAD-GAN	0,324	0,297	0,23	0,333	0,057
MS Azure	0,125	0,066	0,173	0,166	0,118

5.1 Subconjunto de dados Artificiais com anomalia

Para os dados artificiais presentes no conjunto NAB, destaca-se a série *art_daily_nojump*, na Tabela 4 chamado apenas de Art. O treinamento foi realizado conforme descrito nos métodos, porém utilizando diferentes números de épocas, o que permitiu observar comportamentos distintos do modelo. Com apenas 80 épocas de treinamento, conforme mostrado na Figura 17, o modelo identificou picos no sinal periódico como anomalias, refletindo uma sensibilidade exacerbada a mudanças positivas abruptas no padrão do sinal. Essa detecção, embora incorreta, demonstra que o modelo estava inclinando-se a considerar qualquer variação no padrão como potenci-

almente anômala. O F1-*Score* foi de $0{,}45$, o que reflete o impacto negativo dos falsos positivos, embora ainda haja a presença de verdadeiros positivos que contribuíram para a pontuação. A detecção de falsos positivos, apesar de reduzir a precisão, ainda mantém o modelo relevante na detecção de anomalias, pois demonstra uma tendência a capturar eventos anômalos, mesmo que de forma exagerada.

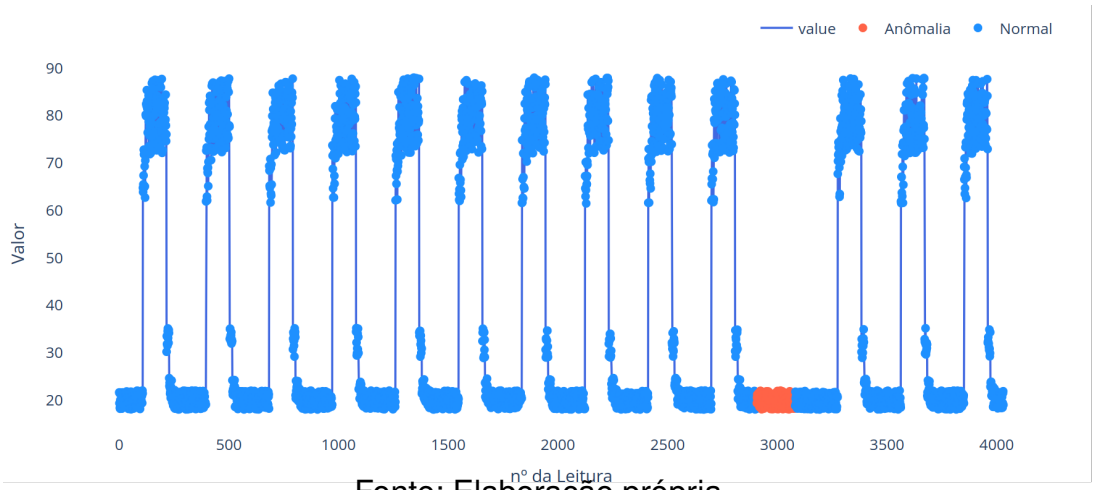
Figura 17 – Dados artificial No Jump com anomalias detectadas - 80 épocas.



Fonte: Elaboração própria.

Quando o treinamento foi estendido para 120 épocas, como ilustrado na Figura 18, o modelo aprimorou sua compreensão do padrão subjacente ao sinal e passou a detectar anomalias com maior precisão. Nesse caso, o modelo conseguiu identificar corretamente o único desvio verdadeiro presente no padrão periódico do sinal, resultando em um F1-Score de 1,0. Esse resultado indica uma detecção perfeita, pois parte da janela anômala foi corretamente identificada pelo modelo.

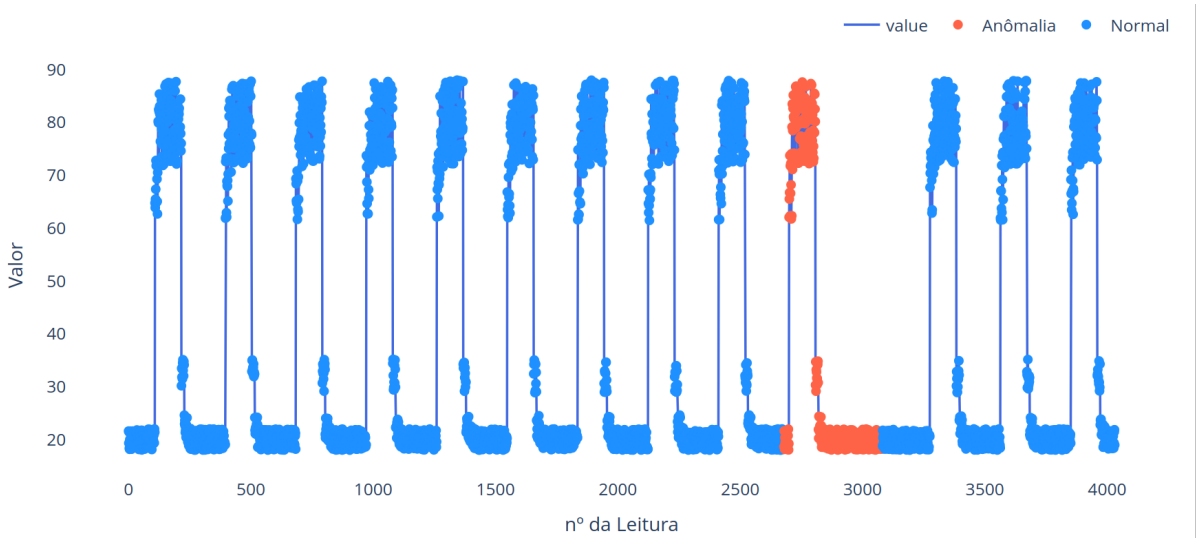
Figura 18 – Dados artificial No Jump com anomalias detectadas - 120 épocas.



Fonte: Elaboração própria.

