Insper

Curso: Engenharia Mecatrônica

Relatório Final de Iniciação Científica

Lucca Oliveira Claro

Orientador: Prof. Raul Ikeda Gomes da Silva

Título: Estudo de Reinforcement Learning para o Mercado Financeiro

São Paulo

2024

Resumo

O mercado ações é um ambiente complexo e volátil, por isso, encontrar uma estratégia favorável para analisá-lo é um desafio. Neste artigo serão utilizadas estratégias de fusão entre algoritmos de *Reinforcement Learning* para uma melhor análise desse mercado, visando maximizar o desempenho. Os algoritmos *actor-critic,* que serão treinados para a estratégia de fusão, são *Advantage Actor Critic*, *Proximal Policy Optimization* e *Deep Deterministic Policy Gradient*. A partir de um treinamento prévio, as fusões buscam priorizar o algoritmo com melhor estabilidade de acordo com as condições do mercado, com o objetivo de deixar o modelo robusto. As fusões são feitas para alcançar um melhor resultado do que os algoritmos individuais. A comparação entre os modelos é feita pelo Índice de Sharpe. O treinamento do modelo será feito pelas ações do Dow Jones e os indicadores técnicos utilizados são: *Moving Average Convergence Divergence*, *Relative Strength Index*, *Commodity Channel Index* e *Average Directional Index*.

Palavras-chave: *Reinforcement Learning, Actor-Critic, Financial market*

Abstract

The financial market is a volatile environment, making it a challenge to find a favorable strategy to do trading. This article will use a strategy that merges three Reinforcement Learning algorithms for better market analysis, aiming to maximize returns. The actor-critic algorithms trained for this ensemble strategy are Advantage Actor Critic, Proximal Policy Optimization, and Deep Deterministic Policy Gradient. Through prior training, this fusion seeks to select the best algorithm to use depending on market conditions, with the goal of making the model robust enough to adjust to environmental conditions. This fusion proposal is made to achieve better results than the individual algorithms, comparing them based on the Sharpe ratio. The model will be trained using the stocks of the Dow Jones and the technical indicators used are: Moving Average Convergence Divergence, Relative Strength Index, Commodity Channel Index, and Average Directional Index.

Keywords: Reinforcement Learning, Actor-Critic, Financial market

Sumário

1 INTRODUÇÃO 5

2 METODOLOGIA..... 6

2.1 Reinforcement Learning 6

2.2 Modelagem do problema 8

2.3 Tratamento dos dados 10

2.4 Criação do ambiente..... 12

2.5 Algoritmos de RL 13

2.6 Estratégias de fusão 16

3 Resultados 18

4 Conclusão 21

5 REFERÊNCIAS 22

1 INTRODUÇÃO

O mercado financeiro é um ambiente complexo e dinâmico, onde a tomada de decisões rápidas e precisas é fundamental para o sucesso dos investidores[1]. Ao longo dos anos, têm surgido diversas abordagens e técnicas de automação computacionais em empresas e fundos de investimentos, com o objetivo de maximizar o desempenho do investimento e otimizar a alocação do capital. No entanto, o desafio se encontra em como considerar todos os fatores relevantes para a análise de um ambiente como esse[2,3].

Um modelo utilizado para resolver esse problema é o MDP (*Markov Decision Process*) com uma programação dinâmica para derivar uma estratégia ideal[4,5,6]. Porém é um modelo limitado, visto que não suporta um grande espaço de estados, devido a um limite de memória e computacional, já que quanto maior o espaço de estado, maior a complexidade da resolução do MDP e consequentemente maior as variáveis a serem armazenadas[7,8,9]. Uma estratégia que vem sendo promissora na alocação de carteiras é a utilização de ML (*Machine Learning*)[10], esta permite a análise de grandes volumes de dados em tempo real, a identificação de padrões não triviais e a tomada de decisões baseadas em algoritmos inteligentes sem a supervisão de um ser humano, uma técnica *self-learning*. Neste contexto, o objetivo é utilizar modelos de ML para analisar movimentos do mercado, identificar oportunidades de arbitragem e gerenciar riscos[10].

Uma técnica *self-learning* para lidar com a aleatoriedade do mercado financeiro e dados com ruídos é o RL (*Reinforcement Learning*)[11]. RL, um arcabouço biologicamente inspirado, que tem sua teoria profundamente enraizada no campo neurocientífico para o controle do comportamento[12,13,14]. É uma técnica de aprendizado de máquina que treina um agente a tomar decisões sequenciais em um ambiente incerto, através da interação com esse ambiente e do recebimento de recompensas ou punições[11]. O RL já possui sucesso nas áreas de navegação de robôs[15], jogos digitais[16] e controle de veículos[17].

Um desafio para utilizar RL no mercado financeiro está na representação do ambiente, já que os dados financeiros estão repletos de ruídos levando a um movimento constante nos dados e dificultado sua análise. Para mitigar isso é utilizado indicadores técnicos[18] que funcionam como filtros e geram novas variáveis para alimentar o modelo. Na procura pelo indicador ideal, acabou-se

criando uma grande quantidade deles, dificultando sua a escolha. Possuem métricas e interpretações diferentes, portanto, isoladamente, são genéricos para uma análise detalhada, por isso, geralmente, a interpretação do mercado é analisada por mais de um indicador[19].

Para lidar com esses desafios surge a técnica do Zhang et. al.[20], na qual é proposta uma fusão entre três algoritmos de RL. Estes atuam em um ambiente que fornece dados do mercado, ilustrado na Figura 1, com o intuito de encontrar uma estratégia, estável, de análise para um mercado complexo e dinâmico. Os três algoritmos *actor-critic*[21] utilizados são *Proximal Policy Optimization* (PPO)[22,23], *Advantage Actor Critic* (A2C)[24,25] e *Deep Deterministic Policy Gradient* (DDPG)[23,26,27].

