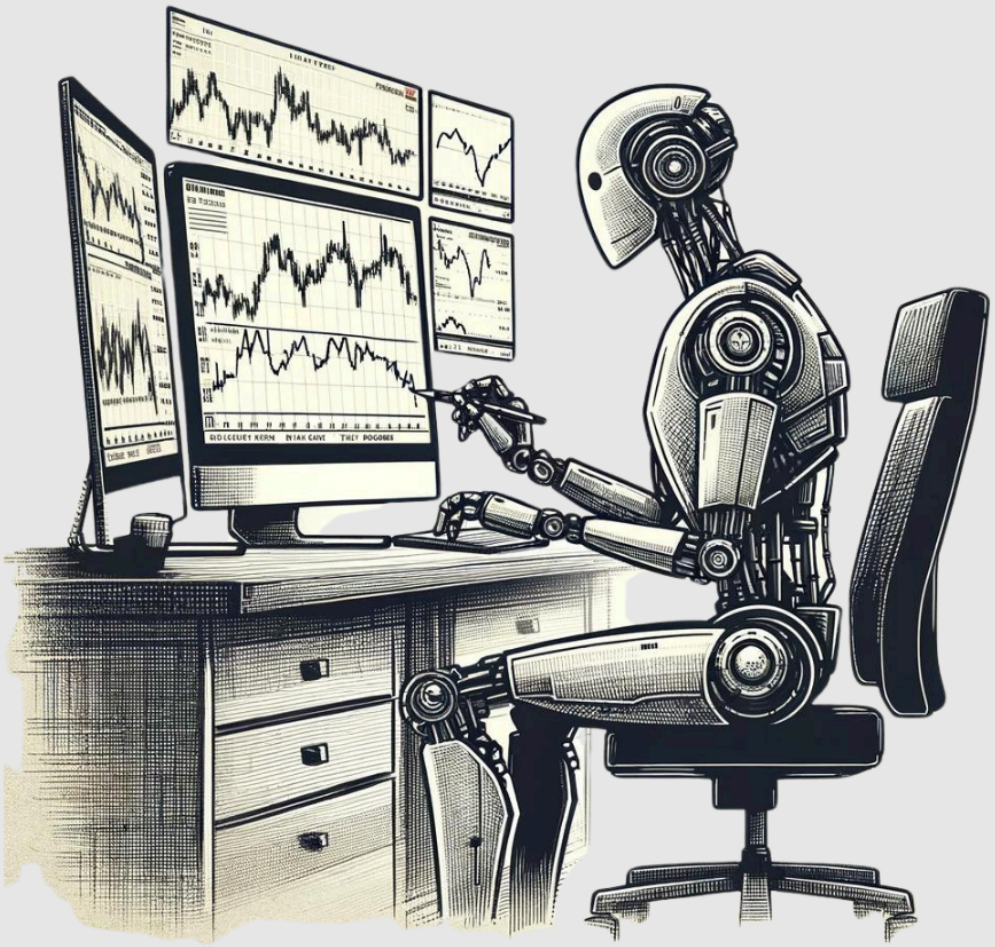
UNIVERSIDADE FEDERAL DE GOIÁS / INSTITUTO DE INFORMÁTICA

Aprendizado por Reforço

Aplicado ao trading na bolsa de valores

Luís Augusto do Prado Assunção



UFG

UNIVERSIDADE
FEDERAL DE GOIÁS

UNIVERSIDADE FEDERAL DE GOIÁS (UFG)
INSTITUTO DE INFORMÁTICA (INF)

LUÍS AUGUSTO DO PRADO ASSUNÇÃO

APRENDIZADO POR REFORÇO

Aplicado ao trading na bolsa de valores

Goiânia
2024

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC no 1240/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei no 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo dos Trabalhos de Conclusão dos Cursos de Graduação disponibilizado no RI/UFG é de responsabilidade exclusiva dos autores. Ao encaminhar(em) o produto final, o(s) autor(a)(es)(as) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG)

Nome(s) completo(s) do(a)(s) autor(a)(es)(as): **LUÍS AUGUSTO DO PRADO ASSUNÇÃO**

Título do trabalho:

APRENDIZADO POR REFORÇO

Aplicado ao trading na bolsa de valores

2. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador) Concorda com a liberação total do documento [**X**] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante: a) consulta ao(à)(s) autor(a)(es)(as) e ao(à) orientador(a); b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo do TCCG. O documento não será disponibilizado durante o período de embargo.