Essa diferença significativa entre os dois treinamentos com diferentes épocas pode ser atribuída ao comportamento do NAB em definir janelas anômalas em torno do sinal analisado. O NAB considera 5% antes e 5% depois de um evento anômalo como parte da janela anômala, o que pode levar a picos e vales periódicos do sinal a serem erroneamente marcados como anômalos. Este problema já foi mencionado por outros autores conforme comentado na Seção 4.2. No entanto, com 120 épocas de treinamento, o modelo foi capaz de superar essa limitação e focar na verdadeira anomalia, enquanto com 80 épocas, ele gerou um número significativo de falsos positivos. A Figura 19 mostra o sinal rotulado a ser identificado.

Figura 19 – Dados artificial No Jump com anomalias verdadeiras.

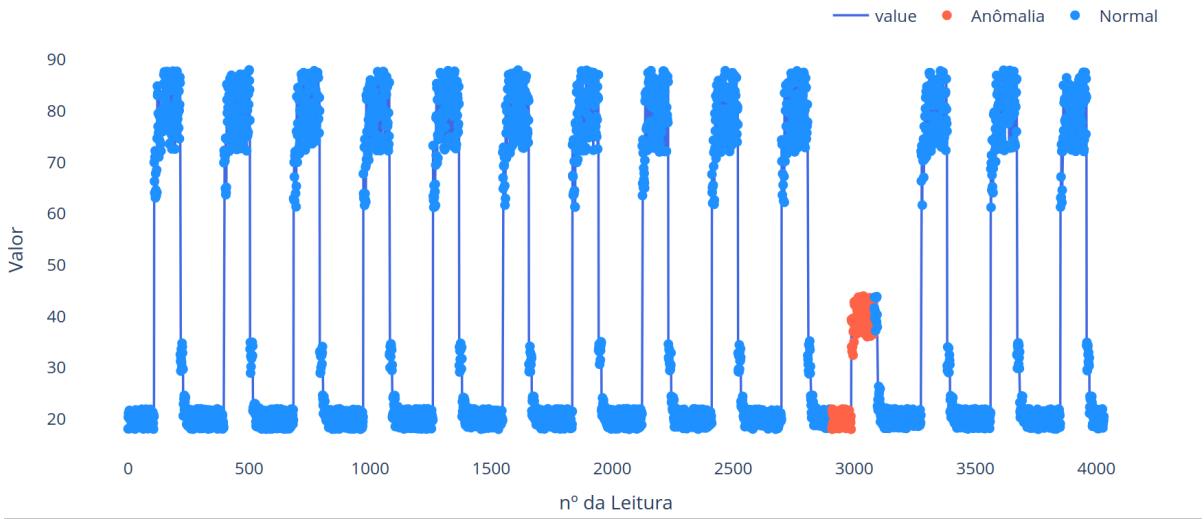


Fonte: Elaboração própria.

Essa dinâmica de análise e treinamento pode ser repetida, considerando a reprodutibilidade dos resultados, ao aplicarmos o modelo em outro sinal periódico idêntico aos já apresentados, mas com uma anomalia distinta. O sinal usado para isso, presente no conjunto NAB, é conhecido como *art_daily_jumpdown* e caracteriza-se por uma subida atenuada em magnitude. Na Figura 20, onde o modelo foi treinado com 120 épocas, obtivemos um F1-*Score* de $1{,}0$, indicando que o modelo identificou a anomalia com precisão, sem gerar falsos positivos.

Por outro lado, na Figura 21, que foi gerada com o modelo treinado por apenas 80 épocas, observamos novamente o comportamento de detectar subidas como anomalias. Esse resultado destaca a sensibilidade excessiva do modelo a mudanças no sinal quando o treinamento é insuficiente, o que, apesar de resultar em um maior

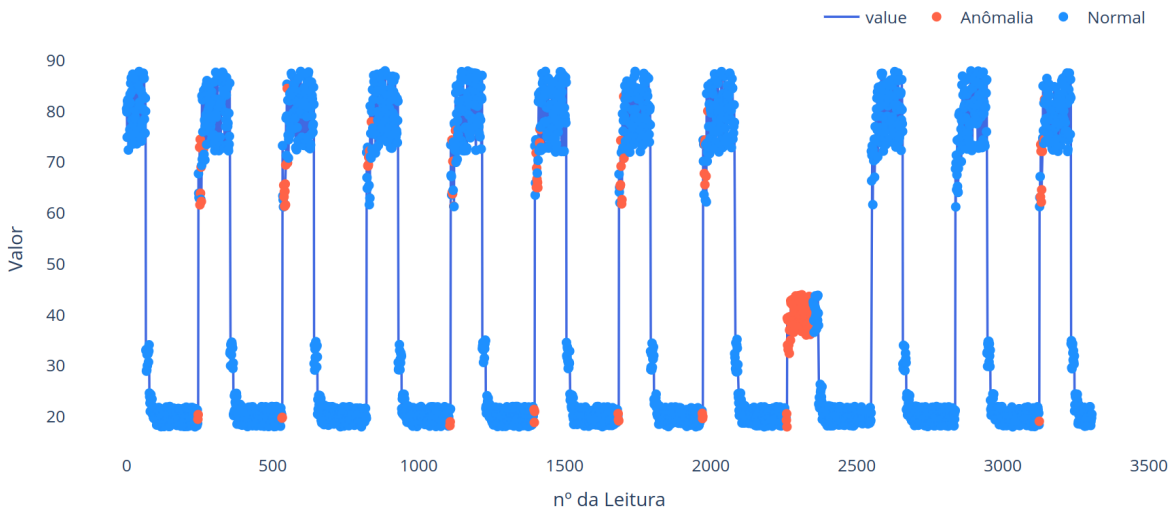
Figura 20 – Dados artificial Jump Down com anomalias detectadas - 120 épocas.



Fonte: Elaboração própria.

número de falsos positivos, ainda permite a detecção de verdadeiros eventos anômalos.