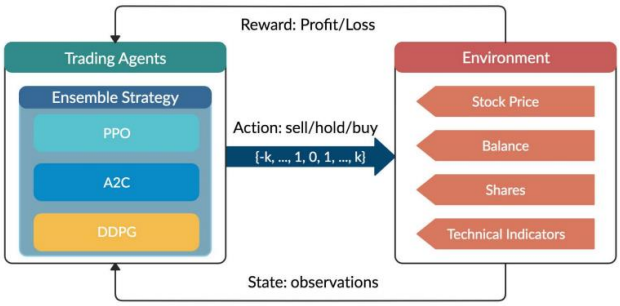

Figura 1. Visão geral da estratégia de trading criada no artigo[20]

Neste artigo será simulada a estratégia do estudo[20] com dados mais atuais do mercado americano. Com o objetivo de criar um modelo de RL que consiga automatizar uma estratégia de análise neste mercado, visando ter o melhor desempenho. Os dados serão tratados para que possam ser utilizados no modelo. O ambiente será simulado, definindo os indicadores técnicos, o espaço de ação, espaço de estados e a função recompensa.

2 METODOLOGIA

2.1 *Reinforcement Learning*

Reinforcement Learning é uma subárea de *Machine Learning*, que envolve ensinar a máquina como mapear situações para ações e como maximizar uma função recompensa. Assim como a categoria de *unsupervised learning*, não é dito quais ações deve-se tomar, mas sim deve-se descobrir quais ações levam a um

melhor desempenho. As situações enfrentadas por RL acabam por ser um ciclo, já que as ações tomadas pelo sistema em treinamento influenciam suas futuras ações, como também em sua recompensa imediata e futura.

A ideia geral dessa técnica é capturar os aspectos mais importantes de um problema, por meio de agentes que interagem com um ambiente. Essas interações geram um aprendizado para os agentes que tentam alcançar a melhor recompensa final possível. Esse agente deve sensoriar o ambiente e com isso tomar ações que o afetam, ele também deve ter metas relacionadas ao estado desse ambiente.

RL é diferente de *supervised learning*[28], um conjunto de técnicas de ML, que consiste em um aprendizado, em que as consequências de suas ações já são preestabelecidas por um agente externo. Portanto já há uma resposta correta para o sistema escolher, pois houve uma rotulagem prévia dos dados. Caso encontre um contexto que não está presente no treinamento, esse modelo acaba tentando generalizar ou extrapolar as respostas dadas. Em problemas de interação, é difícil rotular todas as situações que um agente de RL enfrentará, portanto em uma situação desconhecida, um agente precisa aprender com seus erros e acertos.

RL, também, não pode ser considerado *unsupervised learning*[28], que consiste em achar um padrão oculto em dados não rotulados[26]. Há discussões em que RL é colocado dentro dessa categoria, por não depender de exemplos rotulados, porém seu objetivo é maximizar uma função recompensa e não somente encontrar um padrão escondido, apesar dessa característica poder ajudar no objetivo final.

Um dos principais desafios que aparecem em RL e não em outros aprendizados (*unsupervised* e *supervised learning*) é o *trade-off* entre *exploration-exploitation*[30]. Para alcançar o objetivo de maximizar a recompensa, é necessário que o agente prefira realizar ações com resultados positivos que já tenha utilizado. No entanto, para ter descoberto essa ação, foi necessário realizar ações que não havia utilizado anteriormente. Portanto, o agente precisa realizar o *exploit* das ações que sabe para ter uma boa recompensa, mas também deve explorar novas ações para que tome melhores decisões no futuro[31]. Para isso o agente deve testar diversas ações, mesmo que algumas não alcancem sua meta e com isso favorecer aquelas que parecem ser as melhores. Para ter confiança em uma ação, deve ser feito vários testes em cenários diferentes.

2.2 Modelagem do problema

O desafio principal é criar uma estratégia automática de análise do mercado americano que utilize o RL. Os dados presentes no mercado de ações possuem componentes de aleatoriedade expressados através de ruídos no modelo, ou seja, difícil de serem analisados e previstos. Um meio de análise para resolver isso é via ML, em que a máquina tentará entender esses dados e extrair padrões. Algoritmos de RL são moldados para fazer isso de uma maneira inteligente, na qual são utilizados diversos agentes inseridos em um ambiente que contém esses dados e, a cada iteração, uma ação é realizada e um *feedback* é recebido. Alguns desses tipos de algoritmo são chamados de *actor-critic*[32,33].

Os agentes não terão nenhum conhecimento externo, ou seja, vão aprender com as ações que tomarão e os *feedbacks* que receberão. A cada iteração, os agentes poderão comprar, vender ou segurar ações de empresas contidas no portifólio. A partir da ação é calculado a função *reward* e com ela os agentes recebem o feedback dessa ação, caso o feedback for negativo é menos provável que essa ação seja realizada neste cenário em futuras iterações. O aprendizado no RL é complexo, já que os agentes passam por cenários e ações diferentes a cada iteração, é necessário, também, testar a mesma ação para diferentes situações.

Para auxiliar os agentes serão utilizados quatro indicadores técnicos, que por meio de funções matemáticas e estatísticas, analisam os dados do mercado e fornecem informações como de tendências e de padrões.

O MACD (*Moving Average Convergence Divergence*)[34], é utilizado para identificar tendências do mercado. Utiliza duas MME (Médias Móveis Exponenciais) para analisar a direção e a duração da tendência, calculadas pela Equação 1, em que $preço$ é o valor do *adjusted close* e N é o número de períodos. Para a primeira MME são utilizados 12 períodos, para a segunda são utilizados 26 períodos. Para calcular o MACD, é encontrado a diferença nos valores entre as duas MME, na Equação 1.

$$MME(N)_t = \left(\frac{preço_t - MME(N)_{t-1}}{N+1}\right) + MME(N)_{t-1}$$

$$MACD_t = MME(12)_t - MME(26)_t \tag{1}$$

O RSI (*Relative Strength Index*)[34] é um oscilador de momento que é utilizado para medir a velocidade e a mudança de direção dos preços. O indicador está entre de 0 a 100 e tenta indicar os pontos fortes e fracos do mercado. Para calculá-lo é utilizado a Equação 2, em que a $Médias\ dos\ ganhos$ e $Média\ das\ perdas$ são calculados por um período de 14 dias.

$$RSI = 100 - \left(\frac{100}{1 + \frac{Média\ dos\ ganhos}{Média\ das\ perdas}} \right) \tag{2}$$

O CCI (*Commodity Channel Index*)[35] é um oscilador de momento para identificar um mercado *overbought* ou *oversold*. Para calcular o indicador, primeiro é calculado o preço típico, e depois calculado o CCI pela Equação 3, em que $MM(PT)$ é a média móvel do preço típico, $\sigma(PT)$ é o desvio padrão do preço típico e 0.015 é uma constante para garantir que os resultados fiquem entre -100 e 100.

