UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

THIAGO DA SILVA ARAÚJO

Otimização de Portfólio Financeiro utilizando Aprendizado por Reforço

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Dennis Giovani Balreira
Co-orientador: Prof. Dr. Paulo Salgado Gomes de Mattos Neto

Porto Alegre
2023

“Esquecer é uma necessidade. A vida é uma lousa, em que o destino, para escrever um novo caso, precisa de apagar o caso escrito.”

— MACHADO DE ASSIS

AGRADECIMENTOS

Agradeço ao professor coorientador Paulo Salgado por termos iniciado a pesquisa para esse trabalho e todo o suporte dado nesse período. Ao professor orientador Dennis Giovani por aceitar orientar um tema que não é a sua especialidade, mesmo assim contribuiu muito para o trabalho. Aos colegas do grupo GPPD (grupo de programação paralela e distribuída) e da equipe de robótica RobôCIn, pela ajuda ao longo do desenvolvimento do trabalho. Aos amigos e família por terem auxiliado durante o processo de elaboração deste trabalho. Além disso, as inúmeras pessoas que me ajudaram durante esse período.

RESUMO

Portfólio financeiro pode ser definido como o conjunto de ativos que um investidor detém. Estes ativos podem ser tanto investimentos em renda fixa como em renda variável. Otimização de portfólio financeiro é uma área que visa maximizar os resultados de uma carteira de investimento. Essa campo tem tido muita relevância recentemente, gerando diversos modelos matemáticos para melhorar os resultados dos portfólios. Porém, devido a sua complexidade, muitos dos modelos não conseguem se aproximar o suficiente da realidade a ponto de se adaptar a cenários mais extremos. Por exemplo, dentre eles a queda acentuada que a bolsa de valores sofreu com a pandemia da Covid-19. Em busca de solucionar o problema de adaptação, esse trabalho aplica 5 algoritmos de Aprendizado por Reforço para otimizar os pesos de carteiras de ações em diferentes mercados e obter o maior retorno possível. Além disso, foram exploradas combinações de execução dos algoritmos de forma concorrente, com o objetivo de reduzir o tempo global de treinamento e tornar o processo de melhora dos modelos mais rápido. O retorno final obtido no mercado Brasileiro foi de 1,91 e no Americano foi de 1,86. O tempo global de treinamento dos algoritmos reduziu em 33% e o consumo energético em 15%. Dessa forma, foi possível realizar adaptações nos modelos de forma mais rápida e personalizada para cada mercado financeiro.

Palavras-chave: Otimização de portfólio. Aprendizado por Reforço. Mercado de ações. Concorrencia. Paralelismo.

ABSTRACT

A financial portfolio can be defined as the set of assets that an investor holds. These assets can be either fixed-income or variable-income investments. Financial portfolio optimization is an area that aims to maximize the results of an investment portfolio. This field has been very relevant recently, generating various mathematical models to improve portfolio results. However, due to their complexity, many of the models cannot get close enough to reality to adapt to more extreme scenarios. For example, the sharp drop in the stock market caused by the Covid-19 pandemic. In order to solve the adaptation problem, this work applies 5 Reinforcement Learning algorithms to optimize the weights of stock portfolios in different markets and obtain the highest possible return. In addition, combinations of running the algorithms concurrently were explored, with the aim of reducing the overall training time and making the model improvement process faster. The final return obtained in the Brazilian market was 1.91 and in the American market it was 1.86. The overall training time of the algorithms was reduced by 33% and the energy consumption by 15%. In this way, it was possible to make adaptations to the models in a faster and more personalized way for each financial market.

Keywords: Portfolio Optimization. Reinforcement Learning. Mercado de ações. Concurrency. Parallel.

LISTA DE ABREVIATURAS E SIGLAS

A2C	*Advantage Actor Critic*
AdaBoost	*Adaptative Boosting*
ANN	*Artificial Neural Network*
BDR	*Brazilian Depositary Receipts*
CDB	Certificado de depósito bancário
CDI	Certificado de depósito interbancário
CRA	Certificado de recebíveis do agronegócio
CRI	Certificado de recebíveis imobiliários
CPU	*Central processing unit*
DDPG	Deep Deterministic Policy Gradient
ENEM	Exame Nacional do Ensino Médio
ETF	*Exchange-traded fund*
GPU	*Graph processing unit*
HPC	*High performance computing*
IPO	*Initial public offering*
KNN	*K-Nearest Neighbors*
LCA	Letra de crédito do agronegócio
LCI	Letra de crétito imobiliário
MV	*Mean Variance*
NAN	*Not a number*
MLP	*Multilayer perceptron*
NUMA	*Non-uniform memory access*
P/L	Preço sobre o lucro
PG	*Policy Gradient*

PPO	*Proximal Policy Optimization*
RL	Aprendizado por Reforço (*reinforcement learning*)
SAC	*Soft Actor Critic*
SVR	*Support Vector Machine Regression*
TD3	*Twin-Delayed Deep Deterministic Policy Gradient*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Portfólio financeiro pode ser definido como o conjunto de ativos que um investidor detém. Estes ativos podem ser tanto investimentos em renda fixa como em renda variável [1]. Em renda fixa, podem ser citados: CDB (certificado de depósito bancário), CRA (certificado de recebíveis do agronegócio), CRI (certificado de recebíveis imobiliários), debêntures, LCA (letra de crédito do agronegócio), LCI (letra de crédito imobiliário) e títulos públicos. Por outro lado, em renda variável, tem-se como exemplos: ações, criptomoedas, derivativos, ETF (*exchange-traded fund*), fundos de investimento e fundos imobiliários.

Os investimentos em renda fixa podem ser pré-fixados, quando a rentabilidade é definida no momento do investimento; pós-fixados, onde a rentabilidade está atrelada a algum índice, por exemplo o CDI (certificado de depósito interbancário), dessa forma a rentabilidade pode variar conforme a variação do índice; ou híbridos, que são a união do pré-fixado com o pós-fixado. Assim, parte da rentabilidade já é determinada e o restante está atrelado a algum índice, por exemplo a SELIC, que é a taxa básica de juros da economia. Os investimentos em renda variável são divididos em diversas categorias e os preços dos ativos variam diariamente, não sendo possível determinar a rentabilidade no momento da aplicação.

Em geral, investidores possuem diferentes tipos de perfis, que variam com base no objetivo e tolerância ao risco nos investimentos. Dentre eles, pode-se citar o perfil conservador, que prioriza maior segurança, fazendo com que os investimentos possuam rendimentos menores. O perfil moderado, por outro lado, assume um pouco mais de risco para ter um rendimento maior. Por fim, o perfil arrojado assume grandes riscos por grandes rendimentos. A Figura 1.1 mostra possíveis alocações com base no tipo de perfil de investidor.

A otimização de portfólio financeiro busca maximizar a rentabilidade dos investimentos. Para isso, pode-se otimizar a escolha dos ativos a serem investidos. Por exemplo, dadas as ações da bolsa de valores brasileira, são escolhidas oito ações para se investir, ou a proporção investida em cada ativo, por exemplo, o valor que será investido em cada uma das oito ações. Tem-se utilizado otimizações para maximizar o retorno e minimizar o risco, dado o perfil de investidor. Essa área tem tido muita relevância, sendo criados diversos modelos que tem ajudado a melhorar os resultados e gerar diferentes possibilidades

[1] https://www.sproutfi.com/pt-BR/insights/portfolio-financeiro/, acessado em 02 de agosto de 2023

de carteiras de investimento.

A Teoria Moderna de Portfólios é a base para a otimização de portfólio financeiro. O modelo *Mean Variance* (MV) foi pioneiro, focando em obter o maior retorno com o menor risco possível (MARKOWITZ, 1952). Por meio desse modelo, surgiu o conceito da fronteira eficiente, que é capaz de determinar os melhores retornos possíveis para uma dada configuração de retorno esperado e risco. A Figura 1.2 mostra a fronteira eficiente, sendo representada por meio da linha laranja. Os pontos em laranja são as carteiras ideias, e os pontos em azul representam carteiras diversas. Quanto mais próximo da linha laranja, maior a eficiência da carteira. A partir disso podemos ver que alguns pontos em azul possuem boa eficiência e outros uma eficiência ruim.

Além disso, a Teoria Moderna de Portfólios usa a diversificação como forma de reduzir o risco do portfólio. Dessa maneira, uma carteira de investimentos com múltiplos ativos tende a ter um risco menor do que um ativo individualmente (MARKOWITZ, 1967).

1.1 Motivação

O número de investidores (pessoas físicas) na bolsa de valores brasileira é de 17,6 milhões[2], o que representa em torno de 8% da população do Brasil. Já nos Estados Unidos, esse número é de 158 milhões[3], representando 47% da população dos Estados Unidos. Isso mostra que a bolsa brasileira tem um grande potencial de crescimento, podendo gerar oportunidades para aplicações modelos de otimizações de portfólio financeiro.

[2]https://einvestidor.estadao.com.br/mercado/numero-investidores-b3-aumenta-2022/, acessado em 08 de agosto de 2023

[3]https://www.fool.com/research/how-many-americans-own-stock/, acessado em 08 de agosto de 2023

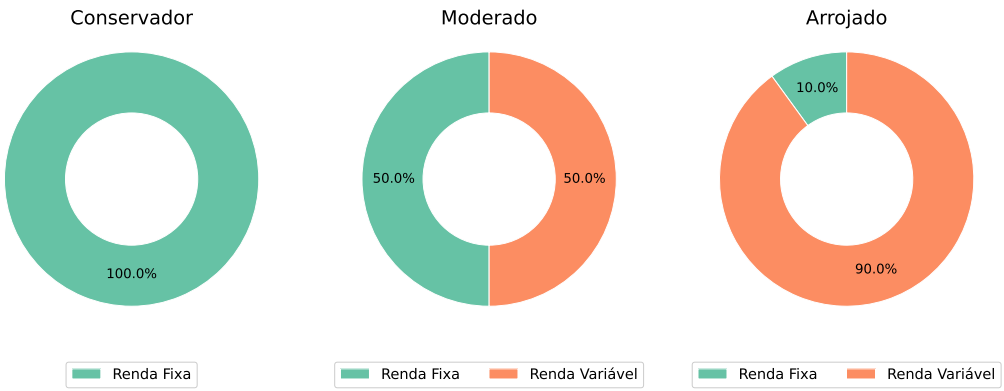


Figura 1.1 – Gráficos exemplificando a alocação de um portifólio financeiro para cada perfil de investidor, com base no percentual aplicado para cada categoria de investimento.

A otimização de portfólio financeiro é um assunto com grande relevância e aberto a novas soluções. Diversas abordagens foram utilizadas para solucionar esse problema, incluindo: modelos matemáticos (SHARPE, 1998)(BLACK; LITTERMAN, 1992), inteligência de enxame (ERTENLICE; KALAYCI, 2018)(ZHU et al., 2011), algoritmos genéticos (CHEONG et al., 2017)(CHEN et al., 2019), aprendizado de máquina (CHEN et al., 2021a)(CHAWEEWANCHON; CHAYSIRI, 2022), aprendizado profundo (SEN; DUTTA; MEHTAB, 2021)(MA; HAN; WANG, 2021) e Aprendizado por Reforço (SATO, 2019)(SOLEYMANI; PAQUET, 2021).

Aprendizado por Reforço tem sido muito utilizado em contextos onde o ambiente é dinâmico, obtendo resultados promissores (POLYDOROS; NALPANTIDIS, 2017)(LEI; ZHANG; DONG, 2018)(BING et al., 2022). O ambiente do mercado financeiro pode ter grandes variações em um curto período de tempo, então a capacidade de adaptação de um modelo é muito importante para se obter bons resultados, por esse motivo o uso de Aprendizado por Reforço pode gerar modelos com bons desempenhos.