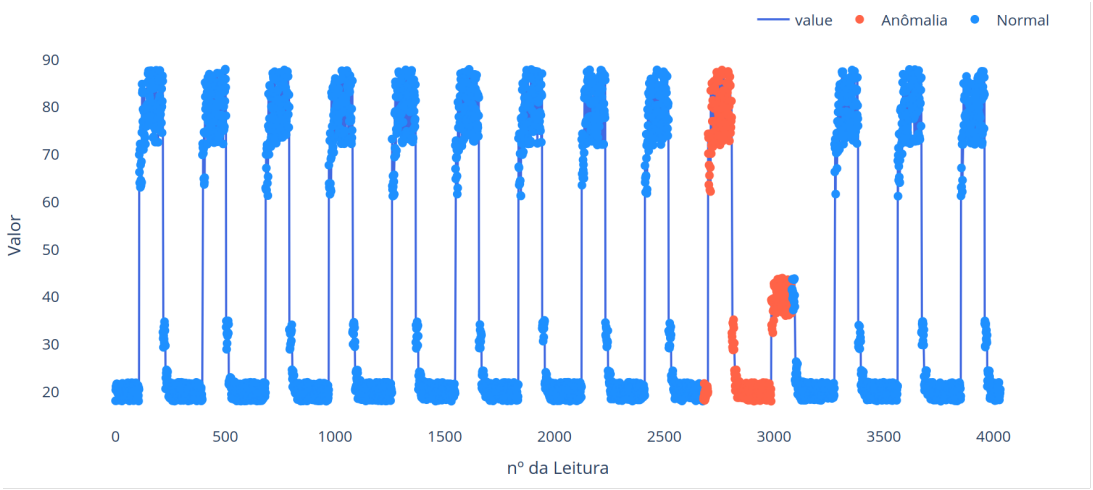
Figura 21 – Dados artificial Jump Down com anomalias detectadas - 80 épocas.



Fonte: Elaboração própria.

Essa comparação entre diferentes épocas de treinamento reforça a importância de um treinamento adequado para o equilíbrio entre sensibilidade e especificidade na detecção de anomalias. A Figura 22 mostra o sinal correto a ser identificado. A recompensa para verdadeiro negativo utilizada foi de $0{,}3$.

Figura 22 – Dados artificial Jump Down com anomalias verdadeiras.



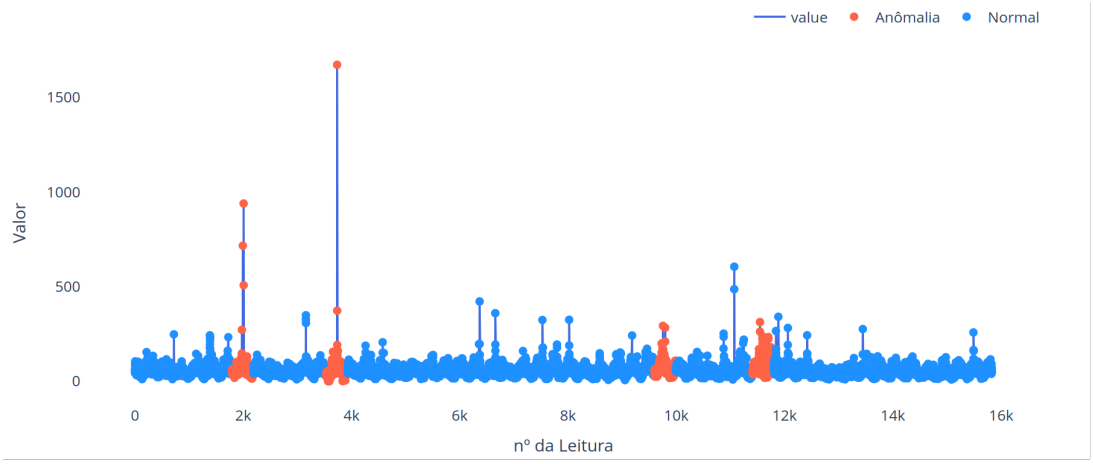
Fonte: Elaboração própria.

5.2 Subconjunto de dados Twitter com anomalias

Analisando agora as séries temporais correspondentes aos sinais reais (não artificiais), que trazem uma maior aplicação prática para as técnicas descritas aqui, sinais os quais não são simples de identificar anomalias e apenas especialistas conseguem apontar. Para realizar essa análise visual detalhada e comparativa de detecção de anomalias, são apresentadas duas figuras de séries temporais utilizando subconjunto de dados do Twitter. A Figura 23 apresenta as anomalias verdadeiras do conjunto de dados, que são destacadas de forma explícita ao longo da série temporal. Essas marcações servem como referência para validar visualmente os resultados obtidos pelos modelos de detecção de anomalias. A identificação precisa das anomalias reais é essencial, pois elas representam eventos que o modelo deve ser capaz de detectar, permitindo uma avaliação inicial de sua eficácia.

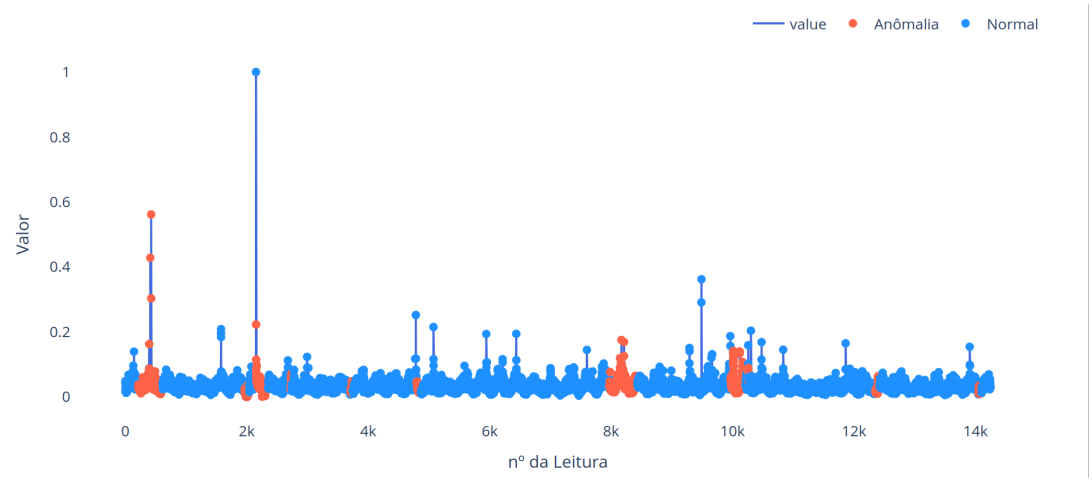
Em seguida, a Figura 24 mostra as anomalias que foram identificadas (estimadas $\hat{A}_2$) pelo modelo DDQN proposto, o qual resultou no F1-*Score* de $0{,}641 \pm 0{,}005$ apresentado na Tabela 4. Ao comparar essas duas figuras, é possível avaliar de maneira mais interpretável o desempenho do modelo, verificando sua capacidade de identificar com precisão os eventos anômalos previamente demarcados. Essa comparação visual proporciona uma análise intuitiva da precisão do modelo, considerando tanto sua capacidade de detectar todas as anomalias verdadeiras quanto sua habilidade de minimizar falsos negativos, a qual é bastante visível nos sinais citados. A recompensa para verdadeiro negativo utilizada foi de $0{,}05$.