Casos de embargo:

- Solicitação de registro de patente;

- Submissão de artigo em revista científica;

- Publicação como capítulo de livro.

Obs.: Este termo deve ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Luís Augusto Do Prado Assunção**, **Discente**, em 16/02/2024, às 10:07, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fernando Marques Federson**, **Professor do Magistério Superior**, em 12/09/2024, às 11:02, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4383401** e o código CRC **244FE9A9**.

Referência: Processo nº 23070.008389/2024-99 SEI nº 4383401

LUÍS AUGUSTO DO PRADO ASSUNÇÃO

APRENDIZADO POR REFORÇO

Aplicado ao trading na bolsa de valores

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.
Orientador: Prof. Dr. Fernando Marques Federson

Goiânia
2024

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

ASSUNÇÃO, LUÍS AUGUSTO DO PRADO
APRENDIZADO POR REFORÇO [manuscrito] : Aplicado ao trading na bolsa de valores / LUÍS AUGUSTO DO PRADO ASSUNÇÃO. - 2024.
80 f.

Orientador: Prof. Dr. FERNANDO MARQUES FEDERSON.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Instituto de Informática (INF), Inteligência Artificial, Goiânia, 2024.

1. inteligência artificial. 2. aprendizado por reforço. 3. trading. I. FEDERSON, FERNANDO MARQUES, orient. II. Título.

CDU 004

LUÍS AUGUSTO DO PRADO ASSUNÇÃO

APRENDIZADO POR REFORÇO

Aplicado ao trading na bolsa de valores

Relatório final de Trabalho de Conclusão de Curso, apresentado à Universidade Federal de Goiás, como parte das exigências para a obtenção do título de Bacharel em Inteligência Artificial.

Data da Aprovação: 08 de fevereiro de 2024.

Prof. Dr. Fernando Marques Federson
Orientador (INF-UFG)

Prof. Dr. Aldo André Díaz Salazar
Coordenador de TCC do BIA (INF-UFG)

Prof. Dr. Vinícius Sebba Patto
Coordenador do BIA (INF-UFG)

Prof. Me. Leonardo Antônio Alves
(INF-UFG)

LUÍS AUGUSTO DO PRADO ASSUNÇÃO

APRENDIZADO POR REFORÇO

Aplicado ao trading na bolsa de valores

RESUMO

Este Relatório de Conclusão de Curso tem como objetivo reunir os resultados da minha jornada para me tornar um especialista em **Aprendizado por Reforço**. Uma ilustração e sua narrativa descrevem os períodos de trabalho. Os Apêndices contêm os Termos de Aceite de Entrega e os resultados obtidos durante cada período de trabalho.

Palavras-chave: inteligência artificial, aprendizado por reforço, trading.

ABSTRACT

This Course Completion Report aims to bring together the results of my journey to become an expert in **Reinforcement Learning**. An illustration and its narrative describe the work periods. The Appendices contain the Delivery Acceptance Terms and the results obtained during each work period.

Keywords: artificial intelligence, reinforcement learning, trading.

Goiânia

2024

Minha Jornada

Luís Augusto do Prado Assunção

Especialista em: Aprendizado por Reforço

MINHA JORNADA

Nome: Luís Augusto do Prado Assunção

Especialidade: Aprendizado por Reforço

Objetivo deste documento

Durante o processo da disciplina Residência em IA[1], foram gerados diversos resultados na construção da minha especialização. A cada semana, um conjunto de resultados foi formalizado por um Termo de Aceite de Entrega e avaliado por uma banca, considerando o planejado e o realizado para o período. Este documento tem como objetivo descrever esses resultados obtidos, fazendo referência aos Termos de Aceite de Entrega e seus documentos associados.