1.2 Objetivos

O trabalho tem como objetivo estudar abordagens baseadas em Aprendizado por Reforço na otimização de portfólio financeiro. Especificamente, na escolha da proporção investida em cada ação de uma carteira de investimentos, buscando maximizar o seu re-

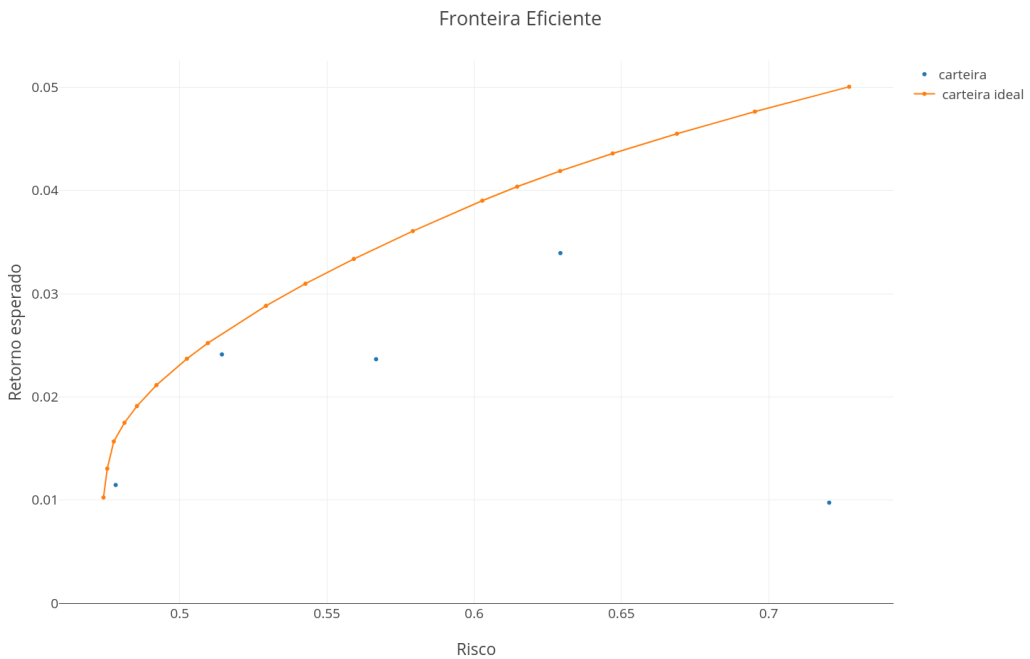


Figura 1.2 – Gráfico ilustrando a fronteira eficiente e a disposição de carteiras diversas e ideias.

torno. A partir desse estudo, serão propostas modificações com o objetivo de melhorar o desempenho e a eficiência energética das abordagens.

Além disso, a aplicação deve permitir a comparação dos resultados entre os algoritmos treinados e *benchmarks* utilizados pelo mercado financeiro. A partir disso, será possível mensurar o desempenho do modelo com estratégias já utilizadas pelo mercado.

Por fim, será explorado o uso da computação concorrente para executar mais de um algoritmo de forma concorrente, com o objetivo de encontrar a melhor combinação de execução dos algoritmos para reduzir o tempo total de treinamento dos algoritmos e o consumo energético. Ademais, a redução do tempo facilitará o processo de melhora dos desempenhos dos algoritmos, pois cada melhoria proposta será validada de forma mais rápida.

1.3 Sobre o trabalho

Este trabalho foi desenvolvido para fins acadêmicos, não fornecemos nenhuma indicação para o mercado financeiro nem nos responsabilidamos pelo uso e retornos obtidos dos modelos desenvolvidos.

1.4 Organização

O resto deste trabalho está dividido em cinco capítulos. O Capítulo 2 apresenta a fundamentação teórica, onde são introduzidos definições importantes para o entendimento do trabalho, incluindo uma introdução aos principais conceitos da bolsa de valores e sobre Aprendizado por Reforço. O Capítulo 3 descreve os trabalhos relacionados, que foram utilizados como inspiração para o desenvolvimento deste trabalho. O Capítulo 4 apresenta a metodologia utilizada para a execução do trabalho, enquanto o Captítulo 5 mostra os resultados obtidos e uma discussão acerca disso. Por fim, o Capítulo 6 apresenta as considerações finais sobre o trabalho, incluindo limitações e possíveis trabalhos futuros.

2 CONCEITOS BÁSICOS

O objetivo deste capítulo é fornecer os conceitos básicos para um completo entendimento do trabalho. Os conceitos chaves apresentados consistem em uma apresentação sobre a bolsa de valores, mercado de ações, aprendizado de máquina, Aprendizado por Reforço, computação concorrente e computação paralela.

2.1 Bolsa de Valores

Uma bolsa de valores é um ambiente de negociações financeiras. Nela, negociam-se ações, títulos de dívida, contratos futuros, *commodities*, entre outros (B3, 2023). Dentre os conceitos mais relevantes associados, pode-se destacar:

- Ações: parte do capital social de uma empresa.
- BDR (*Brazilian Depositary Receipts*): títulos de empresas estrangeiras negociadas na bolsa brasileira.
- Contratos futuros: contratos com a expectativa futura dos valores de determinado ativo. Por exemplo, existe o contrato futuro do índice ibovespa e do dólar.
- *Commodities*: contratos futuros de *commodities*, como boi gordo, soja, etc. Refletem a expectativa do preço desses ativos no futuro.
- Fundos de investimentos: utilizam o capital social para realizar investimentos de acordo com a estratégia do fundo.
- Fundos imobiliário: utilizam o capital social do fundo para comprar ativos físicos do mercado financeiro.
- Opções: são direitos de compra e venda de determinado ativo por um valor determiando.
- Títulos públicos e privados: podem ser títulos emitidos pelo tesouro direto ou empresas privadas.

O fluxo de funcionamento da bolsa consiste em negociações entre investidores. Para se comprar determinado ativo, algum investidor deve vendê-lo, e, para se vender, alguém precisa comprá-lo. Os preços variam com base na oferta e demanda, ou seja, caso tenham mais investidores querendo comprar determinado ativo em vez de vender, a tendência é que o preço suba - e assim por diante.

2.1.1 Mercado de Ações

É o ambiente onde são negociadas frações do capital social de empresas, diversos fatores podem provocar mudanças nos preços das ações, entre eles: divulgação de balanços, especulação e fatores econômicos e políticos (CUTLER; POTERBA; SUMMERS, 1988). Isso pode provocar grande variação nos valores das ações, se tornando um investimento de alto risco. Porém, os retornos provenientes por esse investimento podem ser altos e os riscos minimizados.

As empresas entram na bolsa a partir de um IPO (*initial public offering*). É uma forma da empresa abrir o capital social e arrecadar dinheiro com a oferta inicial de ações. Esses valores podem ser utilizados para investimentos estratégicos da empresa, tendo assim o potencial de impulsionar o crescimento da companhia.

Existem diferentes estratégias de operação na bolsa de valores, incluindo o *day trade*, que consiste em realizar a compra e a venda do ativo no mesmo dia, e o *swing trade*, onde as operações de compra e venda são feitas em dias diferentes. O princípio para se ter lucro nos investimentos com as ações é comprar barato e vender caro.

As ações são negociadas durante o período de operação da bolsa de valores, tendo um preço de abertura e fechamento. Ao longo do dia o seu valor pode variar de forma positiva ou negativa. É possível acompanhar o volume financeiro de negociações de uma ação, podendo ser utilizado para definir uma compra ou venda de ativo. Além disso, existem diversos indicadores que são utilizados para analisar as ações. Dentre eles pode-se citar os fundamentalistas, que são ferramentas para analisar os fundamentos da empresa. Por exemplo, o indicador P/L é o preço sobre o lucro e auxilia o investidor a avaliar se o preço do ativo está justo. Outra forma é utilizando indicadores técnicos, que são indicadores que aparecem no gráfico do preço das ações. Por exemplo, o indicador de média móvel pode facilitar a identificação de pontos onde se pode comprar ou vender o ativo.

A rentabilidade é o percentual de ganho a partir de um valor investido. Por exemplo, um investidor que comprou 100 ações PETR4 por R$30,80 e vendeu no mesmo dia por R$31,20 teve um valor total investido de R$3.080,00 e um valor resgatado de R$3.120,00. Nesta transação foi feito o *day trade*, gerando um lucro bruto de R$40,00 e uma rentabilidade de 1,29%. A Tabela 2.1 apresenta a rentabilidade anual da bolsa brasileira.

Tabela 2.1 – Rentabilidade anual da bolsa brasileira (Ibovespa) nos últimos 5 anos

Ano	2018	2019	2020	2021	2022
Rentabilidade	15,03%	31,58%	2,92%	- 11,9%	4,69%

2.2 Aprendizado de Máquina

Aprendizado de máquina é uma subárea da inteligência artificial que busca melhorar o desempenho de sistemas a partir do aprendizado adquirido pela experiência. Os dados possuem uma grande quantidade de informações que podem gerar conhecimento e permitir que se adquira experiência, por isso o objetivo principal do aprendizado de máquina é desenvolver algoritmos de aprendizado que consigam criar modelos a partir dos dados (ZHOU, 2021).

O modelo consiste em um conjunto de regras que o algoritmo aprendeu para realizar determinada tarefa. As tarefas podem ser de classificação, regressão ou clusterização. Na classificação busca-se classificar determinado atributo alvo a partir de outros atributos fornecidos. Por exemplo, dadas determinadas condições de um paciente, o algoritmo classifica se a pessoa está doente ou não. A regressão consiste em modelar relações entre variáveis dependentes e independentes a partir de métodos estatísticos. Por exemplo, dadas algumas informações dos estudantes, realizar a previsão da nota do ENEM de cada estudante (SOTO, 2021). Por fim, tem-se a clusterização, onde o algoritmo faz o agrupamento dos dados com base em semelhanças ou diferenças. Dentre os tipos de aprendizado, tem-se:

1. Supervisionado: essa abordagem possui um conjunto de dados rotulados, os quais são utilizados para fazer o treinamento dos algoritmos, esse tipo de aprendizado possui duas categorias principais, problemas de classificação ou regressão.
2. Não supervisionado: esse tipo de aprendizado possui dados sem rótulos, os algoritmos são utilizados para analisar e agrupar o conjunto de dados. Os modelos conseguem detectar padrões nos dados sem nenhuma intervenção humana, as categorias principais são clusterização e associação.
3. Aprendizado por reforço: o agente é treinado para realizar ações e recebe recompensas ou penalidade em troca.

Figura 2.1 – Exemplos de aplicação de cada tipo de aprendizado.

2.2.1 Aprendizado por Reforço

Aprendizado por Reforço (RL) é a área do aprendizado de máquina onde o agente interage com o ambiente e realiza ações, que são avaliadas por meio da recompensa recebida. A partir dessa métrica, o agente vai aprendendo quais ações são boas e ruins em determinado estado. O objetivo é maximizar a recompensa, visando que o agente aprenda por meio da tentativa e erro (KAELBLING; LITTMAN; MOORE, 1996). Abaixo são definidos alguns conceitos importantes para um melhor entendimento sobre RL:

1. Agente: quem realiza a tomada de decisão.
2. Ambiente: o meio onde o agente atua.
3. Ações: o mecanismo pela qual o agente transiciona pelos estados.
4. Estado: a representação do ambiente em que o agente está.
5. Política: um mapeamento do ambiente de observação para uma distribuição de probabilidade das ações que podem ser tomadas.
6. Recompensa: mecanismo de feedback do ambiente.

A Figura 2.2 exemplifica o fluxo de interação entre o agente e o ambiente. O agente realiza ações no ambiente, que por sua vez informa o novo estado e a recompensa para o agente. A partir desse processo, o aprendizado do agente vai acontecendo. Inicialmente, há uma *exploration* no ambiente, ou seja, o agente escolhe ações que não são

ótimas para descobrir novas possibilidades. Em seguida, ocorre a *exploitation*, utilizando o seu conhecimento prévio para selecionar as acções que maximizam os resultados. Ambas são importantes porque a *exploration* pode fazer com que o agente aprenda caminhos que não seriam encontrados diretamente pela abordagem gulosa, possibilitando assim a saída de mínimos ou máximos locais. Já a *exploitation* auxilia a encontrar mínimos e máximos localmente.