$$PT_t = \frac{adj\ close_t + adj\ low_t + adj\ high_t}{3}$$

$$CCI = \frac{1}{0.015} \cdot \frac{PT - MM(PT)}{\sigma(PT)} \tag{3}$$

O ADX (*Average Directional Index*)[36] é utilizado para quantificar a força de uma tendência, medindo a movimentação dos preços em uma só direção. Para calculá-lo, primeiro, é preciso encontrar o *True Range* que encontra o maior intervalo entre as opções, depois o index direcional positivo, o index direcional negativo e finalmente, pela Equação 4 é encontrado o valor de ADX.

$$TR_t = \max(adj\ high_t - adj\ low_t, adj\ high_t - adj\ close_{t-1}, adj\ low_t - adj\ close_{t-1})$$

$$+DI_t = \frac{MM(adj\ high_t - adj\ high_{t-1})}{MM(TR_t) \cdot 100}$$

$$-DI_t = \frac{MM(adj\ low_t - adj\ low_{t-1})}{MM(TR_t) \cdot 100}$$

$$ADX_t = MM\left(\frac{|+DI_t - -DI_t|}{|+DI_t + -DI_t| \cdot 100}\right) \tag{4}$$

2.3 Tratamento dos dados

O modelo será simulado com os parâmetros do artigo[20], portanto os dados utilizados foram os das 30 ações contidas no Dow Jones, um índice tradicional do mercado americano. Os dados das ações eram de 01/01/2009 até 05/08/2020, adquiridos no yahoo.financials, que contém *open*, *high*, *low*, *close*, *adjusted close* e volume diário de cada ação, visto na Figura 2. Durante a análise das 30 ações, a ação DOW, por ser uma empresa química recém adicionada ao índice, foi retirada por não possuir informações antes de 2018. No Dow Jones há rotações nas empresas do índice, portanto pode ser um problema em uma simulação mais próxima da realidade.

Para o cálculo dos indicadores técnicos foi necessário ajustar todas as informações, pois os dados não consideram alterações que ocorrem entre fechamento e a abertura no dia seguinte. Por isso foi calculado o *adjusted open*, *adjusted high*, *adjusted low*, ilustrado na Figura 3. Este cálculo começa com a divisão do *adjusted close* pelo *close* de cada dia, como resultado, encontrou-se uma constante que ao ser multiplicada pelas outras medidas, transformou-as em ajustadas. Pela biblioteca do pandas_ta, foi calculado os quatro indicadores para cada dia de cada ação, mostrado na Figura 4.

formatted_date	high	low	open	close	volume	adjclose
2008-11-13	64.870003	57.910000	59.230000	64.430000	11086500	40.980434
2008-11-14	65.949997	62.599998	63.220001	63.060001	6996600	40.109051
2008-11-17	62.930000	60.029999	62.330002	60.799999	6571200	38.671566
2008-11-18	62.419998	59.500000	60.230000	62.169998	9223700	39.542961
2008-11-19	62.360001	58.599998	61.830002	58.770000	8329600	37.683472
...	...	...	...	...	...	...
2020-05-04	148.559998	146.059998	148.539993	148.399994	2229800	128.574478
2020-05-05	150.000000	147.100006	149.889999	147.429993	2269200	127.734055
2020-05-06	148.169998	145.990005	147.789993	146.199997	2168600	126.668381
2020-05-07	147.850006	144.899994	147.699997	145.740005	2800800	126.269844
2020-05-08	149.179993	146.380005	147.080002	148.509995	2169700	128.669769

Figura 2. Dataframe com os dados não processados de uma ação com as informações do yahoo.financials

	formatted_date	high	low	open	close	volume
0	2009-01-02	38.081030	36.881982	36.901217	37.952789	5313900
1	2009-01-05	37.888655	37.260277	37.696295	37.510345	3768800
2	2009-01-06	38.350337	37.580893	37.664250	37.952789	4966200
3	2009-01-07	37.805308	36.920450	37.580888	37.234638	4598100
4	2009-01-08	37.645008	36.837093	36.984570	37.561653	3944900
...	...	...	...	...	...	...
2852	2020-05-04	128.713106	126.547094	128.695774	128.574478	2229800
2853	2020-05-05	129.960721	127.448153	129.865416	127.734055	2269200
2854	2020-05-06	128.375199	126.486443	128.045961	126.668381	2168600
2855	2020-05-07	128.097959	125.542054	127.967990	126.269844	2800800
2856	2020-05-08	129.250259	126.824336	127.430817	128.669769	2169700

Figura 3. Dataframe com os dados da figura 2 com open, high, low, close ajustados

MACD_12_26_9	MACDh_12_26_9	MACDs_12_26_9	RSI_14	CCI_14_0.015	ADX_14	DMP_14	DMN_14	return
-0.810478	0.399216	-1.209694	53.547362	146.951381	23.107340	22.663057	21.488453	0.028676
-0.681113	0.422865	-1.103978	50.926492	115.265576	21.256451	21.775195	20.646608	-0.011658
-0.536702	0.453821	-0.990523	53.383635	131.888445	20.142276	23.390885	19.641454	0.011795
-0.474732	0.412633	-0.887365	49.087330	61.751813	18.452738	21.974948	22.325151	-0.018922
-0.394683	0.394145	-0.788828	51.020362	66.754127	17.034931	20.908248	21.742281	0.008783
...	...	...	...	...	...	...	...	...
2.091571	0.394714	1.696856	52.141333	-10.174329	20.031209	29.897196	24.904101	-0.001346
1.823703	0.101478	1.722226	50.973614	-4.260636	19.488002	30.689527	23.905390	-0.006536
1.508042	-0.171347	1.679389	49.460993	-31.075168	18.747596	29.724529	24.754759	-0.008343
1.211751	-0.374110	1.585861	48.876854	-41.209748	17.825134	28.422123	25.289074	-0.003146
1.157252	-0.342887	1.500140	52.513786	-13.486829	17.224819	28.956193	23.970153	0.019006

Figura 4. Dataframe com uma amostra dos dados dos indicadores

2.4 Criação do ambiente

É preciso criar um ambiente que simula o mercado de ações, para que ocorra um treinamento e um teste dos modelos. Esse deve interagir com o agente e ensiná-lo, fornecendo os dados importantes, como histórico dos preços das ações e indicadores técnicos. O agente irá adquirir essas informações pelo ambiente e a partir disso realizar uma ação. Para implementar o ambiente e treinar o agente será utilizado *Gymnasium* [37,38,39].

