Agente de Negociação de Ações Utilizando Aprendizado Por Reforço

Double Blind

***Resumo*—Automated stock trading is now the de-facto way that investors have chosen to obtain high profits in the stock market while keeping risk under control. One of the approaches is to create agents employing Reinforcement Learning (RL) algorithms to learn and decide whether or not to operate in the market in order to achieve maximum profit. Automated financial trading systems can learn how to trade optimaly while interacting with the market pretty much like a human investor learns how to trade. This paper implements a simple RL agent employing the SARSA algorithm and test it against 6 stocks from brazilian stock market Bovespa. Results from experiments showed that the agent not only was able to detect trend changes in stock prices but also was able to provide high profits when compared to the Buy-and-Hold strategy.**

***Index Terms*—Inteligência Artificial, Aprendizado por Reforço, Investimentos, Bolsa de Valores, Ações**

I. Introdução

Um dos fatores determinantes para o sucesso de uma estratégia de investimento no mercado de ações em bolsas de valores é saber qual a melhor decisão a ser tomada em cada situação do mercado. Investidores experientes estão sempre atentos as cotações de preços de ativos financeiros além de outras informações com o objetivo de detectar padrões que os permitam tomar a melhor decisão.

Porém, a maioria das negociações que ocorrem atualmente nas principais bolsas de valores do mundo são executadas por agentes automatizados (*Financial Trading Systems*) também chamados robôs de negociação. Esses sistemas utilizam geralmente técnicas de Inteligência Artificial para detectar padrões em dados de preços, volume financeiro, notícias e então executar a operação mais adequada para os objetivos do investimento naquele momento.

Uma das possíveis abordagens computacionais utilizadas nesses sistemas é o Aprendizado por Reforço (*Reinforcement Learning*) em que um sistema, também denominado agente, aprende a associar ações (decisões) a situações (estados) por meio da interação com o ambiente com o objetivo de maximizar uma medida de desempenho (acúmulo de recompensas ou reforço) em uma tarefa [13].

Essa abordagem é semelhante a maneira como humanos aprendem várias tarefas durante a vida. Por exemplo, uma criança aprende a andar de bicicleta ou *skate* interagindo com esses objetos (ambientes) e por meio de tentativa e erro vai aprendendo a relação de causa e efeito entre a ação tomada e a consequência obtida em cada situação e assim melhorando o seu desempenho através da experiência ao longo do tempo.

Sistemas que empregam aprendizado por reforço tem obtido sucesso nos últimos tempos em diversas aplicações como jogos RTS (*real-time strategy*), roteamento de veículos autônomos e até aceleração de descobrimento de medicamentos [11], [12], [14].

No contexto de negociação de ações um sistema utilizando aprendizado por reforço é capaz de aprender a associar a melhor decisão (comprar, vender, não operar) a cada estado (situação) do mercado de modo a otimizar uma medida de desempenho (e.g., rendimento financeiro, *Sharpe ratio*, *drawdown*) através da interação com o mercado. Além disso, esse sistema também é capaz de modificar o que aprendeu anteriormente de forma dinâmica a medida que uma determinada decisão não mais contribui para o acúmulo de recompensas em um determinado estado do ambiente. Isso confere a esse tipo de sistema uma característica adaptativa que não é comum na maioria dos sistemas baseados em aprendizado supervisionado.

Considerando ainda que as condições do mercado variam bastante ao longo do tempo, a utilização de agentes autônomos de negociação capazes de se adaptarem dinamicamente as constantes mudanças do mercado seria uma vantagem desejável pois evitaria tempo perdido com retreinamentos e proporcionaria um melhor aproveitamento de oportunidades de negócios.

A partir disso, foi utilizado nesse trabalho o algoritmo de aprendizado por reforço SARSA para desenvolver um agente autônomo de negociação de ações que posteriormente foi testado para a negociação de 6 ações da Bolsa de Valores de São Paulo (B3). Além disso, analisou-se os resultados experimentais do ponto de vista financeiro, comparando-os também com a estratégia *Buy-and-Hold*.

II. Trabalhos Relacionados

Um dos primeiros trabalhos a utilizar aprendizado por reforço no contexto de sistemas de negociação automatizados é o de Moody [9]. Nesse trabalho os autores utilizaram o algoritmo *Q-Learning* e também propuseram o algoritmo chamado aprendizado por reforço recorrente *RRL - Recurrent Reinforcement Learning*. Os resultados demonstraram que os algoritmos de aprendizado por reforço utilizados foram capazes de superar em termos de rendimento financeiro sistemas implementados com algoritmos de aprendizado supervisionado como as redes neurais MLP (*Multi-Layer Perceptron*) na época.

Posteriormente os mesmos autores [10] utilizaram o mesmo sistema para negociar um portfólio contendo o índice *S&P500* e o título do tesouro americano *TBill*. Utilizando como medida de recompensa o índice Sharpe, os experimentos apontaram que o sistema RRL superou o algoritmo *Q-Learning* em rendimento financeiro e risco tendo sido ainda capaz de gerar uma estratégia de investimento com menos operações além de ambos terem também superado o *baseline Buy-and-Hold*.

Em 2007 Lee, Park e outros [8] criaram um sistema multi-agente cooperativo para negociação de ações utilizando o algoritmo *Q-Learning*. Esse sistema constitui-se de quatro agentes sendo dois deles responsáveis por gerar sinais de comprar e vender e os outros dois agentes responsáveis por gerar o melhor valor para a compra ou para a venda. Os resultados dos experimentos desse sistema em dados da bolsa de valores da Coréia do Sul superaram outras abordagens baseadas em aprendizado supervisionado principalmente em relação ao número de ordens.