Minha Jornada

Iniciando minha jornada na **Semana 1**, meu objetivo era estabelecer minha área de especialização na Residência em IA. Influenciado pela disciplina de Aprendizado por Reforço durante meu Bacharelado, que despertou meu interesse, iniciei minha busca por artigos sobre o tema, explorando principalmente as publicações da *Conference on Computational Science and Computational Intelligence* (CSCI). Essa exploração não se limitou a um único evento, incluindo também fontes diversas que julguei pertinentes. Com essa leitura, pude ampliar a minha visão sobre as áreas em que o Aprendizado por Reforço pode ser aplicado. Na **Semana 2**, me dediquei à pesquisa de artigos que exploravam a interseção entre o Aprendizado por Reforço e séries temporais. Terminei por escolher esse tema como área de aplicação. Os detalhes sobre os todos os artigos lidos estão disponíveis no **Apêndice 1**, onde destaco a organização desses estudos conforme a abordagem adotada pelo autor focada em fundamentos, ferramentas ou aplicações.[2]

[1] Dez semanas, entre setembro de 2023 e janeiro de 2024.
[2] Metodologia apresentada pelo professor Federson, F. M; INF/UFG;

Durante a **Semana 3**, após definir a área de aplicação do Aprendizado por Reforço, surgiu a necessidade de determinar o domínio dos dados temporais. Nesse processo, encontrei uma variedade de referências, especialmente nos domínios do varejo e do mercado financeiro. Dado meu interesse maior pela segunda opção, optei por seguir essa direção em minha jornada. A maioria dos estudos no campo financeiro discutiam o uso do Aprendizado por Reforço para automatizar operações de *trading* na bolsa de valores. Contudo, esses trabalhos frequentemente mencionam termos específicos do mercado de investimentos, alguns dos quais eu não estava familiarizado. Decidi, então, dedicar a **Semana 4** a um aprofundamento nesse assunto. Os detalhes sobre os artigos e os termos financeiros estão disponíveis no **Apêndice 2**. Destaco especialmente o survey de Adrian Millea, que compilou uma série de artigos sobre o uso de Aprendizado por Reforço para *trading* na bolsa de valores, apresentando as principais estratégias usadas para abordar esse problema complexo.

Na **Semana 5**, a implementação do ambiente de simulação foi fundamental para a aplicação do Aprendizado por Reforço no contexto de *trading* do mercado financeiro. Nesse cenário, é essencial que o agente tenha um ambiente simulado onde possa interagir e acumular experiência para seu aprendizado. Este ambiente deve refletir os aspectos relevantes do mundo real que está sendo simulado. Ao longo da minha jornada, busquei simular as características principais de uma corretora de investimentos, proporcionando ao agente a visualização das características importantes dos ativos financeiros e a capacidade de realizar operações de compra e venda das ações. A implementação desse ambiente de simulação seguiu os padrões estabelecidos pela biblioteca Gymnasium, desenvolvida pela OpenAI. Os resultados detalhados sobre o ambiente de simulação de *trading* podem ser encontrados no **Apêndice 3**.

Durante a **Semana 6**, optei por selecionar um *framework* de Aprendizado por Reforço para iniciar os testes no ambiente de simulação. Escolhi o RLlib, influenciado positivamente pela minha experiência anterior com essa ferramenta durante a disciplina de Aprendizado por Reforço do Bacharelado. O RLlib oferece uma gama de funcionalidades que facilitam o desenvolvimento de soluções com Aprendizado por Reforço, o que me motivou a aprofundar meu estudo em sua documentação. Na **Semana 7**, além de dar continuidade aos testes, me concentrei em aprimorar a implementação do código disponível

no meu repositório do Github. Os resultados obtidos nessas duas semanas, juntamente com os detalhes sobre o RLlib, descrição dos testes realizados e as alterações feitas no código, estão documentados no **Apêndice 4**.

Na **Semana 8**, durante uma conversa com os professores da Disciplina Residência em IA, compreendi que fornecer apenas o valor da cotação do ativo financeiro não seria suficiente para embasar as decisões de investimento tomadas pelo agente. De fato, Investidores que operam na bolsa de valores frequentemente empregam diversos indicadores para sustentar suas escolhas, visando aumentar a precisão e os lucros. Diante disso, decidi me concentrar nos estudos desses indicadores financeiros, visando aprimorar a representação das informações que compõem o estado do ambiente de simulação fornecido para o agente no processo de aprendizado. Os detalhes sobre os indicadores estudados podem ser encontrados no **Apêndice 5**. As implementações e testes com esses indicadores foram realizadas nas semanas subsequentes.