2.2.2 Computação Concorrente

É um paradigma de programação que lida com a execução simultânea de várias tarefas em um sistema de computador, podendo ser implementadas como programas separados ou como um conjunto de *threads* criadas por um único programa. Essas tarefas podem ser executadas por um único processador, vários processadores em um único equipamento ou processadores distribuídos por uma rede.

Em contraste com a computação sequencial, onde as instruções são executadas uma após a outra em uma única linha de execução, a computação concorrente permite que múltiplas linhas de execução (ou *threads*) compartilhem o mesmo espaço de endereçamento e recursos, como memória e processador. Isso possibilita que as tarefas sejam executadas de maneira mais eficiente e rápida, desde que sejam gerenciadas de maneira adequada (AGHA, 1986).

No entanto, a programação concorrente traz desafios, como garantir a sincronização correta entre as threads ou processos, evitando condições de corrida e *deadlocks*. Ferramentas como *locks*, semáforos, monitores e barreiras são usadas para controlar a concorrência e sincronização (SUTTER; LARUS, 2005).

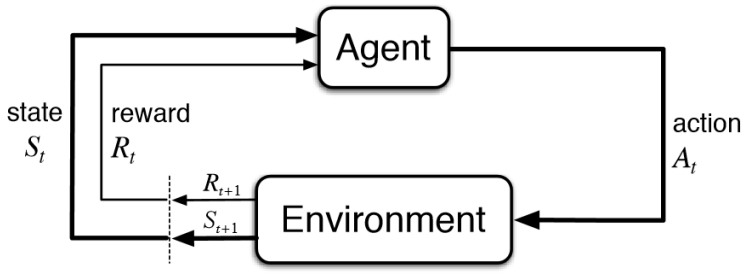


Figura 2.2 – Funcionamento geral do treinamento utilizando Aprendizado por Reforço. O agente interage com o ambiente a partir de uma ação. Na sequência, o ambiente retorna ao agente uma recompensa e um novo estado. Fonte: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

3 TRABALHOS RELACIONADOS

Este capítulo descreve os trabalhos relacionados, que possuem algum grau de relação com o trabalho desenvolvido.

3.1 Otimização de Modelos de Aprendizado por Reforço

A otimização de portfólio financeiro busca maximizar os resultados de uma carteira de investimento. O modelo *Mean Variance* (MV) foi pioneiro, onde se busca maximizar o retorno dos ativos com um determinado risco (MARKOWITZ, 1952). *The Tangency Portfolio*, por outro lado, é um tipo de alocação que busca maximizar o *Sharpe Ratio* (SHARPE, 1994). Essa relação mede os rendimentos ajustados relativos aos riscos, ajudando os investidores a definirem se um alto retorno está compatível com o risco associado. Por fim, *Risk Parity Portfolio* é uma alternativa que utiliza o risco na construção do portfólio de investimentos, tornando-o mais resistente a condições mais extremas do mercado (RONCALLI; WEISANG, 2016).

Além das abordagens clássicas, surgiram modelos de otimização de portfólio financeiro gerados por meio da inteligência artificial. Foi desenvolvido uma abordagem de construção de portfólio com a utilização de um modelo híbrido, sendo aplicado aprendizado de máquina na predição do preço das ações e o modelo *Mean Variance* (MV) para a seleção do portfólio. O modelo híbrido combina o algoritmo *eXtreme Gradient Boosting* (XGBoost) com o algoritmo de inteligência de enxame *Firefly*, para que os hiperparâmetros sejam otimizados. As ações com maior potencial de retono são selecionadas, e o modelo MV é empregado na seleção do portfólio. Os retornos obtidos foram superiores aos métodos tradicionais de otimização de portfólio financeiro (CHEN et al., 2021b).

Outro modelo híbrido foi proposto, consistindo em algoritmos de aprendizado de máquina para a previsão do retorno das ações e o modelo *Mean-VaR (value-at-risk* para a seleção do portfólio. Os algoritmos de regressão utilizados foram *Random Forest*, *Extreme Gradient Boosting* (XGBoost), *Adaptative Boosting* (AdaBoost), *Support Vector Machine Regression* (SVR), *k-Nearest Neighbors* (KNN), e *Artificial Neural Network* (ANN). Inicialmente os modelos realizam a previsão dos retornos das ações. As melhores ações então vão para o segundo estágio, onde o modelo *Mean-VaR (value-at-risk* realizará a seleção. Foi possível constatar que o modelo híbrido formado com o algoritmo AdaBoost teve desempenho superior aos outros (BEHERA et al., 2023).

Embora essas abordagens que utilizam inteligência artificial sejam treinadas com dados do mercado, elas não possuem interações diretas com o mercado. Assim, elas não tem capacidade de adaptação a cenários mais extremos, como a queda acentuada que a bolsa de valores sofreu com a pandemia de Covid-19. Em busca de solucionar o problema de adaptação do modelo, começou a se utilizar Aprendizado por Reforço na otimização de portfólio financeiro (DURALL, 2022).

Um estudo foi realizado sobre a aplicação de Aprendizado por Reforço para solucionar aplicações financeiras. Esse estudo teve como principal contribuição a elucidação da ligação entre problemas de otimização dinâmicos e Aprendizado por Reforço (KOLM; RITTER, 2019). Foi utilizada a técnica de Aprendizado por Reforço Profundo com algoritmos para ambiente contínuo e *Deep Deterministic Policy Gradient* (DDPG), *Proximal Policy Optimization* (PPO) e *Policy Gradient* (PG) para o gerenciamento de portfólio. O desempenho dos algoritmos foi apresentado sob diferentes configurações, seja com diferentes taxas de aprendizado, funções objetivos ou combinação de *features*. O objetivo com essa exploração foi ter mais clareza em relação à otimização dos parâmetros. Os dados utilizados foram do mercado chinês. Além disso, foi proposto o método *Adversarial Training*, mostrando que foi possível melhorar significativamente a eficiência. Assim, o modelo proposto teve melhor desempenho em relação ao modelo comparado (LIANG et al., 2018).

Uma estratégia de *ensemble* com uso de Aprendizado por Reforço Profundo foi desenvolvida para aprender uma estratégia para realizar o *trade* de ações de forma a maximizar o retorno do investimento. A estratégia *ensemble* foi obtida com o uso de três algoritmos, PPO, *Advantage Actor Critic* (A2C) e DDPG, o modelo resultante herda e integra as melhores características dos três algoritmos, gerando um robusto ajuste para diferentes situações do mercado. O desempenho do agente de trade com diferentes algoritmos de Aprendizado por Reforço é avaliado e comparado com o índice Dow Jones e a estratégia tradicional MV. A estratégia *ensemble* obteve um desempenho melhor que os modelos comparados em termos de retorno ajustado pelo risco medido pelo *sharpe ratio* (YANG et al., 2021).

Foi feito um comparativo entre algoritmos de Aprendizado por Reforço profundo e abordagens tradicionais para a otimização dos pesos de uma carteira de ações, sendo compostas por oito ações do mercado de financeiro dos Estados Unidos (DURALL, 2022). São utilizados cinco algoritmos em Aprendizado por Reforço profundo, PPO, A2C, DDPG, *Soft Actor Critic* (SAC) e *Twin-Delayed Deep Deterministic Policy Gra-*

dient (TD3); e quatro algoritmos clássicos, entre eles *Min-Variance*, *Min-Volatility*, *Risk Parity* e *Equal Weights*. O período utilizado para o treinamento e teste do modelo vai de 2010 a 2017. A entrada dos algoritmos de Aprendizado por Reforço consiste em um vetor com o preço diário das oito ações selecionadas. A saída retornada pelo modelo consiste no peso das oito ações que fazem parte do portfólio. Portanto, o modelo recebe os preços diários das ações e retorna os pesos para otimizar o portfólio. O retorno é calculado a partir de uma normalização: pega-se o valor atual da carteira e divide pelo valor inicial da carteira. Caso o resultado da divisão seja maior que um, indica que o portfólio teve valorização, caso contrário foi desvalorizado. Por fim, o trabalho realiza uma comparação entre os modelos de Aprendizado por Reforço e modelos tradicionais.

3.2 Otimização do Tempo de Execução e Consumo Energético

Foi proposto um método que utiliza a aprendizagem por reforço federado concorrente para o problema da afetação de alocação de recursos na computação de borda. A adição de concorrência na abordagem de tomada de decisões trouxe benefícios em escala global na alocação de recursos. Os resultados mostraram uma melhoria na velocidade e na utilização de recursos. (TIANQING et al., 2021). A computação em nuvem requer provisionamento e desprovisionamento automatizados, com base na demanda. Algoritmos são usados para definir instâncias de acordo com as solicitações recebidas. Definir o número de pedidos que são processados em paralelo é uma tarefa desafiadora. A aplicação de aprendizagem por reforço para encontrar a melhor configuração foi investigada, e os resultados mostram um aumento de desempenho quando se utiliza este algoritmo (SCHULER; JAMIL; KÜHL, 2021). Um estudo sobre o uso de técnicas de aprendizagem por reforço aplicadas ao escalonamento dinâmico de tarefas foi realizado, seguida por uma comparação dessas técnicas (SHYALIKA; SILVA; KARUNANANDA, 2020).

3.3 Discussão

Os trabalhos utilizam diversos algoritmos para maximizar o retorno obtido no mercado de ações. Os resultados dos modelos são comparados com modelos tradicionais e entre os próprios modelos, porém não utilizam nenhum *benchmark* para realizar a avaliação dos trabalhos. Isso acaba dificultando mensurar o desempenho dos modelos em

outros contextos. Assim não é possível comparar o desempenho com aplicações da renda variável, além de não ser possível comparar os retornos promovidos pelos algoritmos com uma carteira de ações feita por um banco ou corretora de investimentos.

Além disso, o trabalho de Durall (DURALL, 2022) faz o uso exclusivo de preços diários das ações pode fazer com que o modelo seja bem sensível às grandes oscilações do mercado, muitas vezes fazendo com que o modelo tenha seu desempenho reduzido. Ademais, temos apenas o preço das ações como *feature* para o modelo, tornando-o mais simples, porém com menos informações sobre as ações, podendo dificultar o processo de aprendizado. Por esses motivos, este trabalho se propõe em estender o trabalho (DURALL, 2022) e promover a melhoria do desempenho, além de também cobrir o cenário do mercado de ações do Brasil.

Ao contrário destes trabalhos que aplicam a aprendizagem por reforço na computação concorrente de tarefas, o nosso trabalho propõe utilizar aprendizagem por reforço de uma forma concorrente aplicada à bolsa de valores. Assim se busca otimizar o tempo de treinamento dos algoritmos, reduzindo o tempo global de treinamento e o consumo energético.

4 METODOLOGIA

Neste capítulo é apresentada a metodologia utilizada para a realização do trabalho e o protocolo experimental adotado para a execução dos experimentos.

4.1 Dados

Foram utilizados como entrada o preço de fechamento ajustado e o volume de negociações das ações. A extração dos dados foi feita por meio da biblioteca yfinance[1], no período de 2010 a 2017. O trabalho cobre dois cenários diferentes: uma carteira com ações do mercado dos Estados Unidos e outra com ações do mercado do Brasil. A Tabela 4.1 contém os ativos selecionados para as carteiras de ambos os cenários.