Nesse mesmo ano Chen, Mabu e Hirasawa [2] utilizaram uma abordagem híbrida combinando o algoritmo de aprendizado por reforço *SARSA* e algoritmo genético para criar estratégias de investimento para negociação de ações a partir de indicadores de Análise Técnica. O sistema chamado *GNP-SARSA* foi testado em ações da bolsa de valores de Tóquio no Japão e foi capaz de gerar regras que superaram a estratégia de investimento *Buy-and-Hold*.

Em 2015 Corazza e Sangalli [4] compararam a performance de dois agentes, um implementando o algoritmo *SARSA* e outro *Q-Learning* para negociar um conjunto de ações da bolsa de valores de Milão na Itália. Os experimentos indicaram que ambos os agentes superaram métricas de *baseline* sendo o algoritmo *SARSA* mais sensível a mudanças bruscas no mercado enquanto o algoritmo *Q-Learning* foi melhor ao explorar o mercado gerando mais ordens de compra e venda.

Em 2018, Chen, Chen e Huang [3] desenvolveram um sistema de aprendizado por reforço para clonar estratégias de investimentos de investidores experientes. O sistema desenvolvido foi testado com dados do índice futuro *TAIFEX* da bolsa de valores de Taiwan. Experimentos realizados pelos autores demonstraram que o sistema foi capaz de acertar em até 80% as ações tomadas por um investidor experiente. Ding,Liu e Biang [6] também utilizaram aprendizado por reforço para extrair conhecimento a partir do histórico de negociação de 3 investidores modelos. Utilizando o conhecimento adquirido o sistema desenvolvido pelos autores mostrou-se eficiente em gerar estratégias que superaram *baselines* como *Buy-and-Hold* ao negociar ações do índice *HS300* da Bolsa de Valores de Shangai na China.

Abordagens combinando aprendizado profundo (*Deep Learning*) e aprendizado por reforço tem surgido nos últimos anos sendo chamada de *Deep Reinforcement Learning*.

Deng, Bao, Kong e outros [5] utilizaram *Deep Reinforcement Learning* para criar um agente para negociação de ações e títulos futuros das bolsas de valores da China, Japão e Estados Unidos. O sistema desenvolvido utilizou redes neurais de convolução e LSTM (*Long-Short Term Memory*) além do algoritmo de aprendizado por reforço RRL de Moody e Saffell. Tendo sido o primeiro trabalho que utilizou esse algoritmo para mercado de ações, os autores foram capazes de demostrar a sua viabilidade para esse problema. Outros trabalhos semelhantes como os de Gao [7] e o de Xiong, Liu e Zhong [15] utilizaram redes neurais LSTM (*Long-Short Term Memory*) e o algoritmo de aprendizado por reforço *Q-Learning* apresentando resultados superiores a *baselines* baseados em aprendizado supervisionado e a estratégia *Buy-and-Hold*.

Esse trabalho segue a mesma linha dos anteriores que desenvolveram agentes para negociar ações. O agente de aprendizado por reforço implementado utiliza um negocia uma só ação por vez assim como no trabalho de Corazza e Sangalli [4]. Mas diferentemente destes, esse trabalho utilizou ações separadas em contextos de tendência e lateralidade para analisar o desempenho do sistema nessas situações. Como na maioria dos trabalhos citados o agente implementado realiza tanto operações compradas como vendidas semelhante ao trabalho de Moody [9]. Além disso, seguindo a mesma abordagem desse último trabalho com relação a posição assumida pelo agente, uma vez posicionado o agente não pode aumentar sua posição mas apenas mantê-la ou sair da posição.

III. Fundamentação Teórica

Em um problema de aprendizado por reforço o sistema que aprende uma tarefa é chamado de *agente*. Através da interação com o ambiente o agente é capaz de aprender a associar as melhores ações para cada estado do ambiente. Essa interação consiste essencialmente em escolher e executar uma ação em cada estado do ambiente no tempo e avaliar a consequência da ação tomada naquele estado. Assim, ao executar uma ação, o ambiente responde ao agente no instante seguinte mudando de estado e enviando ao agente uma resposta numérica chamada *recompensa* ou *reforço*. Utilizando o valor dessa recompensa o agente pode então avaliar se a ação executada naquele estado foi boa ou ruim.

De modo mais preciso, o comportamento de um agente de aprendizado por reforço (Figura 1) ocorre da seguinte forma: dado um estado $e_t \in E$ em um instante de tempo $t \in \{0,1,2,3,...\}$, o agente escolhe uma ação $a_t \in A(e)$ que acredita a partir de seu conhecimento levar a estados de maior expectativa de acúmulo de recompensas. Escolhida a ação a_t, o agente executa essa ação fazendo com que o ambiente mude para o estado e_{t+1} e retorne para o agente uma recompensa $r_{t+1} \in \mathbb{R}$. O agente, por sua vez, atualiza sua experiência a respeito da consequência de tomar a ação a_t no estado s_t e receber a recompensa r_{t+1}. Em seguida, o agente prossegue escolhendo uma nova ação a_{t+1} a ser executada no estado e_{t+1} e segue dessa forma a cada novo estado até encontrar o estado final.