Na **Semana 9**, percebi que a utilização do RLlib estava impactando negativamente minha trajetória como especialista em Aprendizado por Reforço. Embora o *framework* oferecesse vantagens na agilidade dos testes e desenvolvimento, sua complexidade interna dificultava a compreensão. Diante disso, optei por explorar uma abordagem alternativa, escolhendo o *Deep Q-Learning*, um algoritmo bem estabelecido na área, e adotando uma implementação mais simples disponível em um repositório do GitHub. No contexto do ambiente de simulação, junto com a adição dos indicadores financeiros, entre outros testes interessantes que conduzi, está o uso de diferentes funções de recompensa. Como os primeiros resultados obtidos com essas mudanças foram positivos, na **Semana 10**, continuei com os testes com as funções de recompensa e outros tipos de algoritmos de Aprendizado por Reforço. Os resultados dessas duas semanas de experimentação estão disponíveis no **Apêndice 6**, com uma descrição dos testes realizados e uma comparação entre diferentes algoritmos de Aprendizado por Reforço.

Na minha jornada para me tornar especialista em Aprendizado por Reforço durante a residência em IA, obtive uma evolução significativa no meu conhecimento nessa área. Ao longo desse percurso, fui exposto a diversas etapas cruciais dos problemas de Reforço, desde a concepção de ambientes de simulação até a exploração de *frameworks* especializados, o que me permitiu aprofundar meu entendimento em cada um desses

aspectos. Além disso, ao explorar a aplicação do Aprendizado por Reforço no mercado financeiro, eu adquiri conhecimentos valiosos sobre conceitos nesse domínio que podem se revelar importantes em trabalhos futuros. Enxergo muitas oportunidades de aprimoramento nos resultados obtidos na simulação do mercado financeiro, seja através da inclusão de mais indicadores, desenvolvimento de novas funções de recompensa ou exploração de outros algoritmos. No entanto, considero também a possibilidade de aplicar o Aprendizado por Reforço em outras áreas do meu interesse, como robótica e jogos eletrônicos. Ao concluir este processo, com certeza, me sinto mais próximo de me tornar um especialista em Aprendizado por Reforço do que estava antes de iniciar a Disciplina Residência em IA.

APÊNDICE 1

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 19 de out. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto e Gustavo Barbosa

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

O foco do trabalho do nosso time é o campo de aprendizado por reforço. Nesse sentido, para realização da atividade proposta para o Gate dessa semana, foram analisados os diversos termos relacionados à área de IA presentes no site do evento Conference on Computational Science and Computational Intelligence (CSCI'23). Dentre os termos presentes, foi identificado aqueles mais relacionadas ao campo de aprendizado por reforço:

- Intelligent agents
- Reinforcement learning
- Neural networks

Onde o termo principal é Reinforcement learning (RL). Já Intelligent agents também se encaixa visto que o foco é desenvolver agentes capazes de aprender e tomar decisões cada vez melhores através das experiências obtidas da interação com o ambiente. Quanto ao termo Neural networks, a pretensão é estudar os algoritmos de RL que utilizam redes neurais no aprendizado visto que hoje são as estratégias que atingem os melhores resultados.

Dentre os três termos escolhidos, foi selecionado o "Reinforcement Learning" para ser feita uma pesquisa pelos trabalhos presentes nos anais do CSCI dentro das diversas edições realizadas. Nesta pesquisa foram encontrados artigos de 2016 até 2019 aos quais foi obtido o acesso. Um breve resumo contextualizando o cenário em que cada um deles aplica aprendizado por reforço foi escrito no documento:

https://drive.google.com/file/d/1rT7SOV_aWJ1FZCN6lfrlSKUg9WLek4gH/view?usp=share_link

Vários desses artigos trazem ideias de como o RL pode ser usado em diversas áreas diferentes como, por exemplo, em jogos, setor de energia, mercado financeiro, entre outros.