A primeira etapa do ambiente é importar os dados tratados, a partir deles, é calculado a turbulência das ações por dia pela Equação 5. A turbulência, se acima do *threshold* de valor 99, indica uma mudança no cenário do mercado de ações, o que gera uma venda de todas as ações, para tentar minimizar possíveis perdas. Assim que a turbulência ficar menor que o *threshold*, volta a compra de ações. A Equação 5 é composta por um y_t que representa o preço no período de tempo t, um μ que representa a média dos preços e o $\sum$ representa a covariância dos preços.

$$turbulence_t = (y_t - \mu) \cdot \sum^{-1} (y_t - \mu)^T \in \mathbb{R} \tag{5}$$

Para o espaço de estados é utilizado um vetor com 177 dimensões, que pode ser considerado como contínuo, uma vez que possui infinitos diferentes estados no ambiente, onde o agente de cada algoritmo pode atuar. Esse espaço de estados possui sete grupos para representar um ambiente de mercado real. $[b_t, Ta_t, Tb_t, p_t, h_t, M_t, R_t, C_t, X_t]$. Cada componente é definido por:

b_t: Representa o capital disponível no período de tempo t.

Ta_t: Representa o valor da turbulência no período de tempo t.

Tb_t: Representa um booleano se a turbulência no período de tempo t é maior que o *threshold.*

p_t: Representa o *adjusted close* de cada ação no período de tempo t.

h_t: Representa a posição de cada ação no período de tempo t.

M_t: Representa o *Moving Average Convergence Divergence* de cada ação no período de tempo t.

R_t: Representa o *Relative Strength Index* de cada ação no período de tempo t.

C_t: Representa o *Commodity Channel* Index de cada ação no período de tempo t.

X_t: Representa o *Average Directional Index* de cada ação no período de tempo t.

Para o espaço de ações é utilizado um vetor com 29 dimensões, cada dimensão está relacionada as 29 ações, essas podem ser compostas por [-1,0,1] vender, segurar ou comprar. O espaço de ações mapeia decisões contínuas entre -1 e 1, devido às características dos algoritmos *actor-critic*, mas a ação a ser realizada pelos agentes é discreta, ou vende, ou compra, ou segura.

A função recompensa r é por onde os agentes recebem o *feedback* de uma ação executada, para que possam aprender. A Equação 6 representa a mudança do valor do portifólio quando uma ação a_t é executada em um estado s_t e recebida em um novo estado s_{t+1}. O objetivo das estratégias é maximizar essa função.

$$r(s_t, a_t, s_{t+1}) = b_{t+1} + asset_{t+1} - (b_t + asset_t) + carry \tag{6}$$

O $asset$ representa a $\sum h_t \cdot p_t$ para todas as ações, o $carry = b_t \cdot \left(\sqrt[252]{(1 + taxa\ livre\ de\ risco)} - 1\right)$, *taxa livre de risco* pode ser o valor do rendimento da *Treasury* americana ou até um índice de ação.

2.5 Algoritmos de RL

Para fazer o treinamento será utilizado três algoritmos *actor-critic* da biblioteca *stable_baseline3*:

Advantage Actor Critic (A2C)[24], que utiliza uma função gradiente para a *policy*. Que aumenta a probabilidade de uma ação ser realizada em um certo ambiente caso tenha um retorno bom e diminui a probabilidade de uma ação ser realizada em um certo ambiente caso tenha um retorno ruim. Esse método diminui a variância do gradiente *policy*, ao adicionar a *advantage* na função, isso, portanto, faz com que a avalição de uma ação não dependa da sua qualidade, mas sim do seu potencial.

A2C utiliza cópias do mesmo agente para atualizar os gradientes com diferentes dados, cada agente interage com o mesmo ambiente e no final é calculado o gradiente do respectivo agente. O algoritmo utiliza uma estratégia coordenada de passar a média dos gradientes para uma rede global, na qual atualiza a rede *actor* e *critic*, aumentando a diversidade do treinamento dos dados.

A função objetiva do A2C é:

$$\nabla J_\theta(\theta) = \mathbb{E}[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) A(s_t, a_t)] \tag{7}$$

Em que $\pi_\theta(a_t|s_t)$ é a rede da *policy*, $A(s_t, a_t)$ é a função *Advantage*, que pode ser escrito como:

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t) \tag{8}$$

Ou

$$A(s_t, a_t) = r(s_t, a_t, s_{t+1}) + \gamma V(s_{t+1}) - V(s_t) \tag{9}$$

Deep Deterministic Policy Gradient (DDPG) [26], que combina *frameworks* de *Q-learning* [40] e gradiente *policy*[41], e utiliza uma rede neural como função de aproximação. DDPG aprende diretamente com as observações do gradiente *policy*, seu funcionamento consiste em mapear o estado para ações que melhor funcionam no espaço de ação.

A cada iteração, o agente DDPG realiza uma ação a_t em um estado s_t, recebe uma recompensa de r_t e vai para o novo estado s_{t+1}. As transições

(s_t, a_t, s_{t+1}, r_t) são armazenadas em um *replay buffer* R. O q-valor y_i é atualizado pela Equação 10.

$$y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}\middle|\theta^{\mu'}, \theta^{Q'}\right)\right), i = 1, \dots, N. \tag{10}$$

A rede crítica é atualizada a parir da minimização da função de perda $L(\theta^Q)$, que é a diferença entre as saídas da rede crítica alvo Q' e a rede crítica Q.

$$L(\theta^Q) = \mathbb{E}_{s_t, a_t, r_t, s_{t+1} \sim buffer}\left[\left(y_i - Q(s_t, a_t|\theta^Q)\right)^2\right] \tag{11}$$

Proximal Policy Optimization (PPO) [22], tem como objetivo controlar a atualização do gradiente *policy* e garantir que a nova *policy* não seja muito diferente da anterior, gerando uma maior estabilidade para o modelo. PPO tenta simplificar o objetivo da *Trust Region Policy Optimization* (TRPO), introduzindo um recorte na função objetiva.

Supondo que a taxa de probabilidade entre a política antiga e nova é expressa na Equação 12.