Um problema de aprendizado por reforço pode ser descrito formalmente como um Processo de Decisão de Markov (*Markov Decision Process*). Considerando uma sequencia discreta de tempo $t \in \{0,1,2,3,...\}$, um MDP pode ser definido por:

- Um conjunto finito E de estados $e_t \in E$ do ambiente.
- Um conjunto finito $A(e)$ de ações possíveis $a_t \in A(e)$ para cada estado $e \in E$.
- Um modelo probabilístico da dinâmica de estados e recompensas do ambiente $p(e_{t+1}, r_{t+1}|e_t, a_t) = \mathbb{P}(E_{t+1} = e_{t+1}, R_{t+1} = r_{t+1}|E_t = e_t, A_t = a_t)$
- Uma função de recompensa $R(e_t, a_t) = \sum_{r \in R} \cdot \sum_{e \in E} p(e, r|e_t, a_t)$

O conhecimento adquirido pelo agente é modelado na forma de uma função chamada *função-valor*. Assim, a maioria dos

algoritmos de aprendizado por reforço consistem em estimar uma função-valor [13]. Um tipo de função-valor é a função de estado-ação $Q(e_t, a_t)$. O processo de estimação dessa função utiliza métodos de programação dinâmica baseados na equação de otimalidade de Bellman [1]

$$Q^*(e_t, a_t) = \sum_{e \in E, r \in R} p(e, r | e_t, a_t)[r + \gamma \max_{a \in A(e)} Q_*(e, a)] \quad (1)$$

O valor dessa função para cada par (e_t, a_t) indica a quantidade esperada de recompensas acumuladas que um agente pode obter se começar no estado e_t e tomar a ação a_t. Uma vez obtida essa função, o agente pode escolher as ações que maximizam $Q(e, a)$ para cada estado que encontrar. Esse mapeamento de estados em ações que levam ao máximo acúmulo de recompensas é chamado de política (*policy*) ótima π^*. A política ótima pode ser extraída da função de estado-ação ótima $Q^*(e_t, a_t)$ [2]

$$\pi^*(e_t) = \arg_{a_t \in A(e_t)} \max Q^*(e_t, a_t) \quad (2)$$

Contudo, para estimar essa função o agente necessita explorar ou experimentar ações que não necessariamente levam a estados de alto valor na função de estado-ação $Q(e_t, a_t)$ e consequentemente podem levar a um menor acúmulo de recompensas. Por outro lado, se o agente sempre escolher a ação que maximiza $Q(e_t, a_t)$ em todo estado e_t ele poderá deixar de conhecer estados de mais alto valor e que levam a um maior acúmulo de recompensas. Esse dilema, chamado *exploration* v. *exploitation*, é comum em aprendizado por reforço e existem várias técnicas que buscam balancear esses dois aspectos do aprendizado. Uma delas é chamada $\varepsilon - greedy$ e consiste em selecionar com probabilidade ε uma ação a_t que não maximiza a função de estado-ação $Q(e_t, a_t)$ em um estado e_t ou então selecionar de maneira gulosa com probabilidade $(1 - \varepsilon)$ a ação que maximiza a função de estado-ação.

Os algoritmos *Policy Iteration* e *Value Iteration* utilizam a equação de otimalidade de Bellman para resolver um problema de aprendizado por reforço modelado como um MDP. Por isso são chamados algoritmos *baseados em modelo* porque necessitam do modelo probabilístico da dinâmica de transição de estados e recompensas do ambiente. Porém, na prática é muito raro senão impossível obter um modelo da dinâmica do ambiente. Por exemplo, é inviável estimar um modelo de transição de estados e recompensas de um determinado jogador de xadrez dada a enorme quantidade de estados possíveis no jogo bem como outras particularidades do próprio jogo e do adversário. Por isso, utiliza-se algoritmos chamados *model-free* que dispensam um modelo do ambiente.

Dentre os algoritmos *model-free* estão os algoritmos de diferença temporal (*temporal difference*) dos quais *SARSA* e *Q-Learning* são exemplos. O que esses algoritmos fazem é observar a diferença entre a estimativa atual da função de estado-ação $Q_t(e_t, a_t)$, o valor descontado da função de estado-ação para o próximo estado e_{t+1} e a recompensa obtida r_{t+1} [1] para então corrigir a estimativa anterior. Assim, quando o

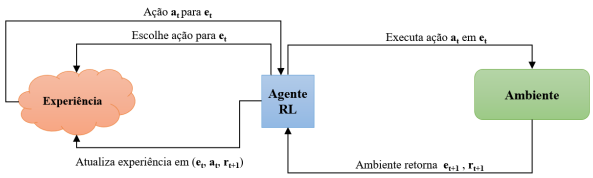

Figura 1: Comportamento de um agente de aprendizado por reforço

agente no estado e_t escolhe uma ação a_t e executa essa ação indo para o estado e_{t+1} e obtém a recompensa r_{t+1} a função de estado-ação pode ser atualizada como [3]

$$Q_t(e_t, a_t) = r_{t+1} + \gamma \cdot \max_{a_{t+1}} Q_t(e_{t+1}, a_{t+1}) \quad (3)$$

Essa técnica baseia-se na ideia de que como o valor da função de estado-ação $Q_t(e_{t+1}, a_{t+1})$ corresponde ao instante posterior ela tem mais chance de estar correta. Esse valor pode ser descontado pelo fator de desconto $\gamma \in (0, 1]$ e somado à recompensa obtida tornando-se o novo valor para a estimativa $Q_t(e_{t+1}, a_{t+1})$ [1].