Dessa forma, com uma estruturação e organização de processos e ideias aprendido durante a disciplina de empreendedorismo do BIA, foi feita uma classificação dos trabalhos em três categorias diferentes. Uma visualização dessa classificação se encontra no link abaixo, além dos artigos do CSCI foram incluídos outros artigos lidos fora do escopo do evento relacionados à RL.

https://drive.google.com/file/d/1wJJJR-Bug5G7ZFLy0ijJF_6lDUlleYzl/view?usp=share_link

A primeira categoria se trata do “conceito” onde se encaixam os artigos lidos cujo propósito é apresentar a ideia geral de como o aprendizado por reforço pode ser utilizado na área em questão. Já na segunda categoria, se encontram as “ferramentas (metodologia)”, onde foram incluídos os artigos que focam em apresentar uma estratégia ou algoritmo para o aprendizado do agente e realizam testes em ambientes mais simples. Já na última categoria de “aplicação (Materialização)” são enquadrados os artigos que focam em fazer testes e que trazem resultados dos métodos de RL aplicados em situações reais.

Em grande parte dos trabalhos lidos é apresentado uma introdução a conceitos básicos de RL, por isso foi uma excelente oportunidade para revisar os fundamentos da área que já haviam sido estudados na disciplina de RL no sexto período do bacharelado em IA. Dentre os conceitos revisados estão:

- Processo de decisão de Markov
- Ambiente, Agente, Estado, Ação, Recompensa
- Retorno esperado
- Função de valor de estado e de estado-ação
- Políticas

Alguns conceitos relacionados às estratégias de aprendizado também foram revisadas, como:

- Off-policy (Ex: Q-learning)
- On-policy (Ex: Sarsa)
- Model-Based
- Model-free

Planejamento: **[descrever o que pretende fazer para realizar a próxima ENTREGA]**

Para a próxima semana planejamos continuar a busca e leitura de artigos na área de RL, porém agora com foco em trabalhos voltados para o contexto de séries temporais. O principal objetivo é dar os primeiros passos para entender as metodologias utilizadas e como as técnicas de RL são adaptadas para esse cenário de séries temporais.

Observação: [caso precise fazer alguma observação, de qualquer “natureza”]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Resumo dos artigos da CSCI de aprendizado por Reforço

Neste texto é apresentado uma breve contextualização dos trabalhos entre 2016 à 2019 presentes no evento Conference on Computational Science and Computational Intelligence (CSCI) relacionadas a Reinforcement Learning (RL). Os artigos lidos apresentam várias ideias e aplicações do uso RL em diferentes áreas como setor elétrico, mercado financeiro, jogos, robótica e outros.

Artigos de 2016

Sahba, em [1], propõe a utilização de RL para segmentação de uma sequência de imagens de um vídeo, o que permite um ajuste dinâmico da segmentação e correções manuais dos usuários no meio do processo. Nesse contexto, as imagens do vídeo são quebradas em sub-imagens do qual são extraídas features pré-estabelecidas para representar um estado. Em relação às ações, elas são definidas como instruções que mudam a cor de cada pixel para ajustar a máscara de segmentação. Já as recompensas são calculadas ao comparar a segmentação feita pelo agente com versões manualmente segmentadas das imagens de treino. O algoritmo utilizado é o Deep Q-learning para aproximar os valores de estado-ação.

Artigos de 2017

Ensinar robôs autônomos a planejar e seguir uma rota sozinho desviando de obstáculos é uma tarefa complicada. Contudo, em [2], kwak, Yoon e Roh propõem a utilização de RL, com Deep learning, para ensinar múltiplos robôs a aprenderem a seguir uma rota desviando de obstáculos pelo caminho. O algoritmo utilizado foi Deep Q-learning (DQN) cuja a entrada da rede neural consiste em dados de um scanner com lasers obtendo distâncias e ângulos dos obstáculos, o ponto da rota mais próximo, velocidade, orientação e ações anteriores. No espaço das ações é composto por 5 movimentos possíveis, dentre eles estão mover para frente ou para trás, rotacionar no sentido horário ou anti-horário e parar. No artigo, como ponto positivo para reprodutibilidade, é apresentado todos os hiperparâmetros necessários para construir a DQN.