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{12}$$

O recorte introduzido na função é:

$$J^{CLIP}(\theta) = \widehat{\mathbb{E}}_t\left[\min\left(r_t(\theta)\hat{\mathrm{A}}(s_t, a_t) clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{\mathrm{A}}(s_t, a_t)\right)\right] \tag{13}$$

Em que $r_t(\theta)\hat{\mathrm{A}}(s_t, a_t)$ é o objetivo do gradiente *policy* normal e $\hat{\mathrm{A}}(s_t, a_t)$ é a função *Advantage* estimada. A função $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)$ limita a taxa $r_t(\theta)$ para estar entre $[1-\epsilon, 1+\epsilon]$. A função objetivo do PPO fornece o mínimo da função *clipped* e normal. PPO dificulta uma mudança na política para fora do intervalo *clipped*. Portanto esse algoritmo aprimora a estabilidade da rede de treinamento da política, a partir de restrições das atualizações a cada iteração.

2.6 Estratégias de fusão

Os algoritmos individualmente, possuem suas próprias estratégias para fazer uma análise do mercado. Eles, individualmente, são arriscados, já que se houver um erro na leitura do mercado, haverá uma grande perda. Para reduzir a volatilidade foram simuladas três estratégias de fusão que utilizam os três algoritmos juntos.

A primeira estratégia a ser implementada é a do artigo[20], que consiste em selecionar de mês em mês o algoritmo mais adequado entre A2C, PPO e DDPG para realizar a análise do mercado. Como os dados das ações são disponibilizados em dias úteis, foi utilizado que 21 dias de dados corresponde a 1 mês. Os algoritmos a serem utilizados a cada período foram selecionados pelo Índice de Sharpe, calculado na Equação 14, em que $\overline{r_p}$ é o retorno do portifólio, r_f é a taxa livre de risco e σ_p é o desvio padrão do portifólio.

$$Sharpe\ ratio = \frac{\overline{r_p} - r_f}{\sigma_p} \tag{14}$$

No primeiro mês só haverá uma análise para descobrir qual é o melhor algoritmo desse período, o vencedor é o que será utilizado no mês seguinte. Paralelamente os três algoritmos, independente do que será utilizado, serão testados para definir qual será o algoritmo utilizado no próximo mês. O objetivo dessa fusão é diminuir a sensibilidade do modelo para tendências do mercado e maximizar o retorno sem ter um risco elevado. Essa estratégia pode ser observada na Tabela 1 e na Figura 5.

Tabela 1. Índice de Sharpe dos algoritmos

Período de Treinamento	A2C	PPO	DDPG	Algoritmo escolhido
01/01/2009 – 01/02/2009	0.126	0.431	**0.444**	DDPG
01/02/2009 – 01/03/2009	0.251	0.347	**0.402**	DDPG
01/03/2009 – 01/04/2009	0.400	**0.430**	0.170	PPO

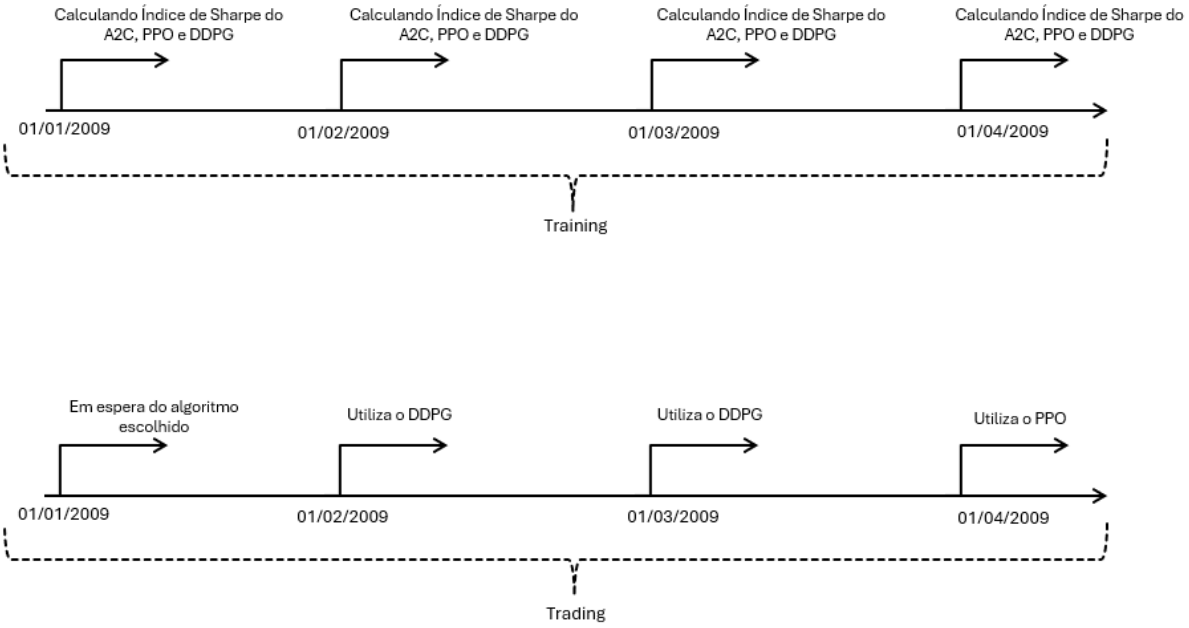

Figura 5. Linha do tempo da estratégia de fusão do artigo[20]

A segunda estratégia a ser implementada é semelhante com a do artigo[20], a diferença é que os três algoritmos serão utilizados. Ao invés de escolher um algoritmo para fazer a análise do mercado, é atribuído uma ponderação do montante inicial para cada algoritmo, uma ponderação que é proporcional ao valor do Índice de Sharpe após 21 dias. A primeira etapa da ponderação é realizar uma normalização visto na Equação 15, em que sr_t^i é o Índice de Sharpe para cada algoritmo em cada mês. Com a normalização de todos os Índice de Sharpe do mês, é feito a ponderação para cada algoritmo pela Equação 16.

$$srp_t^i = sr_t^i - \min(sr_t) + 1 \tag{15}$$

$$w_t^i = \frac{srp_t^i}{sum(srp_t)} \tag{16}$$

Isso possibilita a participação de todos os algoritmos na análise do mercado, porém com seus montantes proporcionais aos seus desempenhos no mês anterior. Esse novo modelo tem uma estratégia que considera os três algoritmos, independente de seus resultados, já que a técnica do artigo[20], indiretamente,

impõe que o algoritmo com o melhor Índice de Sharpe por um mês será o melhor para o próximo.