Figura 23 – Dados do Twitter com anomalias verdadeiras.



Fonte: Elaboração própria.

Figura 24 – Dados do Twitter com anomalias detectadas.

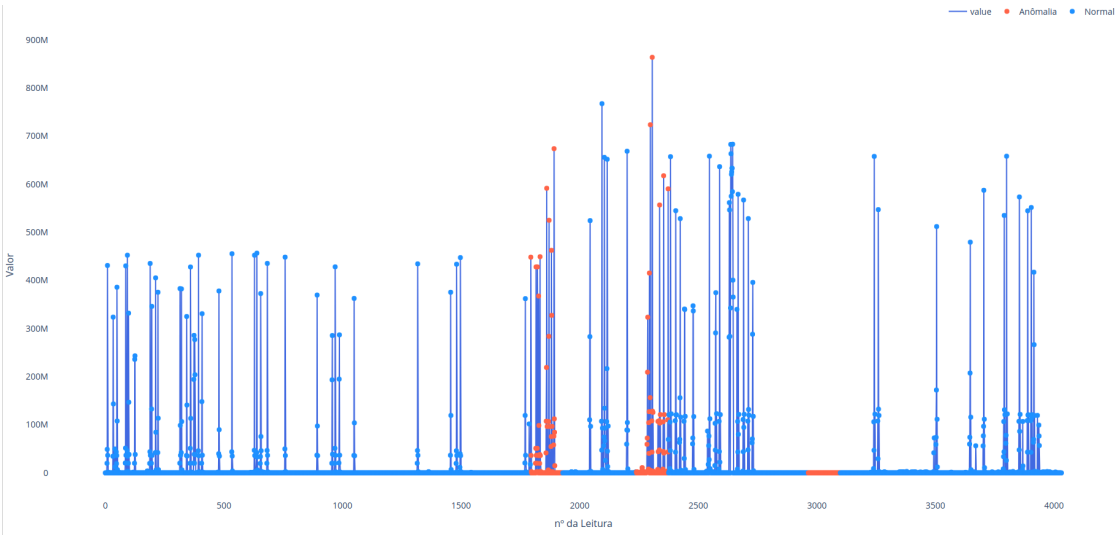


Fonte: Elaboração própria.

5.3 Subconjunto de dados AWS com anomalia

Seguindo a avaliação do desempenho do modelo DDQN proposto, são apresentadas duas figuras de séries temporais usando o subconjunto AWS, correspondendo a dados reais (não artificiais). A Figura 25 mostra as anomalias rotuladas (verdadeiras) presentes no subconjunto de dados ao longo da série temporal. Essas marcações servem para validar visualmente os resultados dos modelos de detecção de anomalias, servindo como uma referência considerável para identificar eventos que o modelo deve ser capaz de detectar. A identificação precisa dessas anomalias permite uma avaliação inicial da eficácia do modelo.

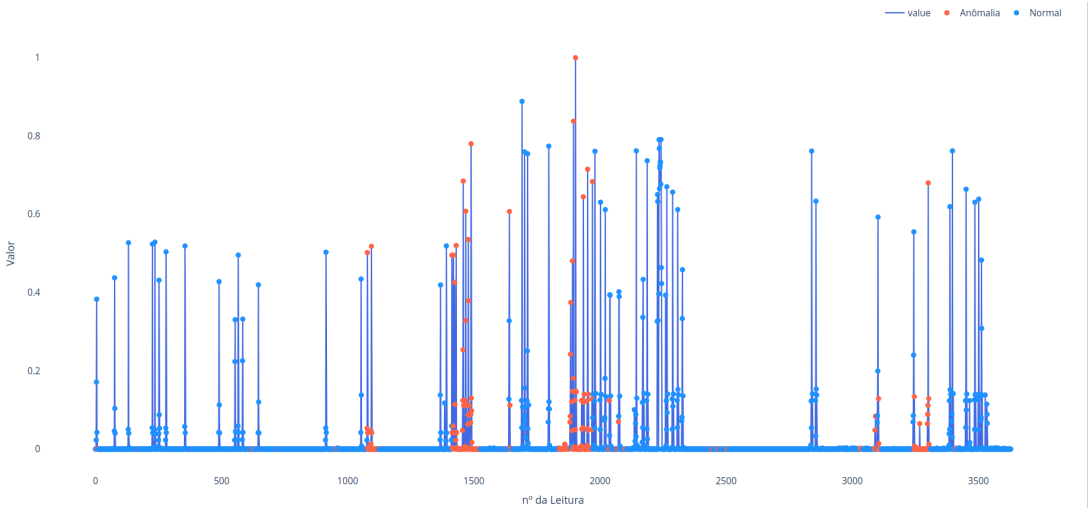
Figura 25 – Sinal AWS com anomalias verdadeiras.



Fonte: Elaboração própria.

Já na Figura 26 são mostradas as anomalias identificadas pelo modelo DDQN, com o valor de $0{,}423$ de F1-Score obtido, conforme indicado na Tabela 4. Ao comparar as duas figuras, observa-se que o modelo conseguiu identificar a maioria das anomalias. No entanto, ele apresentou dificuldades em detectar completamente uma anomalia ocorrida por falta de sinal, marcando apenas pequenos pontos na região como anômalos, em vez de todo o evento, conforme visto na Figura 25. Além disso, uma quantidade significativa de falsos positivos foi observada, o que impactou negativamente o F1-*Score*. A recompensa para verdadeiro negativo utilizada foi de $0{,}29$.

Figura 26 – Sinal AWS com anomalias detectadas.