Os algoritmos *SARSA* e *Q-Learning* utilizam regras diferentes para atualização da função estado-ação mas ambos utilizam o conceito de diferença temporal. No algoritmo *SARSA* [1], descrito em [1], a regra de atualização contém um parâmetro $\eta \in (0, 1]$ chamado taxa de aprendizado. Além disso, existe ainda a taxa de desconto γ que pondera a importância dada as recompensas futuras em relação as recompensas mais recentes.

Inicializa todos os valores de Q(e,a) arbitrariamente;
foreach *episodio* **do**
  Inicializa estado e;
  Escolhe ação a em Q usando $\varepsilon - greedy$;
  repeat
    Executa ação a;
    Obtem recompensa r e proximo estado e';
    Escolhe proxima ação a' em Q com $\varepsilon - greedy$;
    `// Atualiza Q(e,a)`
    $Q(e,a) \leftarrow Q(e,a) + \eta(r + \gamma Q(e', a') - Q(e,a))$;
    $s \leftarrow s'$;
    $a \leftarrow a'$;
  until *s é estado terminal*;
end

Algorithm 1: Algoritmo *SARSA* [1]

IV. Modelagem

O objetivo do agente implementado é maximizar o retorno financeiro negociando uma ação na bolsa de valores. Para isso o agente interage com o mercado comprando, vendendo ou não operando de modo a aprender dinamicamente a melhor decisão (comprar, vender, não operar) a ser tomada em cada estado do mercado.

Como em todo problema de aprendizado por reforço isso envolve a definição de um espaço de estados, um conjunto de ações e uma função de recompensa.

A. Espaço de Estados

Utilizou-se para esse sistema um espaço de estados discreto contendo 9 variáveis categóricas. Logo, para cada instante de tempo $t \in \{0,1,2,3,...,T\}$ temos um estado e_t definido como uma tupla de 9 variáveis descritas abaixo:

1) Tipo de posição: (`LONG`, `SHORT`, `NPOS`). Essa variável descreve o tipo de posição do agente na ação. O valor `LONG` refere-se a uma posição comprada, `SHORT` refere-se a uma posição vendida e `NPOS` denota que o agente não está posicionado e portando não possui nenhuma ação no momento.
2) Variação de preço em $t-2$: (`UP`, `DOWN`). Essa variável descreve a direção da variação do preço da ação no tempo $t-2$. O valor `UP` denota que o preço da ação subiu no tempo $t-2$ e o valor `DOWN` denota que o preço caiu.
3) Variação de preço em $t-1$: (`UP`, `DOWN`). Essa variável descreve o mesmo que a variável anterior mas refere ao tempo $t-1$.
4) Ação tomada em $t-1$: (`BUY`, `SELL`, `NOP`). Essa variável descreve a decisão que foi tomada pelo agente no tempo anterior. `BUY` denota que o agente comprou a ação, `SELL` denota que o agente vendeu a ação e `NOP` significa que o agente não operou.
5) Variação do OBV em $t-2$: (`UP`, `DOWN`). Essa variável descreve a variável do indicador técnico OBV (*On-Balance Volume*) no tempo $t-2$. O valor `UP` indica que houve um aumento do volume financeiro negociado na ação e o valor `DOWN` indica uma diminuição.
6) Variação do OBV em $t-1$: (`UP`, `DOWN`). Essa variável descreve o mesmo que a variável anterior mas no tempo $t-1$.
7) Extremo mais próximo do preço de fechamento em $t-2$: (`MAX`, `MIN`). Essa variável indica se o preço de fechamento da ação no tempo $t-2$ estava mais próximo do preço máximo (`MAX`) da ação ou do preço mínimo (`MIN`).
8) Extremo mais próximo do preço de fechamento em $t-1$: (`MAX`, `MIN`). Essa variável descreve o mesmo comportamento da variável anterior mas no tempo $t-1$.
9) Retorno na posição em $t-1$: (`UP`, `DOWN`, `NPOS`). Indica se a posição atual do agente em relação ao preço da ação em $t-1$ é positiva (`UP`), negativa (`DOWN`) ou não está posicionado (`NPOS`).

Dessa forma, o espaço de estados modelado com as 9 variáveis acima contém 1728 estados possíveis.

B. Conjunto de Ações do Agente

Em cada estado e_t o agente pode tomar as seguintes ações $a_t \in A(e_t)$ dependendo do tipo de posição em que estiver:

- COMPRAR: {`ENTER_LONG`, `EXIT_SHORT`}. A ação `ENTER_LONG` inicia uma posição comprada (`LONG`) e a ação `EXIT_SHORT` sai de uma posição vendida (`SHORT`).
- VENDER: {`ENTER_SHORT`, `EXIT_LONG`}. A ação `ENTER_SHORT` inicia uma posição vendida (`SHORT`) e a ação `EXIT_LONG` sai de uma posição comprada (`LONG`).
- NOP: {`STAY_LONG`, `STAY_SHORT`, `NOP`}. A ação `STAY_LONG` determina que o agente permaneça em uma posição comprada (`LONG`). A ação `STAY_SHORT` determina que o agente permaneça em uma posição vendida (`SHORT`) e a ação `NOP` indica que o agente permaneça não posicionado (`NPOS`).

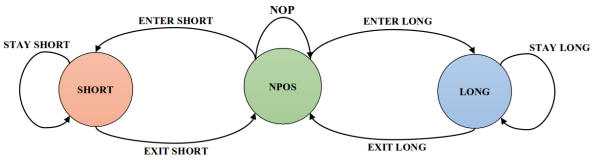

Figura 2: Máquina de estados do agente

O conjunto de ações acima descreve portanto uma máquina de estados, conforme ilustra a Figura 2.