A terceira estratégia de fusão implementada tem o mesmo objetivo da anterior, fazer uma ponderação para a distribuição do capital para cada algoritmo utilizar. Para encontrar essa ponderação foi utilizado a técnica da regressão linear múltipla, na qual utiliza variáveis para prever o valor de outra variável. Para esse caso foram usados três variáveis de cada algoritmo: peso atribuído ao algoritmo no período selecionado; média do retorno até o período selecionado; desvio padrão do retorno até o período selecionado. Essas variáveis foram utilizadas para prever o valor do peso atribuído ao algoritmo no período seguinte, a cada cinco dias úteis o peso para cada algoritmo é recalculado. A regressão escolhida foi múltipla, já que não se pode considerar que as variáveis são independentes, podem estar relacionadas de alguma maneira, portanto a regressão precisa considerar as variáveis juntas.

Um problema enfrentado para implementar essa regressão foi a pequena quantidade de dados, que causou *overfitting* no treinamento, ou seja, uma dificuldade de realizar uma boa generalização dos dados. Para evitar isso foi utilizado um modelo Ridge que incorpora uma penalidade nos coeficientes do modelo. Para utilizar essa estratégia foi necessária uma divisão dos dados para um treinamento e depois um teste, portanto seus dados eram menores.

3. Resultados

Os modelos foram treinados com os dados das 29 ações do *Dow Jones* entre 1/1/2009 e 8/5/2020 e foram testados com dados entre 11/5/2020 e 28/3/2024. Os três algoritmos *actor-critic* e as três estratégias de fusão foram utilizados para fazer uma análise das 29 ações no período de teste. O patrimônio líquido de cada algoritmo pode ser analisado na Figura 6. A comparação entre os modelos será feita pelo Índice de Sharpe, que mede a relação entre o risco e o retorno do investimento. O Índice de Sharpe anualizado dos modelos pode ser visto na Tabela 2.

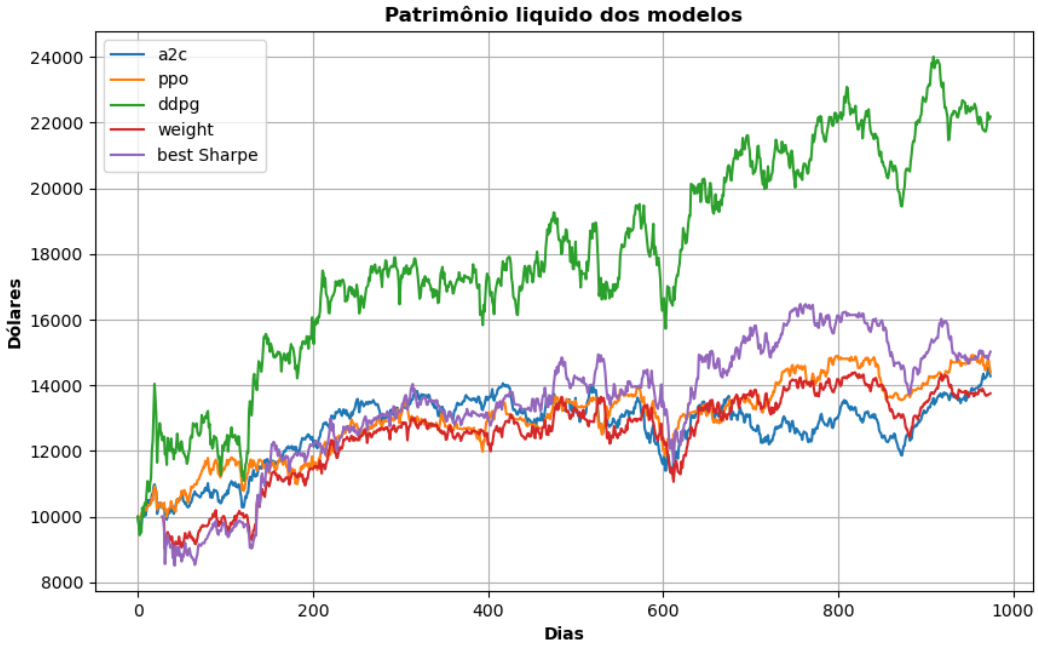

Figura 6. Gráfico do patrimônio líquido de todos os modelos

Pela Figura 6, o DDPG após os testes foi o que teve um patrimônio líquido maior que os outros modelos para essas ações nesse período de quatro anos. Apesar de sua alta performance nesse aspecto, há uma flutuação significante, ou seja, possui alguns períodos em que a queda do patrimônio é grande, o que aumenta o risco do modelo. Isso é reforçado na Tabela 2, diferentemente da Figura 6, o Índice de Sharpe do DDPG é próximo da primeira estratégia de fusão, o que mostra que apesar de ter tido um patrimônio líquido elevado, seu risco, comparado com os outros, é alto.

Tabela 2. Índice Sharpe anualizado de todos os modelos ao final do teste

	A2C	PPO	DDPG	Fusão artigo[20]	Fusão ponderação
Índice de Sharpe	0.263	0.363	0.454	0.422	0.335

Na Figura 7 é possível analisar as duas estratégias de fusão proposta, a do artigo[20] e a da ponderação pelo Índice de Sharpe. A primeira apresenta um maior

patrimônio líquido ao final do período analisado e com menos risco, mostrado na Tabela 2.