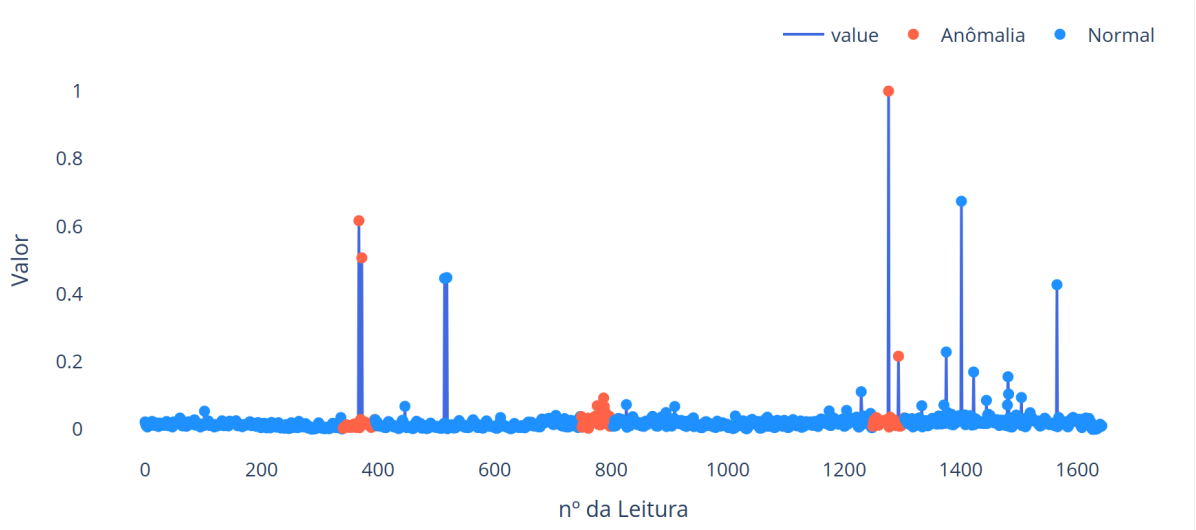


Fonte: Elaboração própria.

5.4 Subconjunto de dados AdExchange com anomalia

Aqui, o subconjunto de dados reais (não artificiais) AdExchange é avaliado. A Figura 27 mostra as anomalias rotuladas (verdadeiras) presentes no subconjunto de dados ao longo da série temporal, novamente as marcações de anomalia são mostradas para a avaliação visual.

Figura 27 – Sinal AdExchange com anomalias verdadeiras.



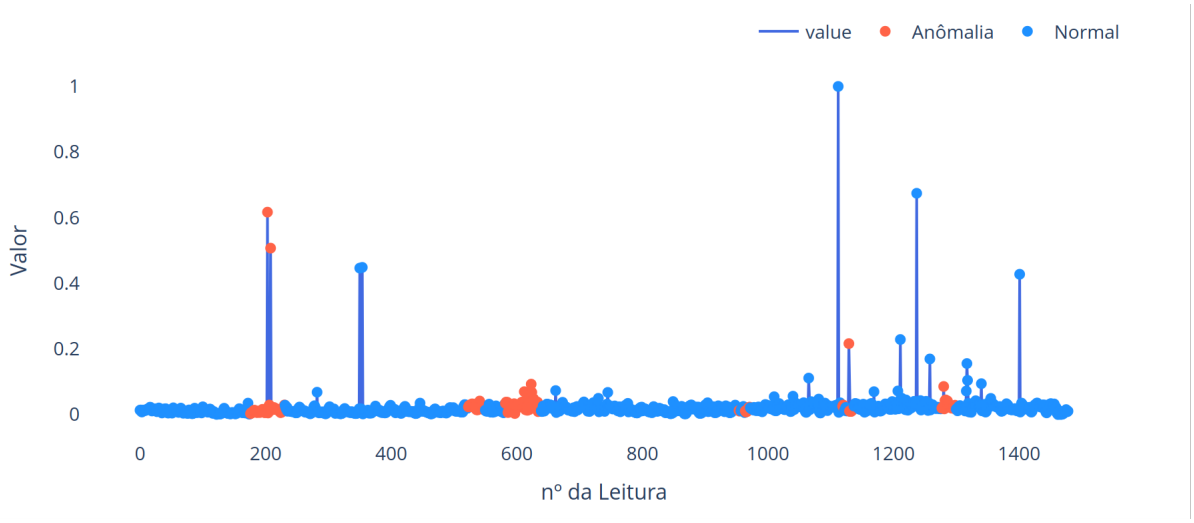
Fonte: Elaboração própria.

Na Figura 28, são apresentados os resultados das anomalias detectadas pelo modelo DDQN, o qual obteve um F1-*Score* de $0{,}516$, conforme descrito na Tabela 4. Embora o modelo tenha identificado a maioria das anomalias, ele encontrou dificuldades em reconhecer integralmente um evento de alto pico, o qual visualmente é perceptível por ser o ponto máximo do sinal, marcando apenas o trecho da sequência como um evento anômalo, conforme ilustrado na Figura 28. Outro aspecto observado foi a quantidade elevada de falsos positivos, o que contribuiu para uma redução no F1-*Score*. Para este modelo, foi utilizada uma recompensa de $3{,}6$ para os verdadeiros negativos.