Em relação a [3], é proposto por Lu et al. um cenário relacionado ao setor elétrico. Nesse ambiente temos o operador da rede elétrica que fornece a energia ao provedor de serviços e este, por sua vez, repassa a energia elétrica para os clientes. Dessa forma, um agente é o provedor de serviço que vai ajustar o preço cobrado de cada cliente de acordo com o perfil de consumo e o preço da energia comprada do operador, a fim de maximizar os lucros obtidos. Além disso, um segundo agente é proposto para representar os clientes cujo objetivo é minimizar os gastos com energia elétrica, contudo outro objetivo também modelado é o desconforto que precisa ser minimizado, onde ele aumenta conforme o cliente deixa de usar equipamentos elétricos para economizar energia. Assim, um dos agentes deve aumentar o lucro do provedor e outro diminuir o gasto do cliente. O algoritmo proposto

para aprendizado se trata do Q-Learning.

Em [4], o Boyd e Barbosa apresentam uma maneira de aplicar RL para criação da inteligência artificial de inimigos em videogames. Como ferramenta é utilizado a Unreal Engine para criação do ambiente cujos agentes interagem, bem como o sistema blueprint que permite a implementação do código de maneira visual. A estratégia adotada para o aprendizado foi Q-learning, onde o espaço de estados é discreto composto por 5 indicadores: Visão do inimigo, sofrendo dano, causando dano e vida em estado crítico. Enquanto as ações que o inimigo (agente) pode realizar se resumem a apenas 3: correr, atacar e explorar. Incluindo algumas restrições de combinações, a tabela de valor estado-ação é pequena, com apenas 80 possibilidades.

O trabalho [5] de Rioual et al. voltado para parte de energia elétrica, assim como em [3], apresenta uma estratégia para o gerenciamento da energia elétrica em grandes sistemas de IoT. Os sensores de um sistema IoT possuem diferentes restrições quanto a necessidade de energia elétrica, alguns possuem bateria e suportam ficar ligado algum tempo sem energia, enquanto outros dependem de um acesso constante à energia para funcionar. Dessa forma, RL é aplicado para gerenciar o uso de energia do sistema a fim de minimizar os custos com energia elétrica, mas manter todos componentes da rede funcionando da maneira esperada.

Artigos de 2018

No artigo [6], Sabharwal e Anjum fazem uma revisão sobre os grandes impactos gerados pelos avanços dos consultores robóticos no setor financeiro. Graças ao machine learning, tais consultores são capazes de analisar milhares de dados e oferecer serviços como detecção de fraude, construção de portfólios de investimento, análise de crédito, entre outros. Em especial, no artigo é mencionado o potencial do RL nessa área e cita outros trabalhos que aplicaram essa estratégia para construção automática de portfólio de investimentos.

Artigos de 2019

Estratégias de RL são aplicadas para um agente jogar o “jogo da cobrinha” dos celulares antigos da Nokia, em [7]. Almalki e Wocjam aplicam o algoritmo SARSA em uma versão personalizada do jogo onde são adicionados novos desafios para o agente lidar.

Asher et al. em [8] apresentam uma forma de visualizar o trabalho em equipe em um ambiente onde há vários agentes em busca de um mesmo objetivo. No artigo é utilizado um ambiente presa-predador a título de exemplo, onde existem 3 agentes predadores cujo alvo é um quanto agente que representa a presa. É utilizada a estratégia multi-agent deep deterministic policy gradient (MADDPG), onde há uma única rede central como crítico que tem acesso às experiências coletadas por todos os agentes. Para mensurar essa cooperação entre os predadores, o autor propõe um método de perturbação nas experiências de um predador e analisa o impacto disso nas ações dos demais agentes.

Referências:

[1] F. Sahba, "Deep Reinforcement Learning for Object Segmentation in Video Sequences," 2016 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2016, pp. 857-860, doi: 10.1109/CSCI.2016.0166.

[2] N. Kwak, S. Yoon and K. Roh, "Learning Motion Policy for Mobile Robots Using Deep Q-Learning," 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2017, pp. 805-810, doi: 10.1109/CSCI.2017.139.