Tabela 4.1 – Ativos da carteiras de ações do Brasil e Estados Unidos

Carteira mercado do Brasil	Carteira mercado dos Estados Unidos
Ambev (ABEV3)	3M Company (MMM)
B3 (B3SA3)	Apple (AAPL)
Eletrobras (ELET3)	General Electric (GE)
Itaú Unibanco (ITUB4)	JPMorgan Chase & Co. (JPM)
Petrobras (PETR4)	Microsoft Corporation (MSFT)
Localiza (RENT3)	Nike Inc. (NKE)
Vale S.A. (VALE3)	NVIDIA Corporations (NVDA)
Weg S.A. (WEGE3)	Vodafone shares (VOD)

Cada cenário possui uma carteira de ações com oito ativos, a escolha dos ativos foi feita de forma a selecionar empresas de segmentos diferentes, buscando montar uma carteira diversificada. Este trabalho pode ser generalizado com ações de outros países ou até mesmo outras ações do Brasil e Estados Unidos.

Os dados passaram por uma etapa de pré-processamento. Inicialmente, foi verificado se existiam valores *not a number* (NAN), pois esses valores iriam atrapalhar o aprendizado dos modelos. Como não foi encontrado nenhum valor NAN, então prosseguiu-se para a segunda etapa. Esta etapa verificou se todas as ações tinham a mesma quantidade de registros, ou seja, a mesma quantidade de preços diários. Isso foi necessário para balancear os dados, garantindo assim que cada ativo tenha a mesma quantidade de informação. Também não foi encontrada nenhuma inconsistência, chegando ao fim a etapa de pré-processamento.

[1] https://github.com/yahoo-finance/yahoo-finance

4.2 Algoritmos de Aprendizado por Reforço utilizados

Foram selecionados cinco algoritmos de Aprendizado por Reforço, o objetivo era realizar o treinamento de cada algoritmo e analisar qual possui o melhor desempenho. Como cada algoritmo tem características diferentes, buscou-se entender quais dessas características influenciam no melhor resultado ao serem aplicados no contexto do mercado de ações. Os seguintes algoritmos foram selecionados:

- **Proximal Policy Optimization (PPO)** é um algoritmo *on-policy* que busca otimizar a política de uma maneira eficiente e estável, realiza múltiplas atualizações na política sem alterar significativamente o mapeamento do ambiente de observação para uma distribuição de probabilidade de ações que podem ser tomadas a cada iteração, o que ajuda a estabilizar o processo de aprendizado e evitar oscilações da política (SCHULMAN et al., 2017).
- **Advantage Actor-Critic (A2C)** é um algoritmo *on-policy* onde as redes de *actor* e *crític* são atualizadas simultaneamente, usando diferentes funções de perda para cada rede. A rede *actor* aprende a melhorar a política maximizando as recompensas esperadas, enquanto a rede *critc* aprende a estimar a função de valor do estado para ajudar o *actor* a tomar melhores decisões.
- **Deep Deterministic Policy Gradient (DDPG)** é um algoritmo *off-policy* utilizado em espaços de ação contínua usando uma política determinística. Além disso, o replay de repetição é utilizado para armazenar experiências passadas do treinamento. Ele possui duas redes, a *actor* e a *critic* (LILLICRAP et al., 2015).
- **Soft Actor-Critic (SAC)** é um algoritmo *off-policy* utilizado tanto no espaço de ações contínuo quanto discreto, buscando maximizar a recompensa cumulativa esperada ao mesmo tempo em que maximiza explicitamente a entropia da política, levando a uma melhor exploração em espaços de ação contínua de alta dimensão (HAARNOJA et al., 2018).
- **Twin Delayed Deep Deterministic Policy Gradient (TD3)** é um algoritmo *off-policy* e uma extensão do algoritmo DDPG, buscando solucionar algumas de suas limitações. Ele usa duas redes *critics* para reduzir o viés de superestimação e a atualização atrasada da rede *actor* para melhorar a estabilidade. TD3 aborda o problema de superestimação nas estimativas de valor Q usando o mínimo das redes críticas alvo.

Os algoritmos de aprendizagem escolhidos têm características diferentes no que diz respeito ao processo de otimização, tais como a sua abordagem à otimização de políticas, o tratamento de funções de valor, as estratégias de exploração e a utilização de buffers de repetição. Assim, ao aplicar cada um deles, é possível identificar quais características são melhores para otimizar os pesos de uma carteira de ações. Os algoritmos também possuem comportamentos distintos em relação ao uso de CPU e memória: os algoritmos *on-policy* (PPO, A2C) possuem menor uso de memória e uso moderado de CPU, pois envolvem muitas atualizações durante as épocas, exigindo um pouco mais de processamento computacional. Por outro lado, os algoritmos *off-policy* (DDPG, SAC e TD3) consomem mais CPU e usam mais memória do que os algoritmos com política para armazenar o buffer de repetição, uma vez que são necessárias mais atualizações e operações no buffer.

4.3 *Pipeline*

O *pipeline* que mostra o fluxo de funcionamento para o treinamento dos modelos está presente na Figura 4.1. É possível observar alguns módulos que são parâmetros para a realização do treinamento e outros auxiliares nesse processo. Os detalhes de cada módulo serão expostos nas próximas subseções, assim como o fluxo de funcionamento da arquitetura como um todo.

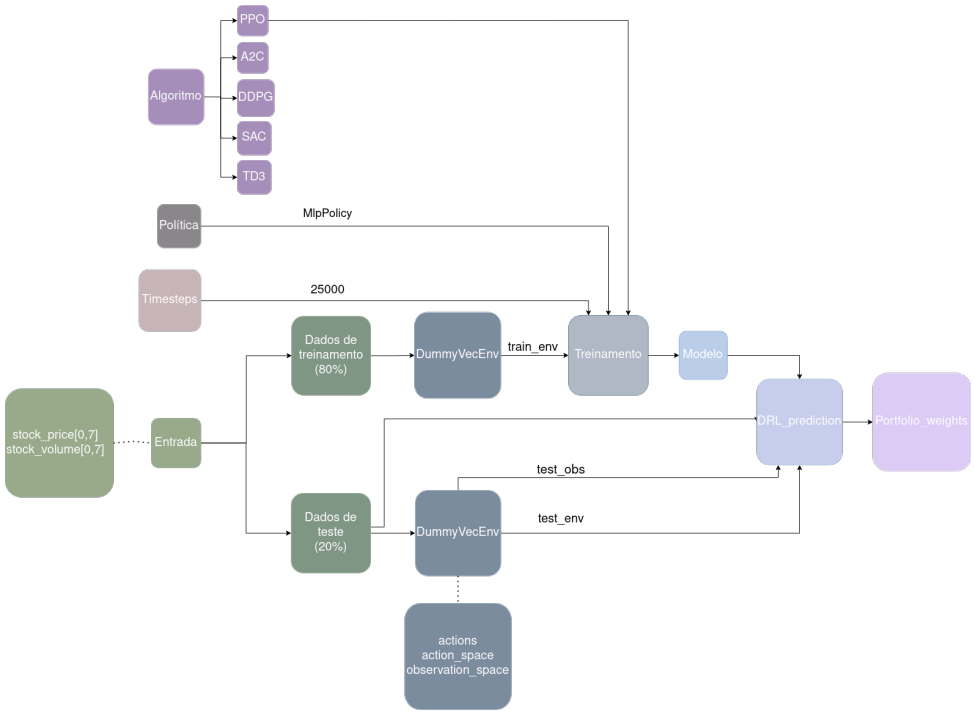


Figura 4.1 – Arquitetura utilizada para o treinamento dos modelos de Aprendizado por Reforço.

4.3.1 Entrada

O módulo de entrada possui os preços e os volumes financeiros de negociações diários das ações, conforme especificado na seção 4.1. Foi necessário dividir os dados para as etapas de treinamento e testes. Com isso foi definido 80% para treino e 20% para teste (DURALL, 2022) (SANDHYA; BANDI; HIMABINDU, 2022) (GHOSH et al., 2022). Para isso, a entrada utilizada para o treino são os primeiros 80% dos valores no conjunto de dados e a entrada de testes os últimos 20%.

O processo de manipulação dos dados foi feito por meio da biblioteca *Pandas* (TEAM, 2020). A Figura 4.2 exemplifica a forma como os dados estão organizados. Inicialmente temos o preço diário das oito ações, depois temos o volume financeiro de negociações dessas ações.

	AAPL	GE	JPM	MMM	MSFT	NKE	NVDA	VOD
Date								
2015-08-10	27.204350	138.956406	55.344597	116.610382	41.677547	52.945286	5.759183	24.251316
2015-08-11	25.788685	136.149734	54.814365	114.482307	40.867416	52.532230	5.730120	23.989449

Figura 4.2 – Exemplo dos dados utilizados para o treinamento e teste dos modelos.

4.3.2 Política e Timesteps

A política é um mapeamento do ambiente de observação para uma distribuição de probabilidade das ações que podem ser tomadas. *Timesteps* é uma forma de quantificar a sequencia de eventos que acontece durante o processo de treinamento.

Neste trabalho, optamos pelo timesteps no valor de 25000 e a política foi Mlp-Policy, já que o trabalho que optamos por estender utiliza esses valores de política e timesteps.

4.3.3 Algoritmo

O módulo Algoritmo seleciona qual algoritmo será selecionado para realizar o treinamento e a geração do modelo. Os algoritmos utilizados estão descritos na seção 4.2. Conforme se pode observar, o trabalho a ser estendido realiza o treinamento de cada

algoritmo de forma sequencial, mas esse trabalho busca configurações de execuções dos algoritmos de forma concorrente.

4.3.4 Treinamento

O modulo de treinamento recebe o algoritmo, política, ambiente de treinamento, *timesteps* e *seed*. Os treinamentos são realizados utilizando as implementações da biblioteca *stable-baseline3* (RAFFIN et al., 2021). O retorno desse módulo será o modelo treinado. É importante detalhar alguns pontos importantes do processo de treinamentos, temos:

- Estado: é o vetor com o estado corrente do processo de treinamento, ele é inicializado com o valor inicial investido na carteira e o preço e volume das oito ações, após cada step esses valores são atualizados. Dessa forma, esse vetor possui 17 posições, a posição 0 possui o valor atual investido, as posições 1 a 8 possuem os preços diários das 8 ações, por fim, as posições de 9 a 16 possuem o volume diário de negociações de cada ação. A Figura 4.3 ilustra o vetor estado.
- Ações: são os pesos das ações, por meio dos pesos o agente realiza interações com o ambiente e vai aprendendo a partir das experiências. A modificação dos pesos das ações se dá por meio do processo de compra e venda de ações, cada step pode ter venda ou compra de ações, o que vai resultar em uma alteração nos pesos das mesmas.
- Recompensa: é calculada a cada step, para o seu cálculo temos a subtração do valor final pelo valor inicial da carteira. O valor inicial é calculado por meio da somatório das multiplicações de preço e peso de cada ação, acrescito com o valor atual investido. O valor final é calculado da mesma forma, mas utiliza os novos pesos e o preço do dia seguinte. Dessa forma, a recompensa será positiva se houver valorização da carteira e negativa se tiver desvalorização.

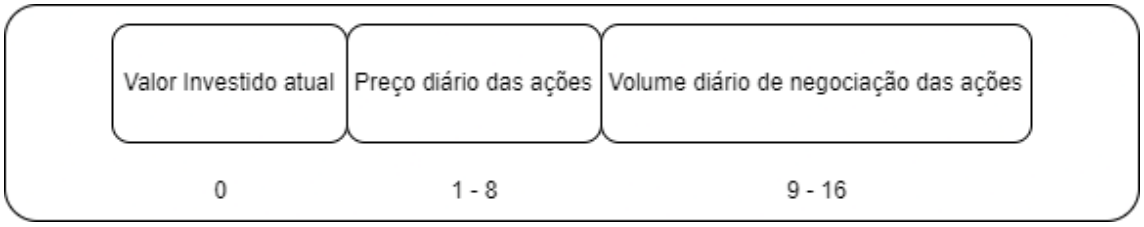


Figura 4.3 – Ilustração do vetor estado utilizado no processo de treinamento.