Note que se o agente estiver posicionado em `LONG` ou `SHORT` as ações que ele pode executar correspondem a permanecer na respectiva posição {`STAY_LONG`, `STAY_SHORT`} ou sair da posição {`EXIT_LONG`, `EXIT_SHORT`}. Portanto, não pode o agente aumentar sua posição.

Toda operação COMPRAR ou VENDER é executada a mercado pelo preço de fechamento atual.

C. Função de Recompensa

Para cada ação $a_t \in A(e_t)$ tomada pelo agente no instante t e no estado atual e_t o ambiente retorna ao agente no instante seguinte $t+1$ uma recompensa R_{t+1} que vai depender do tipo de posição assumida pelo agente no instante atual Assim, nas ações a_t de uma posição comprada `ENTER_LONG`, `STAY_LONG` e `EXIT_LONG` a recompensa será o retorno referente ao preço que o agente comprou a ação P_{enter_long} e o preço da ação P_{t+1} no instante seguinte a ação tomada.

Nas ações a_t de uma posição vendida `ENTER_SHORT`, `STAY_SHORT` e `EXIT_SHORT` a recompensa será o retorno referente ao preço que o agente vendeu a ação P_{enter_short} e o preço P_{t+1} da ação no instante seguinte.

Se o agente não está posicionado (`NPOS`) no instante atual e decide não operar `NOP` então a sua recompensa será r_{t+1} será 0.

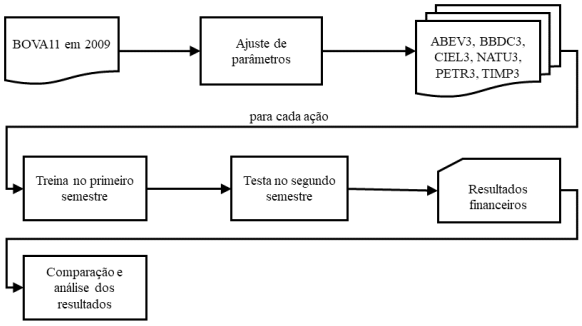

Figura 3: Etapas da metodologia

Portanto, a função de recompensa foi definida como:

$$r_{t+1} = \begin{cases} \frac{P_{t+1} - P_{enter\_long}}{P_{enter\_long}} & \text{se posicao for LONG} \\ \frac{P_{enter\_short} - P_{t+1}}{P_{enter\_short}} & \text{se posicao for SHORT} \\ 0 & \text{se NPOS} \end{cases} \quad (4)$$

V. Metodologia

Utilizou-se dados diários de 6 ações da Bolsa de Valores de São Paulo (B3) classificadas por tendência durante o período de 1 ano para cada ação (ver Tabela I). O objetivo foi observar a robustez e desempenho do agente em contextos de alta, baixa e pouca tendência.

Ação	Tendência	Ano
ABEV3 (Ambev)	Lateral	2014
BBDC3 (Bradesco)		2011
CIEL3 (Cielo)	Alta	2011
NATU3 (Natura)		2009
PETR3 (Petrobras)	Baixa	2010
TIMP3 (Tim)		2015

Tabela I: Ações classificadas por tendência anual

Antes da execução dos experimentos utilizou-se dados diários de BOVA11 do ano de 2009 para ajuste de parâmetros do sistema e análise convergência. Após essa etapa, selecionou-se cada uma das ações da tabela I e treinou-se o sistema no primeiro semestre da ação e testou-se no segundo semestre.

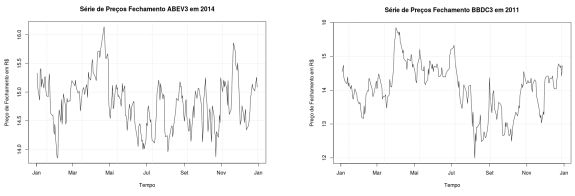

Figura 4: Ações de tendência lateral

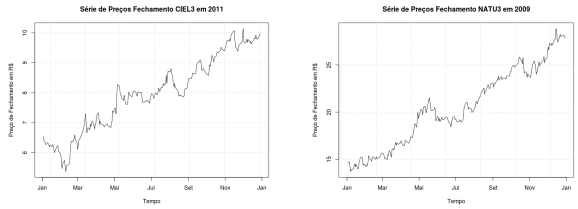

Figura 5: Ações de tendência de alta

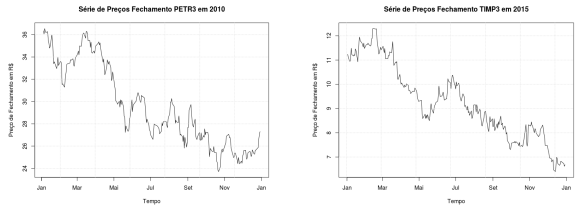

Figura 6: Ações de tendência de baixa

VI. Experimentos e Análise

Para os experimentos, utilizou-se os parâmetros da tabela II:

Obtida a convergência durante o treinamento no primeiro semestre de cada ação, o agente foi executado no segundo segundo semestre com taxa de exploração nula e portanto executando a política aprendida no treinamento.

A. Convergência e Ajuste de Parâmetros

Os parâmetros da tabela II foram obtidos através de busca exaustiva utilizando os dados diários de BOVA11 do ano de 2009.