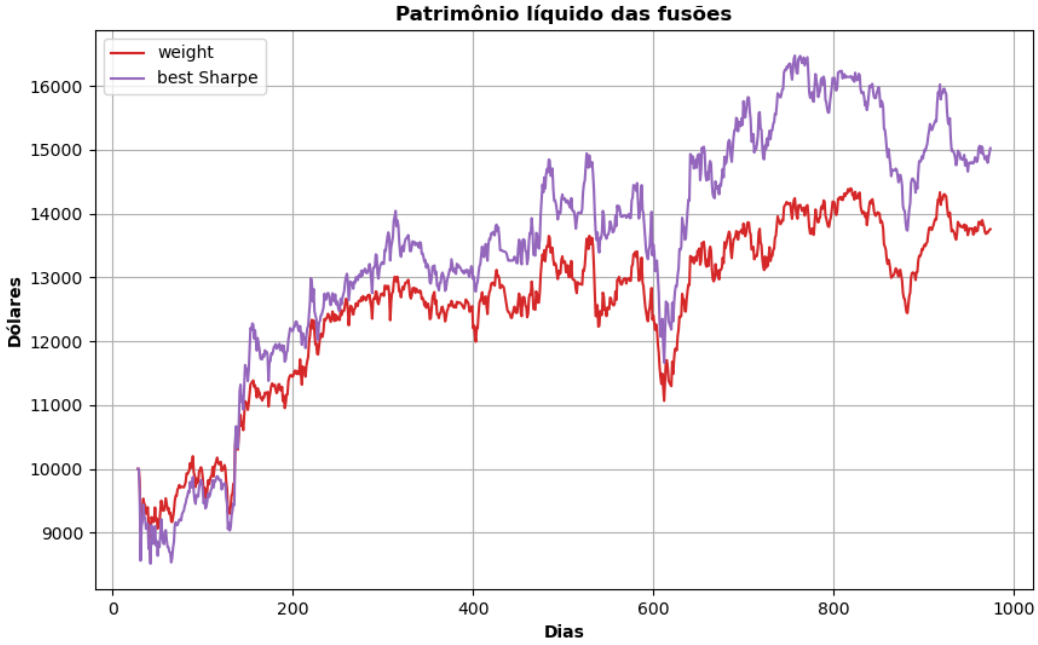

Figura 7. Gráfico do patrimônio líquido das fusões

Para comparar a fusão feita pela regressão Ridge, o período a ser analisado é diferente, já que dois anos de dados dos testes foram utilizados para treinar a regressão, deixando apenas dois anos de dados testados, enquanto os outros modelos tiveram quatro anos. As três fusões foram, então, ajustadas para que ficassem todos com o mesmo período, os últimos dois anos, para que possam ser comparados.

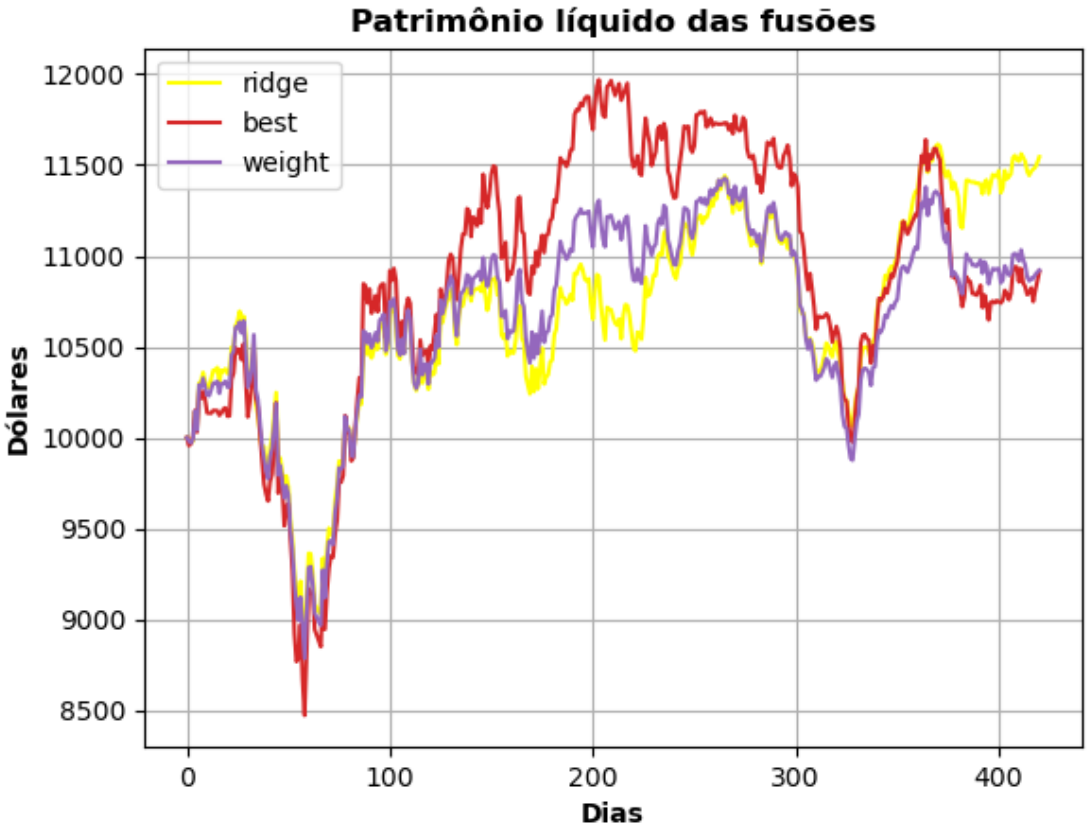

Figura 8. Patrimônio líquido das fusões com período ajustado

Analisando a Figura 8, a fusão que utiliza a regressão *ridge* possui o maior patrimônio líquido para esse período. Outro ponto a se analisar é a Tabela 3, que mostra que o Índice de Sharpe para esse período reduzido, o desta fusão também é o maior. Indicando que para esses dados e neste período, essa fusão teve um melhor desempenho e com pouco risco, comparado as outras.

Tabela 3. Índice de Sharpe das fusões para o período de teste reduzido

	Fusão artigo[20]	**Fusão ponderação**	**Fusão ponderação por ridge**
Índice de Sharpe	0.134	0.133	0.372

4. Conclusão

Neste artigo é explorado se o método de RL, por meio de algoritmos *actor-critic*, consegue simular uma estratégia para o mercado de ações. O objetivo foi de implementar três modelos de fusão de três algoritmos, para alcançar uma melhor

estabilidade comparado com os algoritmos atuando individualmente. A comparação foi feita pelo Índice de Sharpe. Após uma análise dos resultados, as fusões possuíam menos volatilidade, porém com um patrimônio líquido final menor, comparado aos algoritmos.

As próximas etapas desse artigo seriam alterar os dados para outros países, melhorar os algoritmos ML, aumentar tamanho de dados no teste, já que a fusão com regressão ficou com poucos dados para testar, utilizar outros indicadores técnicos e algoritmos para melhor análise e adicionar um recurso de continuar treinando os modelos durante os testes.