4.3.5 Outros módulos

O DummyVecEnv é uma classe da biblioteca *stable-baseline3* que cria um ambiente (*environment*) para treinamento de agentes de Aprendizado por Reforço, permitindo o treinamento de vários ambientes paralelamente para melhorar a eficiência do treinamento (RAFFIN et al., 2021). No entanto, ao contrário de outros ambientes vetorizados mais complexos, o *DummyVecEnv* não distribui as execuções em paralelo entre núcleos de CPU ou máquinas, sendo mais útil para fins de depuração, testes e experimentação. A partir desse módulo teremos as ações, observações e recompensas que serão utilizadas no módulo de treinamento.

Drl_prediction é o módulo responsável por realizar o processo de teste do modelo, tendo como entrada o modelo treinado, os dados de teste, o ambiente de teste e o espaço de observações de teste. Inicialmente, os pesos das ações são inicializados de forma aleatória. A partir desse momento se utiliza o método *predict* para realizar a predição com os dados de teste. Essa predição é feita com os dados diários e os pesos previstos são salvos em um vetor e retornados para o módulo *Portfolio_weights*. O módulo percorre todos os dados de teste e para cada dia retorna os pesos das ações.

O módulo Portfolio_weights possui a predição de todos os pesos feita pelo modelo para os dados de testes. A partir desse módulo é possível plotar os gráficos com o resultado do modelo. Para isso utilizamos o método pct_change() da biblioteca *pandas* e extraímos a percentagem de mudança do preço das ações. Assim multiplicamos essa percentagem pelo peso das ações e conseguimos calcular os ganhos ou perdas da carteira de ações diariamente. Essa informação final é utilizada no processo de construção dos gráficos.

É utilizado o retorno cumulativo, ou seja, caso o valor inicial investido seja de R$1.000,00, o retorno do modelo é relativo a esse valor em termos percentuais. Assim, caso exista valorização, o retorno terá valor superior a 1, caso tenha desvalorização, o retorno será inferior a 1. Por exemplo, caso o retorno do modelo cumulativo final seja de 1.1, isso significa que o capital evoluiu de R$1.000.00 para R$1.100.00.

Portanto, os retornos apresentados na seção 5 são relativos ao fim do período de teste, ou seja, o percentual de retorno da carteira no último dia dos dados de teste.

4.4 Ambiente de execução

Foram utilizados dois ambientes de execução distintos. O primeiro foi utilizado para os experimentos relativos a modificações para estender o trabalho de (DURALL, 2022), adicionando *benchmark* para melhor comparação com aplicações do mercado de renda variáveis e modificações com o objetivo de melhorar o desempenho dos modelos. Esse ambiente utilizou a nuvem e está descrito na subseção 4.4.1. O segundo está relacionado com a busca pela melhor configuração de execução para diminuir o tempo global de treinamento dos algoritmos e a redução do consumo energético, o qual foi descrito na subseção 4.4.2.

4.4.1 Modificações para melhorar o desempenho dos modelos

Os experimentos foram realizados por meio da plataforma *Google Colaboratory* (BISONG, 2019). A plataforma foi utilizada de forma gratuita, por isso em alguns momentos foi possível realizar os treinamentos com acesso a GPU gratuita e em outros utilizando apenas CPU.

4.4.2 Configurações de execução para minimizar o tempo e consumo energético

Os experimentos foram efetuados em uma arquitetura multicore com 2x Intel Xeon E5-2650 v3 Haswell e 20 núcleos físicos (40 com HyperThreading). Cada núcleo tem caches privadas L1 e L2 e partilha uma L3 e 128 GB de memória RAM. A frequência de funcionamento de cada núcleo varia entre 1,2GHz e 2,3GHz e, quando a funcionalidade Turbo está ativa, pode atingir 3.0GHz. Utilizamos o sistema operativo Linux Ubuntu com kernel v.5.6.0. Cada aplicação foi interpretada com Python 3.7.1 e executada com o regulador DVFS definido para desempenho pois é a escolha comum em servidores HPC.

Para obter o tempo de execução, usámos o pacote *Times*. Para isso, o tempo inicial foi obtido no início da execução de cada algoritmo, e quando a execução foi finalizada o tempo final foi calculado. Além disso, o consumo de energia foi extraído utilizando o pacote *CodeCarbon*. O pacote pega a potência do *hardware* num determinado momento e retorna o consumo total de energia em kWh. Com este valor, convertemos a energia de kWh para Joules.

4.5 Conjunto de Experimentos

Assim como na seção 4.4, esta seção está dividida em 2 subseções. Uma irá abordar o conjunto de experimentos na melhoria do desempenho dos modelos, enquanto a outra abordará os experimentos realizados na busca pela melhor configuração de execução, a fim de minimizar o tempo global de treinamento e o consumo energético.

4.5.1 Modificações para melhorar o desempenho dos modelos

A arquitetura inicial possuía apenas o preço diário das ações nos dados de entrada. Isso pode dificultar o aprendizado do agente e a identificação de padrões do mercado. Por isso foi adicionado o volume financeiro de negociações de cada ação nos dados de entrada. Além disso, realizar a predição do peso das ações utilizando apenas o preço das ações do dia corrente pode gerar modelos com grandes oscilações, já que o preço dos ativos pode variar muito de um dia para o outro. Em razão disso, foi acrescentado o preço médio das ações no modelo utilizando a média de 5, 10, 15, 20, 25 e 30 dias. Por fim, incluiu-se *benchmarks* para avaliar os modelos com desempenhos utilizado pelo mercado de renda variável. Foram usados os índices Ibovespa, Dow Jones, Nasdaq e S&P.

Com isso, foram feitos os seguintes experimentos para avaliar as mudanças promovidas na arquitetura:

- **Experimento I**: adicionando o volume financeiro de cada ação como *feature* para os modelos;
- **Experimento II**: utilizando a média dos preços de cada ação com diferentes períodos no vetor de estados com e sem a adição do volume;
- **Experimento III**: adicionando os *benchmarks* para uma melhor comparação dos resultados.

As modificações foram feitas de forma incremental. Dessa forma, para cada modificação foi realizado um experimento para avaliar o resultado da mudança, sendo que este resultado será comparado com os demais obtidos pelo trabalho base.

4.5.2 Configurações de execução para minimizar o tempo e consumo energético

O conjunto de experiências foi organizado em cinco cenários diferentes:

- **Cenário I**: todos os algoritmos de aprendizagem paralela são executados em ordem sequencial, um após o outro. Neste cenário, cada algoritmo pode utilizar todos os recursos de *hardware* disponíveis de acordo com a implementação paralela, sem os partilhar.
- **Cenário II**: são implementados dois algoritmos de aprendizagem de forma paralela para serem executados de forma concorrente. Deste modo, é possível observar os dois algoritmos mais adequados para partilhar os recursos de *hardware*, melhorando simultaneamente o desempenho geral e o consumo de energia.
- **Cenário III**: o mesmo do Cenário II, mas considerando três algoritmos de aprendizagem paralela.
- **Cenário IV**: o mesmo do Cenário II, mas considerando quatro algoritmos de aprendizagem paralela..
- **Cenário V**: todos os algoritmos de aprendizagem paralela são executados em simultâneo.

Para gerenciar a execução dos Cenários II, III, IV e V, contamos com o pacote Processing [2]. Trata-se de um pacote para a linguagem Python que suporta a geração de vários processos através do modelo de threading da biblioteca padrão. Assim, quando vários algoritmos de aprendizagem são implementados para execução, partilham os recursos de hardware disponíveis (como memórias cache, memória principal e núcleos, por exemplo).

[2] https://pypi.org/project/processing/

5 RESULTADOS E DISCUSSÃO

5.1 Otimização de Modelos de Aprendizado por Reforço

A discussão sobre os experimentos para melhorar o desempenho dos modelos necessita de resultados base. A partir disso será possível comparar se as mudanças propostas tiveram impacto positiva ou negativo no desempenho dos modelos. Por esse motivo, foram extraídos os resultados do trabalho (DURALL, 2022). O retorno obtido no final da etapa de testes de cada algoritmo no mercado de ações dos Estados Unidos está presente na Tabela 5.1. Esse retorno representa uma valorização caso seja maior que 1 e desvalorização caso seja menor do que 1. Por exemplo, se o retorno obtido pelo algoritmo PPO for de 1.2, então a carteira de ações teve valorização de 20%.

Tabela 5.1 – Retornos finais por algoritmo do artigo base

Algoritmo	Retorno final
PPO	1.25
A2C	1.32
DDPG	1.56
SAC	**1.76**
TD3	1.52

Além disso, esse trabalho propôs uma carteira de ações para o mercado financeiro do Brasil. Os resultados foram obtidos apenas com a adição das ações brasileiras nos dados de entrada, sem nenhuma outra modificação nos algoritmos. Estes resultados estão presentes na Tabela 5.2. Visto que o mercado de ações brasileiro possui mais volatilidade do que o americano, os retornos obtidos tendem a ser maiores, tanto no aspecto positivo quanto negativo, pois a amplitude de variação do mercado é maior.

Tabela 5.2 – Retornos finais por algoritmo com ações do Brasil sem modificações

Algoritmo	Retorno final
PPO	1.79
A2C	1.60
DDPG	1.59
SAC	1.62
TD3	**1.84**

As subseções seguintes irão tratar das melhorias propostas para a melhora do desempenho dos modelos, tanto no cenário do mercado americano quanto brasileiro.

5.1.1 Experimento I

A adição do volume financeiro de negociações das ações teve o objetivo de agregar mais informações sobre as ações, assim seria possível o modelo encontrar mais padrões e possivelmente correlacionar o volume com o preço das ações.

A Tabela 5.3 possui os resultados para a carteira de ações dos Estados Unidos com a adição do volume. É possível notar que os algoritmos PPO, A2C e SAC tiveram melhora no desempenho de *0.28*, *0.31* e *0.10*, respectivamente, em relação aos valores base Tabela 5.1. Como a política desses algoritmos muda de forma mais suave, este fato pode ter melhorado o processo de exploração e permitido que os algoritmos encontrassem pesos melhores para a carteira de ações. Enquanto os algoritmos DDPG e TD3 tiveram piora no desempenho de *0.25* e *0.32*, respectivamente, esses algoritmos possuem adição de ruído na exploração. Com isso, o acréscimo do volume pode ter causado aumento no ruído, levando a um desempenho pior do que a execução apenas com os preços da Tabela 5.1.

Tabela 5.3 – Retornos finais por algoritmo adicionando volume financeiro de negociações de cada ação americana

Algoritmo	Retorno final
PPO	1.53
A2C	1.63
DDPG	1.31
SAC	**1.86**
TD3	1.20

A Tabela 5.4 possui os retornos finais para a carteira de ações do Brasil com a adição do volume. Apenas o algoritmo SAC teve uma melhoria de 0.08, sendo que os outros algoritmos tiveram uma piora de desempenho. Isso deve ser explicado pelo fato do mercado brasileiro ter mais oscilações, uma vez que a adição do volume adicionou ainda mais variações, dificultando o processo de aprendizagem dos algoritmos. Esses resultados motivaram a execução do Experimento II, buscando reduzir essa alta amplitude.

5.1.2 Experimento II

Inicialmente foi utilizado apenas o preço médio das ações com diferentes períodos. A Tabela 5.5 possui os resultados das execuções para o mercado dos Estados Unidos. Os algoritmos que tiveram o resultado final reduzido no Experimento I apresentaram melhora

Tabela 5.4 – Retornos finais por algoritmo adicionando volume financeiro de negociações de cada ação brasileiras

Algoritmo	Retorno final
PPO	**1.75**
A2C	1.50
DDPG	1.49
SAC	1.70
TD3	1.50

significativa nessa execução, como foi o caso do DDPG e TD3. Inclusive o TD3 teve uma leve melhora em relação aos resultados do trabalho base, sendo superior em *0.04*. Os outros algoritmos tiveram uma piora significativa em relação ao Experimento I, embora o PPO e A2C ainda tenham obtido resultados melhores que o trabalho base em alguns períodos específicos. Esses resultados validam o fato de que o uso dos valores médios do preço das ações diminui o ruído e melhora o desempenho dos algoritmos prejudicados.