5.5 Dados Traffic Real com anomalia

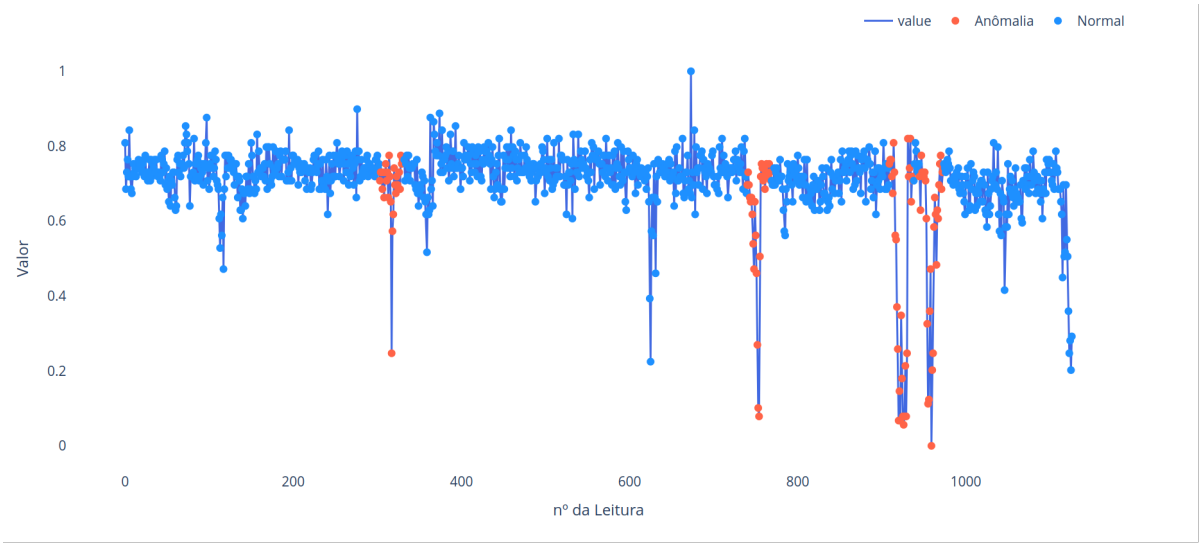
A Figura 29 mostra as anomalias reais presentes no subconjunto de dados Traffic Real ao longo da série temporal. Conforme outras já citadas, serve para análise visual das detecções que serão feitas pela rede, logo abaixo. Os dados também descrevem eventos reais e não artificiais.

Figura 28 – Sinal AdExchange com anomalias detectadas.



Fonte: Elaboração própria.

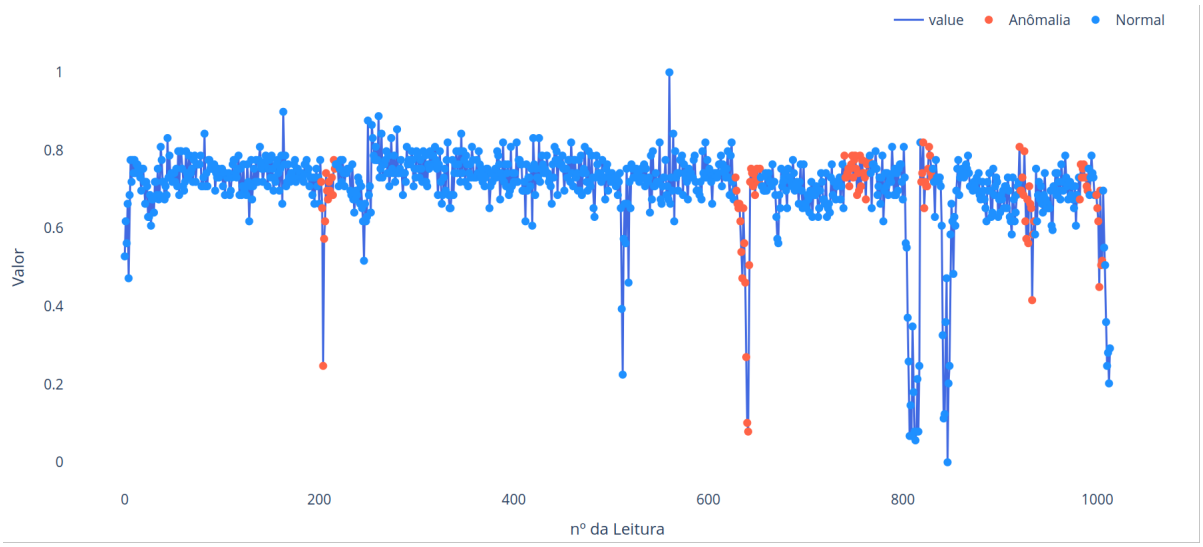
Figura 29 – Sinal Traffic Real com anomalias verdadeiras.



Fonte: Elaboração própria.

Na Figura 30 são mostradas as anomalias identificadas pelo modelo DDQN, com um F1-*Score* obtido de $0{,}471$, conforme indicado na Tabela 4. Ao comparar as janelas de detecção, houve uma detecção completa apenas nas duas anomalias iniciais, tendo apresentado falsos positivos e inexatidão no restante do sinal. No entanto, ao compararmos na Tabela supracitada, é possível identificar uma boa performance. A recompensa para verdadeiro negativo utilizada foi de 4.

Figura 30 – Sinal Traffic real com anomalias detectadas



Fonte: Elaboração própria

É importante ressaltar que os sistemas de detecção de anomalias desenvolvidos e explorados neste trabalho foram submetidos a cenários de alta complexidade devido a escolha do conjunto de dados do Numenta Anomaly Benchmark (NAB), o qual trouxe grande relevância aos resultados pelos desafios associados a trabalhar com séries temporais advindas do mundo real, com ruídos e padrões irregulares. Essa experiência destacou não apenas as dificuldades, mas também o impacto potencial deste estudo, que ecoa tanto em aplicações práticas quanto no avanço acadêmico. O processo de aprendizado, guiado pelos princípios do aprendizado por reforço, reflete um esforço contínuo, onde o agente, interagindo com o ambiente, aprende e se adapta. Cada decisão acertada, fruto dessa interação, marca um pequeno triunfo no caminho para a compreensão de cenários complexos. Também é importante ressaltar que o *framework*, proposto e disponibilizado neste trabalho de conclusão de curso oferece uma flexibilidade singular: ele não apenas treina modelos em um ambiente controlado, mas os prepara para enfrentar dados reais e desafiadores, como os do Twitter. Mais do que uma ferramenta técnica, ele se torna um convite. Um convite para que outros explorem seus próprios dados, ajustando poucos hiperparâmetros, como a taxa de penalidade (investigada) e outras possibilidades de melhorias citadas. Desse modo, amplia-se a aplicabilidade do *framework* proposto em futuros trabalhos, bem como em grupos de pesquisa e instituições relacionadas.

6 Conclusões e Comentários Finais

Este trabalho teve como objetivo principal o estudo e a aplicação da técnica de Double Deep Q-Network (DDQN) para a detecção de anomalias em séries temporais. Durante o desenvolvimento, definimos e executamos o escopo do problema, focando na aplicação da DDQN, explorando inicialmente as dificuldades da detecção de anomalias, como o acesso limitado a conjuntos de dados e a complexidade inerente às aplicações práticas. Utilizamos *benchmarks* bem estabelecidos para garantir a reprodutibilidade e possibilitar comparações futuras, conforme discutido na Seção 3.