Referências

[1] Kendall, M. G., Hill, A. B. (1953). The analysis of Economic Time-Series-Part I: Prices. **Journal of the Royal Society of Statistics**, Series A, 116, 11-34

[2] Youngmin Kim, Wonbin Ahn, Kyong Joo Oh, and David Enke. 2017. “An intelligent hybrid trading system for discovering trading rules for the futures market using rough sets and genetic algorithms.” **Applied Soft Computing**, vol 55 (02 2017), 127–140.

[3] Yong Zhang and Xingyu Yang. 2016. Online portfolio selection strategy based on combining experts’ advice. **Computational Economics**, vol 50 (05 2016).

[4] Dimitri Bertsekas. 1995. Dynamic programming and optimal control. Vol. 1

[5] Ralph Neuneier. 1996. Optimal asset allocation using adaptive dynamic programming. **Conference on Neural Information Processing Systems**, 1995 (05 1996).

[6] Ralph Neuneier. 1997. Enhancing Q-learning for optimal asset allocation. **Conference on Neural Information Processing Systems** (NeurIPS), 1997.

[7] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press. Chapter 3.6 discusses the curse of dimensionality in MDPs.

[8] Bertsekas, D. P. (2005). Dynamic programming and optimal control. Athena Scientific. Chapter 3.6 discusses the curse of dimensionality in dynamic programming.

[9] Kochenderfer, M. J., & Wheeler, T. A. (2019). Algorithms for decision making. Cambridge University Press. Chapter 5.6 discusses the curse of dimensionality in MDPs.

[10] PRADO, López. Advances in Financial Machine Learning. USA: WILEY, 2018.

[11] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press, 1998.

[12] D. Prokhorov, G. Puskorius, and L. Feldkamp, “Dynamical neural networks for control,” in A Field Guide to Dynamical Recurrent Networks. New York, NY, USA: IEEE Press, 2001.

[13] D. Zhao and Y. Zhu, “MEC—A near-optimal online reinforcement learning algorithm for continuous deterministic systems,” IEEE Trans. Neural Netw. Learn. Syst., vol. 26, no. 2, pp. 346– 356, Feb. 2015.

[14] W. Schultz, P. Dayan, and P. R. Montague, “A neural substrate of prediction and reward,” Science, vol. 275, no. 5306, pp. 1593–1599, 1997.

[15] H. R. Beom and K. S. Cho, “A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning,” IEEE Trans. Syst., Man, Cybern., vol. 25, no. 3, pp. 464–477, Mar. 1995.

[16] V. Mnih et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[17] H. J. Kim, M. I. Jordan, S. Sastry, and A. Y. Ng, “Autonomous helicopter flight via reinforcement learning,” in Proc. Adv. Neural Inf. Process. Syst., 2003, pp. 799–806

[18] C. J. Neely, D. E. Rapach, J. Tu, and G. Zhou, “Forecasting the equity risk premium: The role of technical indicators,” Manage. Sci., vol. 60, no. 7, pp. 1772–1791, 2014.

[19] J. J. Murphy, Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications. New York, NY, USA: New York Institute of Finance, 1999.

[20] Hongyang Yang, Xiao-Yang Liu, Shan Zhong, and Anwar Walid. 2020. Deep reinforcement learning for automated stock trading: An ensemble strategy. In Proceedings of the first ACM international conference on AI in finance. 1–8

[21] Vijay Konda and John Tsitsiklis. 2001. Actor-critic algorithms. Society for Industrial and Applied Mathematics 42 (04 2001).

[22] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. arXiv:1707.06347 (07 2017).

[23] Zhipeng Liang, Kangkang Jiang, Hao Chen, Junhao Zhu, and Yanran Li. 2018. Adversarial deep reinforcement learning in portfolio management. arXiv: Portfolio Management (2018).

[24] Volodymyr Mnih, Adrià Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. The 33rd International Conference on Machine Learning (02 2016).

[25] Zihao Zhang. 2019. Deep reinforcement learning for trading. ArXiv 2019 (11 2019).

[26] Timothy Lillicrap, Jonathan Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. International Conference on Learning Representations (ICLR) 2016 (09 2015)

[27] Zhuoran Xiong, Xiao-Yang Liu, Shan Zhong, Hongyang Yang, and A. Elwalid. 2018. Practical deep reinforcement learning approach for stock trading. NeurIPS Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy, 2018. (2018).

[28] Berry MW, Mohamed A, Yap BW. Supervised and Unsupervised Learning for Data Science. New York, NY: Springer; 2019.

[29] Zanero S, Serazzi G. Unsupervised learning algorithms for intrusion detection. Paper presented at: Proceedings of the IEEE Network Operations and Management Symposium. Salvador, Bahia, Brazil: IEEE; 2008:1043-1048.

[30] Laureiro-Martínez, Daniella, et al. "Understanding the exploration–exploitation dilemma: An fMRI study of attention control and decision-making performance." Strategic management journal 36.3 (2015): 319-338.

[31] Yogeswaran, Mohan, and S. G. Ponnambalam. "Reinforcement learning: Exploration–exploitation dilemma in multi-agent foraging task." Opsearch 49 (2012): 223-236.

[32] Peters, Jan, and Stefan Schaal. "Natural actor-critic." Neurocomputing 71.7-9 (2008): 1180-1190.

[33] Grondman, Ivo, et al. "A survey of actor-critic reinforcement learning: Standard and natural policy gradients." IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42.6 (2012): 1291-1307.

[34] Chong, Terence Tai-Leung, Wing-Kam Ng, and Venus Khim-Sen Liew. "Revisiting the Performance of MACD and RSI Oscillators." Journal of risk and financial management 7.1 (2014): 1-12.

[35] Maitah, Mansoor, et al. "Commodity channel index: Evaluation of trading rule of agricultural commodities." International Journal of Economics and Financial Issues 6.1 (2016): 176-178.

[36] Gurrib, Ikhlaas. "Performance of the Average Directional Index as a market timing tool for the most actively traded USD based currency pairs." Banks and Bank Systems 13.3 (2018): 58-70.

[37] Brockman, Greg, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. "Openai gym." arXiv preprint arXiv:1606.01540 (2016).

[38] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. 2017. OpenAI Baselines. https://github.com/openai/baselines.

[39] Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore, Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. 2018. Stable baselines. https://github.com/hill-a/stable-baselines.

[40] Richard Sutton and Andrew Barto. 1998. Reinforcement learning: an introduction. IEEE Transactions on Neural Networks 9 (02 1998), 1054

[41] Richard Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. Conference on Neural Information Processing Systems (NeurIPS), 1999 (02 2000).