Além disso, foi feita a execução utilizando o mercado dos Estados Unidos com os preços médios das ações e com o volume financeiro, os resultados estão presentes na Figura 5.6. Os resultados obtidos são semelhantes ao Experimento I, onde os algoritmos que possuem ruído na exploração tem o desempenho reduzido no caso do DDPG e TD3. Porém, o TD3 teve um resultado com melhora expressiva para o período de 30 dias, sendo bem melhor do que o obtido no Experimento I. O fato de ter utilizado o preço médio das ações para diferentes períodos contribuiu para essa melhora.

Tabela 5.5 – Retornos finais por algoritmo com ações dos Estados Unidos utilizando preços médios com diferentes períodos.

Período	PPO	A2C	DDPG	SAC	TD3
1 dia	1.25	1.32	1.49	1.37	**1.56**
5 dias	1.33	1.12	1.49	1.44	**1.54**
10 dias	1.33	1.33	1.49	1.54	**1.56**
15 dias	1.33	1.11	1.48	1.30	**1.54**
20 dias	1.21	1.32	1.49	1.35	**1.56**
25 dias	1.40	1.12	1.48	1.29	**1.51**
30 dias	1.31	1.34	1.49	1.28	**1.51**

Foram executados os mesmos experimentos para o mercado do Brasil, a Tabela 5.7 possui os resultados utilizando apenas o preço médio das ações. Os retornos finais alcançados pelos algoritmos tiveram um resultado melhor que nos Experimentos I e que os valores base. O retorno obtido pelo TD3 no período de 15 dias teve uma melhora de *0.41* em relação ao Experimento I e *0.07* para os valores base. Isso mostra a importância do uso da média em mercados que sofrem mais variações.

Tabela 5.6 – Retornos finais por algoritmo com ações dos Estados Unidos utilizando preços médios com diferentes períodos e volume financeiro.

Período	PPO	A2C	DDPG	SAC	TD3
1 dia	1.53	1.63	1.31	**1.86**	1.20
5 dias	1.55	**1.65**	1.24	1.46	1.24
10 dias	1.25	1.35	1.23	**1.44**	1.24
15 dias	1.16	1.35	1.24	**1.46**	1.24
20 dias	1.26	1.35	1.24	**1.46**	1.29
25 dias	1.25	1.35	1.24	**1.46**	1.24
30 dias	1.31	1.27	1.24	1.44	**1.53**

Por fim, foi adicionado o volume financeiro em conjunto com os preços médios no mercado do Brasil, cujos resultados estão presentes na Tabela 5.8. O mesmo comportamento presente nos resultados do Experimento I se repetiram, sendo que aqueles algoritmos mais sensíveis a ruídos possuíram uma redução no retorno final obtido, enquanto os outros algoritmos alcançaram retornos finais melhores devido a nova *feature* que melhorou o processo de aprendizagem desses modelos.

Esse experimento permitiu notar as diferenças que o mercado de ações de países distintos possuem. Enquanto no mercado brasileiro o uso da média levou a resultados melhores, no mercado dos Estados Unidos o desempenho da maioria dos algoritgmos teve piora. Isso mostra a importância de realizar as otimizações com determinado mercado de ações como foco, melhorando os resultados obtidos.

Tabela 5.7 – Retornos finais por algoritmo com ações do Brasil utilizando preços médios com diferentes períodos.

Período	PPO	A2C	DDPG	SAC	TD3
1 dia	1.79	1.60	1.59	1.62	**1.84**
5 dias	1.65	1.59	1.57	1.39	**1.84**
10 dias	1.53	1.55	1.64	**1.89**	1.76
15 dias	1.52	1.57	1.74	1.57	**1.91**
20 dias	1.62	1.65	1.63	1.51	**1.74**
25 dias	1.58	1.55	**1.74**	1.44	1.71
30 dias	1.61	1.52	1.58	1.70	**1.73**

5.1.3 Experimento III

Esse experimento consiste em adicionar *benchmarks* dos mercados do Brasil e Estados Unidos para uma melhor comparação dos resultados obtidos pelo modelo. A Tabela 5.9 possui o Ibovespa, que basicamente seria o desempenho caso o investidor

Tabela 5.8 – Retornos finais por algoritmo com ações do Brasil utilizando preços médios com diferentes períodos e volume financeiro.

Período	PPO	A2C	DDPG	SAC	TD3
1 dia	**1.75**	1.50	1.49	1.70	1.50
5 dias	**1.76**	1.50	1.58	1.51	1.58
10 dias	1.80	1.60	1.48	**1.85**	1.55
15 dias	**1.76**	1.71	1.47	1.72	1.49
20 dias	**1.78**	1.60	1.45	1.69	1.54
25 dias	**1.78**	1.64	1.48	1.53	1.24
30 dias	1.81	**1.84**	1.49	1.63	1.53

tivesse as ações que formam esse índice, da mesma forma temos os *benchmarks* Dow Jones, Nasdaq, S&P. Dessa forma, temos os *benchmarks* que cada mercado utilizaria para analizar o desempenho de estratégias de investimento.

Tabela 5.9 – Retorno final de cada um dos índices da bolsa brasileira e das bolsas americanas.

Benchmark	Retorno final
Ibovespa	1.23
Dow Jones	1.14
Nasdaq	1.07
S&P	1.08

Observando o mercado dos Estados Unidos, o índice Dow Jones teve o melhor retorno final. O melhor resultado obtido pelos algoritmos foi do SAC no período de 1 dia, com 1.86, utilizando o preço médio e volume financeiro. Nesse caso, caso o investimento inicial fosse R$1.000,00, o retorno do investimento pelo benchmark Dow Jones seria R$1.230,00, enquanto o algoritmo SAC obteria R$1.860,00. Levando em consideração o mercado dos Estados Unidos, que possui variações menores, o retorno obtido pelo modelo foi muito bom e conseguiu superar os *benchmarks* utilizados pelo mercado americano.

No Brasil, existe apenas uma bolsa de valores, por isso o benchmark utilizado foi o Ibovespa. O melhor desempenho entre os algoritmos treinados com os dados das ações brasileiras foi do TD3, utilizando apenas os preços médios das ações no período de 15 dias. O retorno final foi de 1.91. O mercado brasileiro tende a ter mais oscilações, por isso o retorno estimado tende a ser maior que dos Estados Unidos, tanto positivamente quanto negativamente. Caso o insvestidor tivesse investido R$1.000,00 no índice Ibovespa, o retorno final do investimento seria R$1.230,00, enquanto a carteira de ações com os pesos gerados pelo algoritmo TD3 seria de R$1.910,00. O desempenho foi muito bom e superou o índice Ibovespa, utilizado para mensurar o quão bom são os investimentos em renda variável.

Desse modo, o algoritmo que teve melhor desempenho no mercado americano foi

o SAC e teve um retorno final de 1.86, enquanto no mercado brasileiro foi o TD3 que teve um retorno final de 1.91. Em relação ao artigo base, o SAC se mostrou superior aos resultados anteriores.

5.2 Otimização do Tempo de Execução e Consumo Energético

Esta seção discute os resultados de desempenho e energia da execução de cada cenário descrito na subseção 4.5.2. Para tal, começamos por avaliar o comportamento de cada algoritmo de aprendizagem na subseção 5.2.1. Em seguida, discutimos na subseção 5.2.2 os resultados da utilização da melhor combinação de execução dos algoritmos de Aprendizado por Reforço de forma concorrente.

5.2.1 Desempenho de cada algoritmo de Aprendizado por Reforço

Iniciamos a discussão com os resultados do Cenário I, onde todos os algoritmos paralelos são executados um após o outro, desse modo os algoritmos são executados de forma sequencial. Assim, a Tabela 5.10 mostra o tempo de execução de cada algoritmo, dado em segundos. Como observado, o PPO é o algoritmo com o menor tempo de execução, enquanto o TD3 demorou mais tempo a executar entre os cinco algoritmos. O comportamento de cada algoritmo de aprendizagem é explicado por diferentes razões, como descrito em cada um dos itens a seguir:

Tabela 5.10 – Resultados do desempenho de cada algoritmo no Cenário I

Algoritmo	**Tempo de Execução**
PPO	420s
A2C	480s
DDPG	8640s
SAC	13500s
TD3	10560s
Tempo de treinamento total	33600s

- Complexidade do algoritmo: Cada algoritmo tem as suas próprias estratégias de otimização subjacentes, arquiteturas de rede e procedimentos de atualizações. Por exemplo, o TD3 envolve redes Q gémeas para uma melhor estabilidade, suavização da política de objectivos e atualizações atrasadas, o que o torna computacionalmente mais dispendioso quando comparado com os outros algoritmos de Aprendi-

zado por Reforço.

- Eficiência da Amostra: Os algoritmos que são mais eficientes em termos de amostragem podem necessitar de menos interacções com o ambiente para obter um bom desempenho. O PPO é conhecido por ser relativamente eficiente em termos de amostras, enquanto o TD3 requer mais amostras para obter resultados semelhantes, o que leva a tempos de execução mais longos.
- Frequência de Atualização: Os algoritmos podem diferir na frequência com que atualizam as suas funções de política ou de valor. O A2C atualiza a política e as funções de valor de forma síncrona, o que pode levar a atualizações mais frequentes do que os métodos assíncronos. O TD3, por outro lado, efetua várias atualizações para cada interação com o ambiente devido a atualizações atrasadas da rede Q, o que contribui para um tempo de execução mais elevado.
- Exploitation: Têm impacto no número de interacções ambientais necessárias para aprender uma política eficaz. O TD3 e o DDPG são algoritmos que requerem mais *exploitation* utilizando a suavização da política de objetivos, o que leva a uma convergência inicial mais lenta em comparação com o PPO e o A2C, que têm uma estratégia de *exploitation* mais determinista. **Arquitetura da rede:** A complexidade das arquiteturas de redes neurais utilizadas para as funções de política e de valor pode ter impacto no tempo de execução. Assim, as arquiteturas mais complexas requerem mais cálculos e demoram mais tempo para atualizar. O TD3 utiliza normalmente uma rede mais complexa do que a PPO, o que leva a uma grande diferença no tempo de execução. Além disso, o DDPG e o SAC também possuem arquiteturas mais complexas que o PPO e A2C.
- Ruído na Exploração: Os algoritmos de aprendizagem por reforço que utilizam ruído de exploração, como o DDPG e o SAC, necessitam normalmente de interações adicionais com o ambiente para aprender uma política eficaz, o que contribui para o tempo de execução. Além disso, a utilização de ruído estocástico durante o treino no TD3 introduz aleatoriedade que afeta a velocidade de convergência e o tempo de execução.
- Redes Alvo e Atualizações Atrasadas: Os algoritmos de aprendizagem como o DDPG e o TD3 utilizam redes alvo e atualizações atrasadas, o que leva a uma convergência mais lenta do que outros métodos apresentados na Tabela 5.10.

Para a avaliação realizada nas subseções seguintes, foram considerados os resulta-

dos deste cenário como o *baseline*, uma vez que esta é a forma padrão como os algoritmos de aprendizagem por reforço paralelo são executados em arquiteturas *multicore*. Assim, dado que todo o processo de aprendizagem considera a execução dos cinco algoritmos, o tempo total para treinar e gerar os modelos de previsão é a soma do tempo de cada algoritmo, resultando em 33600s (cerca de 9h e 20 minutos).