Após uma criteriosa revisão no conjunto de dados, o foco foi direcionado para o processamento desses dados, assim como para a caracterização do modelo, que incluiu a definição do espaço de estados e ações, da função de recompensa, da arquitetura de rede neural e da implementação do algoritmo DDQN. Através dessa abordagem, foi possível abordar tanto os desafios teóricos quanto os práticos na aplicação dessa técnica de aprendizado de máquina.

A implementação foi bem-sucedida, alcançando resultados que se mostraram competitivos em comparação com técnicas estabelecidas e emergentes no campo de ML. Em particular, o modelo destacou-se na detecção de anomalias em dados provenientes do Twitter, liderando com a maior pontuação no F1-*Score*, o que reforça a eficácia do método proposto.

Os resultados obtidos foram consistentes e validados utilizando um modelo de avaliação já consagrado por pesquisadores de instituições renomadas, como o MIT. O framework desenvolvido foi disponibilizado para que futuras pesquisas possam explorar e aprimorar os métodos aqui apresentados, dada a natureza geral adotada no trabalho e a vasta possibilidade de aprofundamento em aspectos específicos da técnica, com o objetivo de elevar ainda mais os resultados obtidos.

6.1 Trabalhos futuros

Para aumentar a robustez dos resultados, sugere-se a realização de estudos acadêmicos focados na coleta e identificação de anomalias, bem como na criação de métricas mais variadas e menos complexas para bases de comparação. Além disso,

sugere-se que a técnica desenvolvida e apresentada neste trabalho seja aplicada a outros conjuntos de dados, utilizando diferentes métricas de comparação, para avaliar sua eficácia em diferentes contextos.

Mesmo sem a inclusão de novos conjuntos de dados, há um amplo espaço para melhorias na estrutura da rede DDQN. Um aprofundamento na definição do espaço de estados e ações, a fim de encontrar representações de dados mais eficazes, é uma área promissora. Além disso, pode-se aprofundar a investigação de arquiteturas de redes neurais que proporcionem resultados superiores, bem como o refinamento dos hiperparâmetros.

6.2 Considerações finais

Este trabalho foi fruto de um caminho que se iniciou no âmbito do GPEB – Grupo de Pesquisas em Engenharia Biomédica, por meio do projeto de pesquisa intitulado *Desenvolvimento de um ambiente virtual de aprendizagem didático para o ensino de aprendizado de máquina e inteligência artificial aplicado ao treinamento de veículos autônomos*. Essa trajetória foi enriquecida pelas experiências vividas nas disciplinas de atividades complementares voltadas a ML e RL, também na disciplina de Projeto Integrador III, onde conceitos fundamentais foram aplicados e solidificados em um tutorial, para fomento do uso de RL, denominado *Tutorial ML agents - Simulação de veículos autônomos em ambiente 3D usando reinforcement learning* e disponível em <https://github.com/joaoOldenburg/Reinforcement_learning_with_ML_AGENTS_in_unity>. Como resultado desse percurso, foi possível alcançar uma evolução significativa, partindo de métodos de RL tradicionais para abordagens mais avançadas de DRL, migrando de Q-Learning para Deep Q-Learning.

Um marco importante da trajetória foi a publicação e apresentação do artigo, intitulado *Simulação de veículos autônomos em Ambientes 2D usando aprendizado por reforço aplicados para a solução de Labirintos* (disponível em <https://sbic.org.br/wp-content/uploads/2023/10/ST27/CBIC_2023_paper053.pdf>), no XVI Congresso Brasileiro de Inteligência Computacional (CBIC), que consolidou não apenas o domínio técnico necessário, mas também a base matemática indispensável para o desenvolvimento deste Trabalho de Conclusão de Curso. Essa trajetória de aprendizado e crescimento evidencia os esforços realizados e reforça o impacto positivo do trabalho,

tanto no ambiente acadêmico quanto na formação pessoal e profissional.

E para sua continuidade, com a configuração adequada de seus hiperparâmetros, o *framework* desenvolvido neste trabalho está apto para ser aplicado na detecção de anomalias em séries temporais. Com o intuito de incentivar investigações futuras e novas aplicações de algoritmos de Aprendizado de Máquina, disponibilizamos o *framework* através do seguinte link: <https://github.com/joaoOldenburg/Anomaly_detection_with_deep_Q_network>.

Referências

ARSHAD, K. et al. Deep reinforcement learning for anomaly detection: A systematic review. **IEEE Access**, v. 10, nov. 2022. Citado 3 vezes nas páginas 15, 16 e 28.

BOLLERSLEV, T. Generalized autoregressive conditional heteroskedasticity. **Journal of Econometrics**, Elsevier, v. 31, n. 3, p. 307–327, 1986. Citado na página 20.

BOUDT, K.; DANíELSSONB, J.; LAURENT, S. Volatility models and volatility dynamics. In: . [S.l.]: Elsevier, 2013. Citado 2 vezes nas páginas 20 e 21.

BOZZETTO, M. U. **Gans para detecção de anomalias em séries temporais: um estudo de caso**. 2023. Trabalho de Conclusão de Curso (TCC), Universidade Federal do Rio Grande do Sul (UFRGS). Citado 3 vezes nas páginas 17, 18 e 39.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. **ACM Comput. Surv.**, v. 41, n. 3, p. 1–58, jul. 2009. Citado 2 vezes nas páginas 14 e 17.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection for discrete sequences: A survey. **IEEE Trans. Knowl. Data Eng.**, v. 24, n. 5, p. 823–839, maio 2012. Citado na página 14.

CHERKASSKY, V. The nature of statistical learning theory. **IEEE Transactions on Neural Networks**, v. 8, n. 6, p. 1564–1564, 1997. Citado na página 21.

DUDA, R. O. **Pattern Classification**. [S.l.]: Wiley-Interscience, 2001. Citado na página 24.

GALVãO, Y. M. Utilizando python e métricas de aplicações para detecção de anomalias. **Medium**, 2024. Citado na página 19.