Para a taxa inicial de exploração foram experimentados valores do intervalo $[0.1, 0.7]$ variando em incrementos de 0.1. Para a taxa de aprendizado foram testados os valores $\{0.1, 0.5, 0.7, 0.01, 0.05, 0.07, 0.001, 0.005, 0.007\}$. Para a taxa de desconto testou-se os valores do intervalo $[0.1, 1.0]$ variando em incrementos de 0.1. Para o número de episódios (iterações) escolheu-se o número 300.000 por ser o valor onde ocorreu a convergência do algoritmo nos dados de BOVA11.

Observa-se que no início a recompensa total varia bastante mas a medida que a taxa de exploração diminui o sistema converge ao final de 300.000 iterações no treinamento.

Ressalte-se ainda que a taxa de exploração ε_t inicia no valor 0.5 na iteração $t = 1$ e decai exponencialmente até o valor 0.0001 na iteração $t = 300000$ seguindo a fórmula:

$$\varepsilon_t = \exp^{\ln(0.5) + t \cdot \frac{\ln(0.0001) - \ln(0.5)}{300000}} \quad (5)$$

Parâmetro	Valor
Taxa inicial de exploração	0.5
Taxa de aprendizado	0.007
Fator de desconto	0.7
Número de episódios	300000

Tabela II: Parâmetros do algoritmo *SARSA*

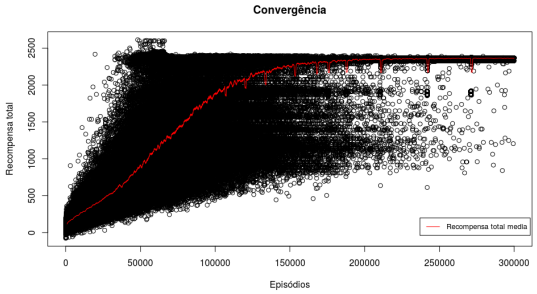

Figura 7: Curva de convergencia do agente SARSA

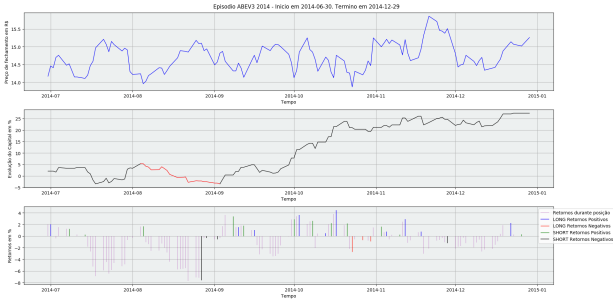

Figura 8: Teste na ação ABEV3 no segundo semestre de 2014

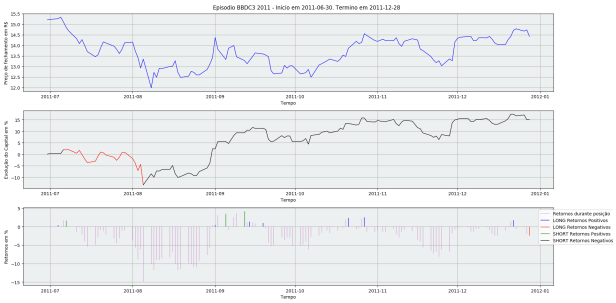

Figura 9: Teste na ação BBDC3 no segundo semestre de 2011

Uma vez alcançada a convergência o agente pode executar a política aprendida na parte de testes da ação correspondente ao segundo semestre.

B. Resultados e Análise para Ações de Tendência Lateral

Nos testes para as ações com tendência lateral, ABEV3 (Figura 8) e BBDC3 (Figura 9), a execução do agente gerou resultado financeiro (vide Tabela 10) que superou a estratégia *Buy-and-Hold* em ambas as ações.

Na Figura 8 o gráfico superior apresenta a série de preços da ação ABEV3 no segundo semestre de 2014. No gráfico do meio observa-se a evolução do rendimento do capital acumulado pelo agente. No gráfico inferior tem-se os retornos relativos para cada ação tomada pelo agente.

Considerando o gráfico do meio nessa figura observa-se que nos meses de julho e agosto o agente obteve pouco rendimento do capital acumulado tendo no mês de setembro ocorrido o máximo *drawdown* nesse teste. A partir desse resultado negativo o agente foi capaz de mudar dinamicamente sua estratégia para nos meses seguintes alcançar o rendimento final de 25.38% executando na maioria das vezes operações compradas `LONG` o que pode ser visto pelas barras azuis que indicam fechamentos positivos de posições `LONG` e as barras roxas superiores que indicam retornos positivos durante as posições.

O mesmo comportamento foi observado no teste da ação BBDC3 na figura 9. Nos mês de julho e início de agosto o agente incorreu no máximo *drawdown* o que fez com que ele mudasse sua estratégia para nos meses seguinte obter o rendimento final de 15.05%.

A partir desses testes pode-se perceber que o agente de aprendizado por reforço foi capaz de mudar dinamicamente a estratégia de investimento perante resultados negativos para maximizar o ganho financeiro final. Esse resultado ilustra uma propriedade importante de sistemas de aprendizado por reforço em cenários de lateralidade.

Ação	Rendimento Total	Buy-and-Hold	Máximo Drawdown
ABEV3	27.38%	4.35%	-8.37%
BBDC3	15.05%	-3.21%	-14.95%

Figura 10: Resultados financeiros para ações com tendência lateral

C. Resultados e Análise para Ações de Tendência de Alta

Já nas ações com tendência de alta, CIEL3 (vide Figura 11) e NATU3 (vide Figura 12) a execução do agente gerou rendimentos financeiros positivos (Tabela 13) embora não tenham superado os rendimentos da estratégia *Buy-and-Hold*.