5.2.2 Otimização da execução de algoritmos de aprendizado por Reforço através da execução concorrente

Esta subseção discute as vantagens de utilizar a execução concorrente de algoritmos de Aprendizado por Reforço para reduzir o tempo total de treinamento. Para o efeito, as Tabelas 5.11, 5.12 e 5.13 apresentam o tempo de execução de todas as combinações de execução dos algoritmos. A coluna sequencial representa o tempo de execução dos algoritmos um após o outro, enquanto a coluna concorrente denota o tempo de execução dos algoritmos em concorrência. Além disso, a coluna *speedup* destaca as melhorias de desempenho da execução concorrente em relação à sequencial. Por conseguinte, quanto mais elevado for este valor, maior a redução do tempo de execução. Em seguida, será discutido cada cenário separadamente, juntamente com a melhor solução encontrada através de uma busca exaustiva que cobriu todas as combinações possíveis de execução dos algoritmos de Aprendizado por Reforço.

5.2.2.1 Cenário II

Os resultados para este cenário são apresentados na Tabela 5.11. A primeira observação é que a execução concorrente traz benefícios de desempenho em todas as combinações de algoritmos de Aprendizado por Reforço. No melhor resultado, a execução concorrente dos algoritmos PPO e A2C otimiza o desempenho em *2,14*. Estes dois algoritmos utilizam arquiteturas de redes neurais semelhantes para as suas aproximações da política e de função de valor. Assim, quando um algoritmo atualiza a sua política, o outro atualiza a função de valor, reduzindo o tempo de inatividade global do sistema e conduzindo a uma melhor eficiência computacional. No entanto, há situações em que o desempenho simultâneo é semelhante ao da execução sequencial. É possível observar essa situação para a combinação A2C e SAC, com um speedup de apenas 1,12. Esse comportamento ocorre porque ambos os algoritmos são mais intensivos em memória. Portanto,

a competição por esse recurso compartilhado para armazenar experiências passadas para treinamento limitou as melhorias de desempenho.

Tabela 5.11 – Resultados do desempenho para cada combinação de algortimos no Cenário II

Combinação de Algoritmos	Sequencial	Concorrente	Speedup
PPO, A2C	900s	420s	**2.14**
PPO, DDPG	9060s	8160s	1.15
PPO, SAC	13920s	10980s	1.27
PPO, TD3	10980s	9120s	1.20
A2C, DDPG	9120s	8100s	1.12
A2C, SAC	13980s	9780s	1.43
A2C, TD3	11040s	8820s	1.25
DDPG, SAC	22140s	14040s	1.58
DDPG, TD3	19200s	12720s	1.51
SAC, TD3	24060s	13860	1.73

5.2.2.2 Cenário III

Quando três algoritmos são executados de forma concorrente, a competição pelos recursos partilhados aumenta, levando a mais penalizações no tempo total de execução da execução de forma concorrente do que no Cenário II, como mostra a Tabela 5.12. Neste cenário, a exploração da execução dos algoritmos A2C, SAC e TD3 em simultâneo pode proporcionar o maior aumento de desempenho em relação à execução sequencial (1,70 de aceleração). Estes três algoritmos apresentam características complementares de utilização da CPU e da memória, produzindo um melhor tempo de execução global. Por exemplo, o A2C é um algoritmo assíncrono em que cada agente funciona de forma independente e síncrona, o que o torna adequado para a execução de forma concorrente. Embora o SAC e o TD3 sejam mais dependentes da memória e partilhem o barramento fora do chip durante os acessos para atualizações de políticas e valores, a paralelização de cada algoritmo na forma como cada agente pode executar de forma assíncrona garante que não haverá penalizações devido a estas operações. Por outro lado, quando três algoritmos de memória intensiva são implantados para execução simultânea (DDPG, SAC e TD3), a competição por componentes de memória partilhada leva a uma penalização do desempenho dos algoritmos. Neste caso, a melhor estratégia é executar cada algoritmo um após o outro.

Tabela 5.12 – Resultados do desempenho para cada combinação de algortimos no Cenário III

Combinação de Algoritmos	Sequencial	Concorrente	Speedup
PPO, A2C, DDPG	9540s	8160s	1.16
PPO, A2C, SAC	14400s	9900s	1.45
PPO, A2C, TD3	11460s	9780s	1.17
PPO, DDPG, SAC	22560s	15600s	1.44
PPO, DDPG, TD3	19620s	14580s	1.34
A2C, DDPG, SAC	22620s	16200s	1.39
A2C, DDPG, TD3	19680s	12900s	1.52
A2C, SAC, TD3	24540s	14400s	**1.70**
DDPG, SAC, TD3	32700s	63360s	0.51

5.2.2.3 Cenário IV

A Tabela 5.13 apresenta o tempo de execução quando quatro algoritmos são executados de forma concorrente. Como se pode observar, em alguns casos é possível otimizar a execução de quatro algoritmos. Na situação mais significativa, o aumento de velocidade é de 1,65. Por outro lado, quando os algoritmos têm requisitos de CPU e memória semelhantes, não é possível melhorar o desempenho explorando a computação concorrente (por exemplo, PPO, DDPG, SAC e TD3). Estes algoritmos não alternam a computação dos agentes, deixando pouco espaço para explorar a utilização de recursos partilhados por *threads* de diferentes algoritmos.

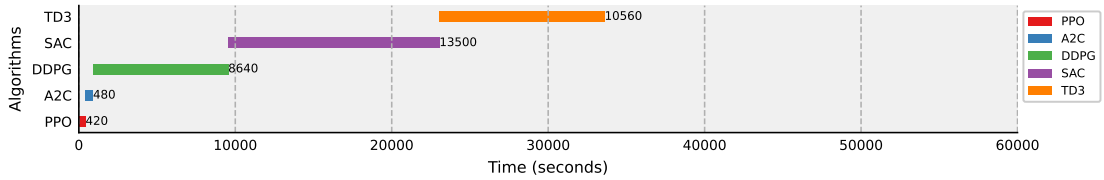
Tabela 5.13 – Resultados do desempenho para cada combinação de algortimos no Cenário IV

Combinação de Algoritmos	Sequencial	Concorrente	Speedup
PPO, A2C, DDPG, SAC	23040s	15060s	1.52
PPO, A2C, SAC, TD3	24960s	15120s	**1.65**
PPO, A2C, DDPG, TD3	20100s	13380s	1.50
PPO, DDPG, SAC, TD3	33120s	67920s	0.48
A2C, DDPG, SAC, TD3	33180s	65700s	0.50

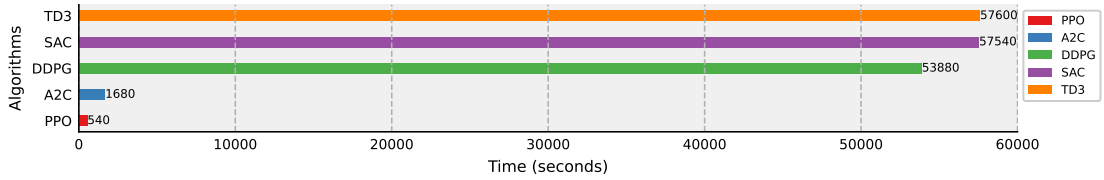
5.2.2.4 Cenário V

Diferente de todos os cenários anteriores, quando todos os algoritmos são executados concorrentemente, o tempo de execução de todo o treinamento aumenta por um fator de 1,71 em relação à execução seqüencial, como mostra a Figura 5.1. Esse comportamento ocorre porque cada algoritmo cria 40 *threads* (já que a arquitetura alvo possui 40 núcleos), totalizando 200 *threads*, e o excesso de *threads* apenas adiciona overhead ao gerenciamento de *threads* e competição por recursos compartilhados.Portanto, a execução simultânea de todos os algoritmos apresenta o pior resultado entre todos os cenários

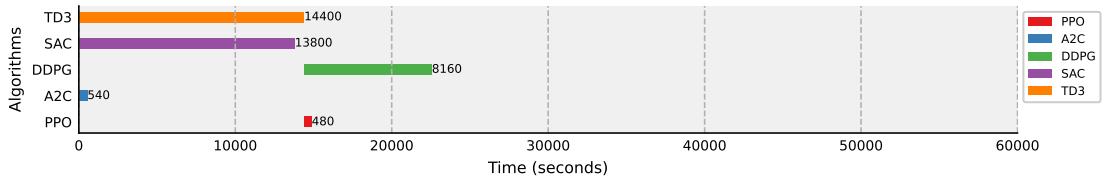
avaliados.



(a) Execução do Cenário I



(b) Execução do Cenário V



(c) Melhor combinação de execução dos algoritmos

Figura 5.1 – Comparação das execuções de 3 Cenários diferentes, no Cenário I os algoritmos são executados um após o outro, no cenário V todos os algoritmos são executados juntos, por fim, a melhor combinação executa inicialmente 3 algoritmos e depois 2 de forma concorrente.

5.2.2.5 Melhor solução

Para encontrar a melhor combinação de execução concorrente dos cinco algoritmos de aprendizagem por reforço, foi feita uma busca exaustiva que cobriu todas as combinações possíveis de execução dos algoritmos. Por isso, na Figura 5.1 temos o comportamento de execução do Cenário I na Figura 5.1(a), a forma padrão como os algoritmos de aprendizagem por reforço paralelos são executados, do Cenário V na Figura 5.1(b), os piores resultados, e a melhor combinação encontrada através da busca exaustiva na Figura 5.1(c). A Figura 5.2 apresenta o consumo de energia destes três cenários. Como se pode observar, o tempo total de execução é reduzido em 33% quando é aplicada a melhor combinação possível dos cinco algoritmos. Ao ser capaz de otimizar a utilização de recursos partilhados, também foi reduzido o consumo total de energia em 15% em comparação com a linha de base (Cenário I) e 63% em comparação com o Cenário V.

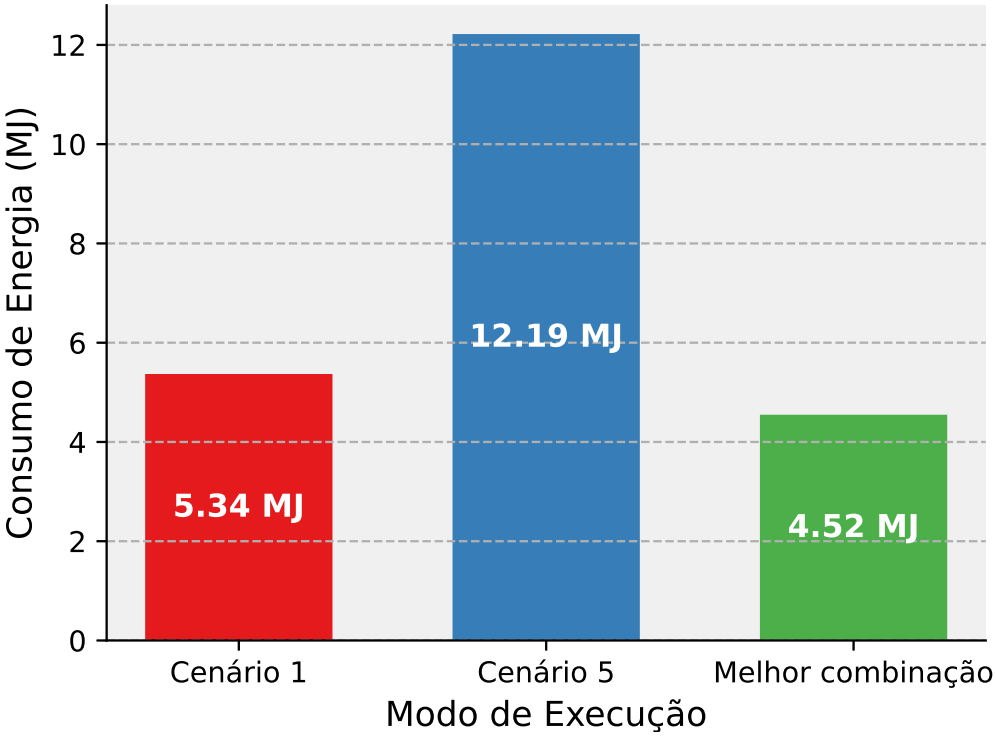


Figura 5.2 – Consumo energético de cada Cenário exibido na Figura 5.1.