GEIGER, A.; ALNEGHEIMISH, D. L. andSarah; CUESTA-INFANTE, K. V. A. Tadgan: Time series anomaly detection using generative adversarial networks. In: **IEEE International Conference on Big Data (Big Data)**. [S.l.: s.n.], 2020. Citado 4 vezes nas páginas , 42, 52 e 53.

GOERNITZ, N. et al. Hidden markov anomaly detection. In: BACH, F.; BLEI, D. (Ed.). **Proc. in the 32nd International Conference on Machine Learning**. Lille, France: PMLR, 2015. v. 37, p. 1833–1842. Citado na página 23.

GRAESSER, L.; KENG, W. L. **Foundations of Deep Reinforcement Learning**. [S.l.]: Editora Blücher, 2020. Citado 8 vezes nas páginas 28, 31, 32, 33, 34, 35, 36 e 48.

GUPTA, M. et al. Outlier detection for temporal data: A survey. **IEEE Transactions on Knowledge and Data Engineering**, v. 26, n. 9, p. 2250–2267, 2014. Citado na página 18.

HAN, J.; PEI, J.; KAMBER, M. **Data Mining: Concepts and Techniques**. [S.l.]: Elsevier, 2011. Citado na página 19.

HUNDMAN, K. et al. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In: ACM. **Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. [S.l.], 2018. p. 387–395. Citado na página 42.

INJADAT, M. et al. Bayesian optimization with machine learning algorithms towards anomaly detection. In: **Proc. IEEE Global Commun. Conf. (GLOBECOM)**. [S.l.: s.n.], 2018. p. 1–6. Citado na página 14.

JUVONEN, A.; SIPOLA, T.; HäMäLäINEN, T. Online anomaly detection using dimensionality reduction techniques for http log analysis. **Computer Networks**, v. 91, p. 46–56, 2015. ISSN 1389-1286. Citado na página 14.

LAVIN; ALEXANDER; SUBUTAI, A. Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In: IEEE. **2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)**. [S.l.], 2015. p. 38–44. Citado 2 vezes nas páginas 38 e 39.

LI, Y.; WU, J. Low latency cyberattack detection in smart grids with deep reinforcement learning. **International Journal of Electrical Power Energy Systems**, 2022. Citado na página 25.

LOPES, A. **Com R$ 2,5 bi em prejuízos por fraudes, BC estuda responsabilizar bancos por golpes; IA pode ajudar?** 2023. Acessado em: 02 de agosto de 2024. Disponível em: <https://exame.com/inteligencia-artificial/com-r-25-bi-em-prejuizos-por-fraudes-bc-estuda-responsabilizar-bancos-por-golpes-ia-pode-ajudar/>. Citado na página 15.

LOPEZ-MARTIN, M.; CARRO, B.; SANCHEZ-ESGUEVILLAS, A. Application of deep reinforcement learning to intrusion detection for supervised problems. **Expert Systems with Applications**, Elsevier, 2020. Citado 2 vezes nas páginas 34 e 36.

MARTINS, B. A. **Detecção de anomalis em sérires temporais veriáveis com LSTM-RNNS**. Dissertação (Mestrado) — Universidade do Estado do Rio de Janeiro, 2020. Citado na página 22.

MNIH, V. et al. Human-level control through deep reinforcement learning. **Nature**, v. 518, n. 7540, p. 529–533, jun 2015. Citado 3 vezes nas páginas 30, 31 e 32.

MOLINA, A. L. B. **WEAPON: AN UNSUPERVISED LEARNING ARCHITECTURE FOR USER BEHAVIOR ANOMALY DETECTION**. Dissertação (Mestrado) — Universidade de Brasília, 2022. Citado na página 17.

NHCAA. **National Health Care Anti-Fraud Association**. 2023. Acessado em: 03 de dezembro de 2024. Disponível em: <https://www.nhcaa.org/>. Citado na página 15.

PORSCH, D. et al. Simulação de veículos autônomos em ambientes 2d usando aprendizado por reforço aplicados para a solução de labirintos. In: **Anais Congresso Brasileiro de Inteligência Computacional (CBIC)**. Salvador, Brasil: [s.n.], 2023. p. 1–7. Citado na página 30.

RUFF, L. et al. A unifying review of deep and shallow anomaly detection. **Proceedings of the IEEE**, v. 109, 2021. Citado na página 17.

SILVA, A. V. C. **Middleware para detecção de anomalias no compartilhamento de conteúdo para sistemas baseados em blockchain**. Dissertação (Mestrado) — Universidade de São Paulo (USP), 2020. Citado na página 15.

SILVA, E. d. S.; HEMKEMAIER, D. Inteligencia artificial aplicada em robôs autônomos para a solução de labirintos dinâmicos. In: **Proc. in the 8th Workshop of Robotics in Education**. Curitiba, Brasil: [s.n.], 2017. p. 13–18. Citado na página 29.

SINGH, N.; OLINSKY, C. Demystifying numenta anomaly benchmark. In: **Proc. in the International Joint Conference on Neural Networks (IJCNN)**. Anchorage, USA: [s.n.], 2017. Citado na página 43.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. 2st. ed. London, UK: MIT press, 2018. Citado 3 vezes nas páginas 25, 29 e 30.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introdução ao Datamining: Mineração de Dados**. [S.l.]: Ciência Moderna, 2009. Citado na página 20.

WATSON. Ibm global ai adoption index 2022: New research commissioned by ibm in partnership with morning consult. may 2022. [Online]. Disponível em: <https://www.ibm.com/downloads/cas/GVAGA3JP>. Citado na página 14.

YANAI, F. K. **Detecção de anomalias no funcionamento de software com machine learning**. Dissertação (Mestrado) — Pontifícia Universidade Católica de São Paulo (PUC-SP), 2020. Citado na página 15.

YU, D.; ZHANG, A.; GAO, Z. Fault diagnosis using redundant data in analog circuits via slime module algorithm for support vector machine. **Journal of Ambient Intelligence and Humanized Computing**, v. 14, 2023. Citado na página 22.

YU, M.; SUN, S. Policy-based reinforcement learning for time series anomaly detection. **Engineering Applications of Artificial Intelligence**, v. 95, oct. 2020. Citado na página 26.