[3] R. Lu, S. H. Hong, X. Zhang, X. Ye and W. S. Song, "A Perspective on Reinforcement Learning in Price-Based Demand Response for Smart Grid," 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2017, pp. 1822-1823, doi: 10.1109/CSCI.2017.327.

[4] R. A. Boyd and S. E. Barbosa, "Reinforcement Learning for All: An Implementation Using Unreal Engine Blueprint," 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2017, pp. 787-792, doi: 10.1109/CSCI.2017.136.

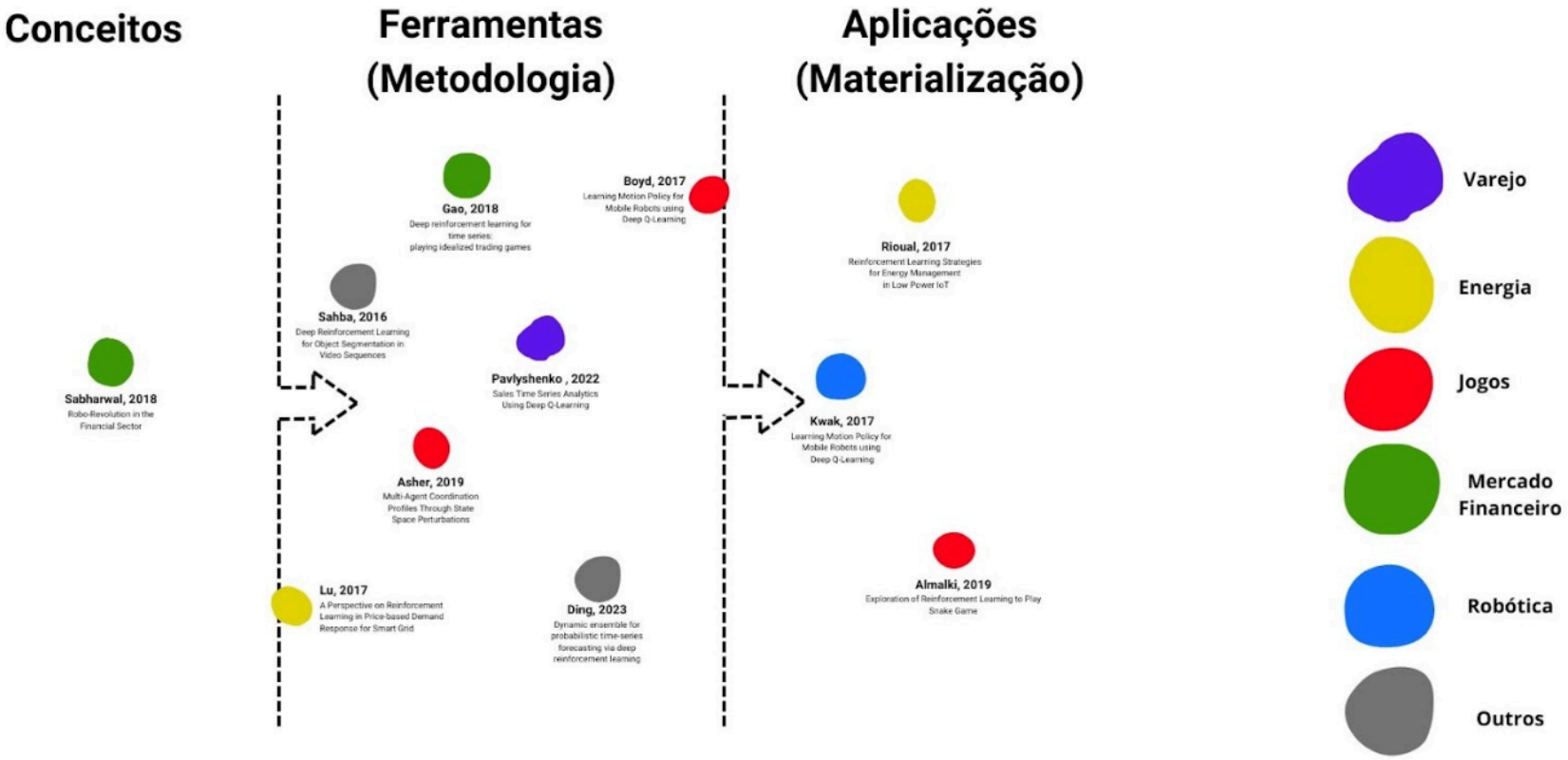
[5] Y. Rioual, J. Laurent, E. Senn and J. -P. Diguet, "Reinforcement Learning Strategies for Energy Management in Low Power IoT," 2017 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2017, pp. 1377-1382, doi: 10.1109/CSCI.2017.240.