6 CONCLUSÃO

Este trabalho buscou otimizar a aplicação de Aprendizado por Reforço na otimização de portfólio financeiro, especificamente na escolha da proporção investida em cada ação de uma carteira de investimento, buscando maximizar o retorno final. Foram propostas modificações com o objetivo de melhorar o desempenho e a eficiência energética dos algoritmos. Além disso, foram introduzidos benchmarks utilizados pelo mercado financeiro. Por meio deles, é possível menseurar o desempenho do modelo com estratégias já utilizadas pelo mercado. Portanto, o trabalho se propôs a melhorar o desempenho dos modelos, reduzir o tempo de treinamento e consumo energético para torna-los mais eficientes, e inserir o *benchmark* para melhorar a comparação com estratégias já utilizadas pelo mercado.

Observou-se que o comportamento do mercado de ações financeiro de cada país possui diferenças significativas. Isso levou a algumas melhorias serem positivas para um mercado e negativas para outro. O algoritmo com o melhor desempenho no mercado do Brasil teve um retorno acumulado final de 1,91, enquanto no mercado dos Estados Unidos foi de 1,86. O uso do preço médio das ações com diferentes períodos se mostrou uma boa alternativa para o mercado do Brasil, onde existe variações maiores do que mercados mais consolidados, como o dos Estados Unidos. Além disso, adicionar o volume financeiro de negociações se mostrou promissor para tornar os modelos melhores, porém a aplicação diretamente do volume sem utilizar a média dos volumes com diferentes períodos causou piora no desempenho. Em suma, percebe-se que cada mercado tem suas características e os modelos necessitam ser modificados com o intuito de melhorar o desempenho para um mercado em específico, assim os resultados obtidos tendem a serem melhores.

Além disso, foram investigados cenários distintos de computação concorrente para otimizar o desempenho e o consumo de energia do treinamento de modelos de Aprendizado por Reforço. Estes algoritmos desempenham um papel importante atualmente, uma vez que prevêem o peso das ações de uma carteira de ações. Através de execuções exaustivas, foi demonstrado que a seleção da combinação ideal de algoritmos a executar de forma concorrente conduz a melhorias significativas de desempenho e energia em relação à prática comum adotada pelos programadores de *software* e utilizadores finais: 33% de reduções no tempo de execução e 15% de poupanças de energia.

O trabalho produziu a escrita de um artigo para *XIII Brazilian Symposium on Computing Systems Engineering* (SBESC), com o título *Concurrent Computing for Accelera-*

ting Financial Machine Learning Model Training, ainda não foi recebido feedback sobre o artigo.

6.1 Limitações

O trabalho utilizou apenas o preço e volume financeiro de negociações de cada ação como entradas para o modelo, com isso ,acabou por não fazer o uso de indicadores que são utilizados para analisar as ações, como por exemplo o P/L (preço sobre o lucro), mencionado anteriormente. O uso dos indicadores poderia facilitar o aprendizado dos modelos, tornando mais fácil a identificação de padrões pelo modelo.

Além disso, os cenários executados para encontrar a melhor configuração de execução dos algoritmos de forma concorrente não utilizaram as melhorias propostas. Dessa forma, não foi possível mesclar as melhorias feitas para o desempenho com a melhor configuração de execução. Por fim, esses cenários foram executados apenas com a utilização de CPU, desse modo não foram exploradas outras arquiteturas heterogêneas.

6.2 Trabalhos Futuros

A utilização da média dos preços e volumes de cada uma das ações como os dados fornecidos para os modelos. Além disso, um estudo para a utilização dos indicadores fundamentalistas e técnicos das ações deveria ser feito, para tornar possível a utilização pelo modelo. Além disso, explorar arquiteturas heterogêneas para executar ainda mais algoritmos simultaneamente e melhorar a etapa de formação desses modelos.

REFERÊNCIAS

AGHA, G. **Actors: a model of concurrent computation in distributed systems**. [S.l.]: MIT press, 1986.

B3. **Consultas**. 2023. Available from Internet: <https://www.b3.com.br/pt_br/market-data-e-indices/servicos-de-dados/market-data/consultas/>.

BEHERA, J. et al. Prediction based mean-value-at-risk portfolio optimization using machine learning regression algorithms for multi-national stock markets. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 120, p. 105843, 2023.

BING, Z. et al. Meta-reinforcement learning in non-stationary and dynamic environments. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 45, n. 3, p. 3476–3491, 2022.

BISONG, E. Google colaboratory. In: ____. **Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners**. Berkeley, CA: Apress, 2019. p. 59–64. ISBN 978-1-4842-4470-8. Available from Internet: <https://doi.org/10.1007/978-1-4842-4470-8_7>.

BLACK, F.; LITTERMAN, R. Global portfolio optimization. **Financial analysts journal**, Taylor & Francis, v. 48, n. 5, p. 28–43, 1992.

CHAWEEWANCHON, A.; CHAYSIRI, R. Markowitz mean-variance portfolio optimization with predictive stock selection using machine learning. **International Journal of Financial Studies**, MDPI, v. 10, n. 3, p. 64, 2022.

CHEN, C.-H. et al. An effective approach for the diverse group stock portfolio optimization using grouping genetic algorithm. **IEEE Access**, IEEE, v. 7, p. 155871–155884, 2019.

CHEN, W. et al. Mean–variance portfolio optimization using machine learning-based stock price prediction. **Applied Soft Computing**, Elsevier, v. 100, p. 106943, 2021.

CHEN, W. et al. Mean–variance portfolio optimization using machine learning-based stock price prediction. **Applied Soft Computing**, Elsevier, v. 100, p. 106943, 2021.

CHEONG, D. et al. Using genetic algorithm to support clustering-based portfolio optimization by investor information. **Applied Soft Computing**, Elsevier, v. 61, p. 593–602, 2017.

CUTLER, D. M.; POTERBA, J. M.; SUMMERS, L. H. **What moves stock prices?** [S.l.]: National Bureau of Economic Research Cambridge, Mass., USA, 1988.

DURALL, R. **Asset Allocation: From Markowitz to Deep Reinforcement Learning**. 2022.

ERTENLICE, O.; KALAYCI, C. B. A survey of swarm intelligence for portfolio optimization: Algorithms and applications. **Swarm and evolutionary computation**, Elsevier, v. 39, p. 36–52, 2018.

GHOSH, S. et al. Options trading using artificial neural network and algorithmic trading. **International Journal of Next-Generation Computing**, v. 13, n. 5, 2022.

HAARNOJA, T. et al. Soft actor-critic algorithms and applications. **arXiv preprint arXiv:1812.05905**, 2018.

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: A survey. **Journal of artificial intelligence research**, v. 4, p. 237–285, 1996.

KOLM, P.; RITTER, G. Modern perspectives on reinforcement learning in finance. **SSRN Electronic Journal**, 01 2019.

LEI, X.; ZHANG, Z.; DONG, P. Dynamic path planning of unknown environment based on deep reinforcement learning. **Journal of Robotics**, Hindawi, v. 2018, 2018.

LIANG, Z. et al. Adversarial deep reinforcement learning in portfolio management. **arXiv: Portfolio Management**, 2018.

LILLICRAP, T. P. et al. Continuous control with deep reinforcement learning. **arXiv preprint arXiv:1509.02971**, 2015.

MA, Y.; HAN, R.; WANG, W. Portfolio optimization with return prediction using deep learning and machine learning. **Expert Systems with Applications**, Elsevier, v. 165, p. 113973, 2021.

MARKOWITZ, H. Modern portfolio theory. **Journal of Finance**, v. 7, n. 11, p. 77–91, 1952.

MARKOWITZ, H. M. **Portfolio selection: efficient diversification of investments**. [S.l.]: J. Wiley, 1967.

POLYDOROS, A. S.; NALPANTIDIS, L. Survey of model-based reinforcement learning: Applications on robotics. **Journal of Intelligent & Robotic Systems**, Springer, v. 86, n. 2, p. 153–173, 2017.

RAFFIN, A. et al. Stable-baselines3: Reliable reinforcement learning implementations. **Journal of Machine Learning Research**, v. 22, n. 268, p. 1–8, 2021. Available from Internet: <http://jmlr.org/papers/v22/20-1364.html>.

RONCALLI, T.; WEISANG, G. Risk parity portfolios with risk factors. **Quantitative Finance**, v. 16, n. 3, p. 377–388, 2016. Available from Internet: <https://EconPapers.repec.org/RePEc:taf:quantf:v:16:y:2016:i:3:p:377-388>.

SANDHYA, P.; BANDI, R.; HIMABINDU, D. D. Stock price prediction using recurrent neural network and lstm. In: IEEE. **2022 6th International Conference on Computing Methodologies and Communication (ICCMC)**. [S.l.], 2022. p. 1723–1728.

SATO, Y. Model-free reinforcement learning for financial portfolios: a brief survey. **arXiv preprint arXiv:1904.04973**, 2019.

SCHULER, L.; JAMIL, S.; KÜHL, N. Ai-based resource allocation: Reinforcement learning for adaptive auto-scaling in serverless environments. In: IEEE. **2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)**. [S.l.], 2021. p. 804–811.

SCHULMAN, J. et al. Proximal policy optimization algorithms. **arXiv preprint arXiv:1707.06347**, 2017.

SEN, J.; DUTTA, A.; MEHTAB, S. Stock portfolio optimization using a deep learning lstm model. In: IEEE. **2021 IEEE Mysore Sub Section International Conference (MysuruCon)**. [S.l.], 2021. p. 263–271.

SHARPE, W. F. The sharpe ratio. **Journal of Portfolio Management**, v. 21, p. 49–58, 1994.

SHARPE, W. F. The sharpe ratio. **Streetwise–the Best of the Journal of Portfolio Management**, Princeton University Press NJ, v. 3, p. 169–85, 1998.

SHYALIKA, C.; SILVA, T.; KARUNANANDA, A. Reinforcement learning in dynamic task scheduling: A review. **SN Computer Science**, Springer, v. 1, p. 1–17, 2020.

SOLEYMANI, F.; PAQUET, E. Deep graph convolutional reinforcement learning for financial portfolio management–deeppocket. **Expert Systems with Applications**, Elsevier, v. 182, p. 115127, 2021.

SOTO, T. Regression analysis. In: **Encyclopedia of Autism Spectrum Disorders**. [S.l.]: Springer, 2021. p. 3906–3906.

SUTTER, H.; LARUS, J. Software and the concurrency revolution: Leveraging the full power of multicore processors demands new tools and new thinking from the software industry. **Queue**, ACM New York, NY, USA, v. 3, n. 7, p. 54–62, 2005.

TEAM, T. pandas development. **pandas-dev/pandas: Pandas**. Zenodo, 2020. Available from Internet: <https://doi.org/10.5281/zenodo.3509134>.

TIANQING, Z. et al. Resource allocation in iot edge computing via concurrent federated reinforcement learning. **IEEE Internet of Things Journal**, IEEE, v. 9, n. 2, p. 1414–1426, 2021.

YANG, H. et al. Deep reinforcement learning for automated stock trading: An ensemble strategy. In: **Proceedings of the First ACM International Conference on AI in Finance**. New York, NY, USA: Association for Computing Machinery, 2021. (ICAIF '20). ISBN 9781450375849. Available from Internet: <https://doi.org/10.1145/3383455.3422540>.

ZHOU, Z.-H. **Machine learning**. [S.l.]: Springer Nature, 2021.

ZHU, H. et al. Particle swarm optimization (pso) for the constrained portfolio optimization problem. **Expert Systems with Applications**, Elsevier, v. 38, n. 8, p. 10161–10169, 2011.