Na figura 11 do teste da ação CIEL3 no segundo semestre de 2011 nota-se que no mês de agosto o agente obteve rendimentos consideráveis executando operações `LONG` e acompanhando a alta da ação nesse mês. Apesar da queda do preço da ação no mês de setembro o agente ainda foi capaz de manter razoavelmente o rendimento obtido no mês anterior executando poucas operações o que pode ser notado pelas poucas barras de retornos no gráfico inferior dessa figura. Uma vez que o preço da ação voltou a subir em tendência de alta a partir do setembro o agente foi capaz de perceber esse tendência e executou uma série de operações compradas que proporcionaram o rendimento final de 21.09%.

O teste da ação NATU3 no segundo semestre de 2009 ficou ainda mais evidente essa característica de mudança dinâmica de estratégia. Nesse teste o agente de aprendizado por reforço foi capaz de perceber a tendência de alta ao longo de todo período testado e executar uma maior parte de operações compradas (`LONG`) e com isso obteve um rendimento de 41.54%. Note ainda que houve uma queda no preço da ação durante o mês de outubro o que fez com que o agente incorresse no máximo *drawdown*. Percebendo essa tendência de baixa o agente ainda tentou executar uma operação vendida

SHORT mas obteve perda o que pode ser observado pela barra vermelha no gráfico inferior bem como pela tendência de alta no preço da ação no gráfico superior. Assim, com esse resultado negativo o agente foi novamente capaz de mudar sua estratégia e executar operações compradas para acompanhar a tendência de alta da ação.

Novamente, esse testes demonstraram a capacidade do agente de detectar e seguir as tendência geral das ações bem como mudar dinamicamente a estratégia em mudanças momentâneas de tendências.

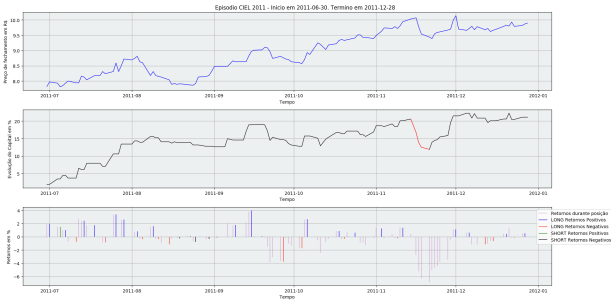

Figura 11: Teste na ação CIEL3 no segundo semestre de 2011

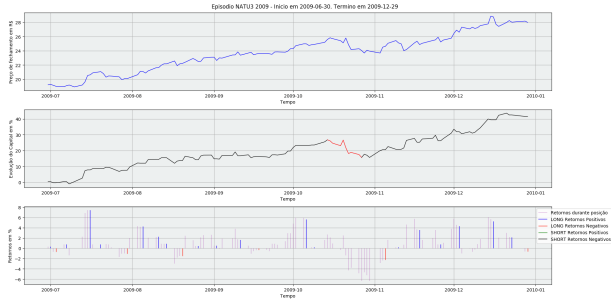

Figura 12: Teste na ação NATU3 no segundo semestre de 2009

Ação	Rendimento Total	Buy-and-Hold	Máximo Drawdown
CIEL3	21.09%	25.09%	-7.21%
NATU3	41.54%	44.21%	-8.77%

Figura 13: Resultados financeiros para ações com tendência de alta

D. Resultados e Análise para Ações de Tendência de Baixa

Na execução do agente nas ações de tendência de baixa, PETR3 (vide Figura 14) e TIMP3 (vide Figura 15) o agente foi capaz de produzir rendimentos financeiros (vide Tabela 16) superiores a estratégia *Buy-and-Hold* em ambas as ações.

Ação	Rendimento Total	Buy-and-Hold	Máximo Drawdown
PETR3	19.88%	0.96%	-10.72%
TIMP3	40.06%	-31.90%	-6.07%

Figura 16: Resultados financeiros para ações com tendência de baixa

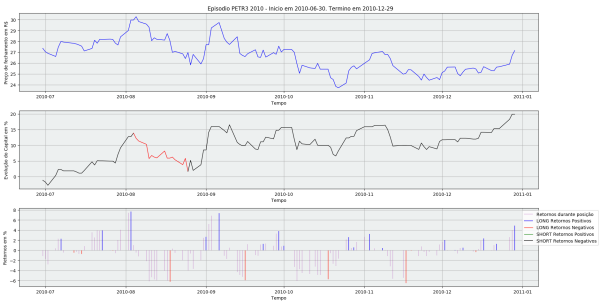

Figura 14: Teste na ação PETR3 no segundo semestre de 2010

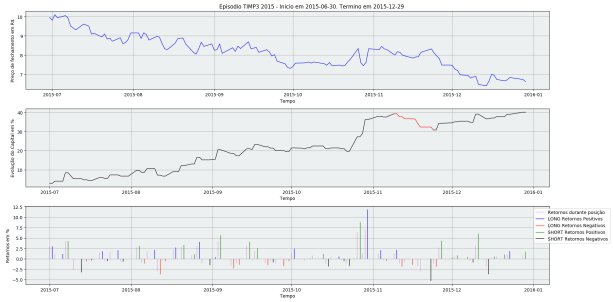

Figura 15: Teste na ação TIMP3 no segundo semestre de 2014

No teste da ação PETR3 (vide Figura 14) no segundo semestre de 2010 o agente iniciou com um rápido crescimento no rendimento no mês de julho apesar de algumas oscilações no preço da ação nesse mês. Com a queda do preço da ação no mês de agosto o agente incorreu numa perda que gerou o seu máximo *drawdown*. Em seguida, devido a oscilação de alta da ação no final do mês de agosto o agente percebeu rapidamente essa tendência recuperando a perda anterior e mantendo a média do rendimento no patamar de 15% para alcançar ao final do mês de dezembro um rendimento final de 19.88%.

O teste da ação TIMP3 (vide Figura 15) no segundo semestre de 2015 também mostrou a capacidade do agente de aprendizado por reforço de detectar a tendência de baixa desde agosto até o final de setembro e executar uma maior parte de operações vendidas SHORT como ser pode observar pela maior quantidade de barras verdes no gráfico inferior da Figura 15 que indicam fechamentos de posições SHORT com retorno positivo.

Note também que no final do mês de outubro houve uma oscilação do preço da ação que foi rapidamente percebida pelo agente que executou uma operação vendida para aproveitar a queda da ação e logo em seguida uma operação comprada para aproveitar a leve a alta do preço da ação tendo em ambas as operações alcançado retornos positivos.

Isso também demonstra como esse tipo de sistema é capaz de detectar mudanças momentâneas de preços e aproveitar

essas oscilações para maximizar o ganho financeiro.

VII. Conclusões

O agente de aprendizado por reforço implementado para negociação de ações utilizando o algoritmo *SARSA* obteve no geral resultado satisfatório em termos de rendimento financeiro nas ações testadas.

Embora não tenha superado a estratégia *Buy-and-Hold* nas ações com tendência de alta, o sistema proporcionou resultados bem próximos dessa estratégia. Além disso, foi capaz de produzir resultados superiores nas ações com tendência de baixa e também nas ações de tendência lateral.

Como esperado para um sistema de aprendizado por reforço, ficou evidenciado a capacidade do sistema de mudar dinamicamente a estratégia ao longo da execução, buscando aproveitar mudanças de momentâneas de tendências.

Assim, embora não tenham sido considerados aspectos de custos de operação bem como estratégias de gerenciamento de risco, o sistema de aprendizado por reforço implementado demonstra a princípio a viabilidade desse tipo de abordagem para criação de robôs de negociação de ações.

Referências

[1] Alpaydin, E. Introduction to Machine Learning The MIT Press, 2014

[2] Chen, Y.; Mabu, S.; Hirasawa, K. & Hu, J. Genetic network programming with sarsa learning and its application to creating stock trading rules Proc. IEEE Congress Evolutionary Computation, 2007, 220-227

[3] Chen, C. T.; Chen, A. & Huang, S. Cloning Strategies from Trading Records using Agent-based Reinforcement Learning Algorithm Proc. IEEE Int. Conf. Agents (ICA), 2018, 34-37

[4] Corazza, M. & Sangalli, A. Q-Learning and SARSA: A Comparison between Two Intelligent Stochastic Control Approaches for Financial Trading SSRN Electronic Journal, Elsevier BV, 2015

[5] Deng, Y.; Bao, F.; Kong, Y.; Ren, Z. & Dai, Q. Deep Direct Reinforcement Learning for Financial Signal Representation and Trading IEEE Transactions on Neural Networks and Learning Systems, Institute of Electrical and Electronics Engineers (IEEE), 2017, 28, 653-664

[6] Ding, Y.; Liu, W.; Bian, J.; Zhang, D. & Liu, T.-Y. Investor-Imitator: A Framework for Trading Knowledge Extraction Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &38; Data Mining, ACM, 2018, 1310-1319

[7] Gao, X. Deep reinforcement learning for time series: playing idealized trading games http://arxiv.org/abs/1803.03916v1, 2018

[8] Lee, J. W.; Park, J.; O, J.; Lee, J. & Hong, E. A Multiagent Approach to Q-Learning for Daily Stock Trading IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, Institute of Electrical and Electronics Engineers (IEEE), 2007, 37, 864-877

[9] Moody, J. E. & Saffell, M. Kearns, M. J.; Solla, S. A. & Cohn, D. A. (Eds.) Reinforcement Learning for Trading Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998], The MIT Press, 1998, 917-923

[10] Moody, J. & Saffell, M. Learning to trade via direct reinforcement IEEE Transactions on Neural Networks, Institute of Electrical and Electronics Engineers (IEEE), 2001, 12, 875-889

[11] Reddy, G.; Wong-Ng, J.; Celani, A.; Sejnowski, T. J. & Vergassola, M. Glider soaring via reinforcement learning in the field Nature, 2018, 562, 236-239

[12] Serrano, A.; Imbernón, B.; Pérez-Sánchez, H.; Cecilia, J. M.; Bueno-Crespo, A. & Abellán, J. L. Accelerating Drugs Discovery with Deep Reinforcement Learning: An Early Approach Proceedings of the 47th International Conference on Parallel Processing Companion, ACM, 2018, 6:1-6:8

[13] Sutton, R. S. & Barto, A. G. Introduction to Reinforcement Learning MIT Press, 2018

[14] Tavares, A. R. & Chaimowicz, L. Tabular Reinforcement Learning in Real-Time Strategy Games via Options 2018 IEEE Conference on Computational Intelligence and Games (CIG), IEEE, 2018

[15] Xiong, Z.; Liu, X.-Y.; Zhong, S.; Hongyang; Yang & Walid, A. Practical Deep Reinforcement Learning Approach for Stock Trading 32nd Conference on Neural Information Processing Systems (NIPS 2018), Montreal, Canada, 2018