[6] C. L. Sabharwal and B. Anjum, "Robo-Revolution in the Financial Sector," 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2018, pp. 1289-1292, doi: 10.1109/CSCI46756.2018.00249.

[7] A. J. Almalki and P. Wocjan, "Exploration of Reinforcement Learning to Play Snake Game," 2019 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2019, pp. 377-381, doi: 10.1109/CSCI49370.2019.00073.

[8] D. Asher, M. Garber-Barron, S. Rodriguez, E. Zaroukian and N. Waytowich, "Multi-Agent Coordination Profiles through State Space Perturbations," 2019 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2019, pp. 249-252, doi: 10.1109/CSCI49370.2019.00051.

Classificação dos artigos de aprendizado por Reforço



Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 26 de out. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto
Gustavo Barbosa

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Para o *stage* 2, o time de Reinforcement Learning (RL) focou em estudar as formas de aplicação de RL em séries temporais por meio de trabalhos já publicados. Para encontrar os artigos científicos na área, foi feita uma busca dos trabalhos citados nas referências de alguns trabalhos lidos durante o *stage* 1 relacionados ao tema de séries temporais.

Além disso, em um primeiro contato com a ferramenta Parsifal, foi estudado um pouco sobre construção de strings de busca para encontrar artigos nas bibliotecas disponíveis. Dessa forma, foi realizada uma busca inicial por surveys que juntassem as áreas de RL e séries temporais para obter visão mais ampla dessa área, mas até o momento nenhum foi encontrado. Contudo, com o Parsifal foram obtidos artigos que propõem alguns métodos para uso RL e séries.

Dos artigos lidos até o momento, há métodos com propostas bem diferentes, por exemplo, alguns aplicam RL diretamente na série temporal, outros usam RL para escolher hiperparâmetros de modelos de forecasting, além daqueles que usam RL para combinar as previsões de vários modelos distintos. No documento abaixo é apresentado uma visão geral desses artigos lidos:

https://docs.google.com/document/d/1j0RlVg-zmyfADe4NR8WsVKRnhbms-FvT4K4L68YHFP0/edit?usp=sharing

Já os artigos completos podem ser encontrados na pasta do drive:

https://drive.google.com/drive/folders/1bSw9WaRtUBQ_W94DWSSMA248D0zrT9qO?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Como atividades para próximo *stage*, foi planejado:

- Estudar sobre os algoritmos mais conhecidos como PPO, DQN, DDPG, etc...
- Continuar procurando e estudando artigos sobre RL e séries temporais
- Começar a procurar por algum dataset para fazer testes

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Aprendizado por reforço e séries temporais

Este texto fornece uma visão geral dos artigos lidos pelos membros do time de Reinforcement Learning (RL) durante o *Stage 2*. O objetivo deste estudo é compreender melhor as metodologias utilizadas para aplicar RL no contexto de séries temporais. Dentre os métodos encontrados, há a aplicação de RL diretamente na série, uso de RL para selecionar os hiperparâmetros dos modelos e também abordagens em que RL é usado para combinar/selecionar os resultados de vários modelos de forecasting.

Analisar os padrões de dados temporais e tomar uma ação a partir disso é útil em diversos contextos. Por exemplo, no varejo, decisões como alterar o preço dos produtos e repor o estoque de acordo com a demanda de cada produto são medidas essenciais para aumentar o lucro obtido. É nesse cenário que Pavlyshenko, em [1], propõe um método utilizando Deep Q-Learning (DQN) para tomar tais decisões de acordo com o comportamento de vendas ao longo do tempo.

O primeiro problema abordado no trabalho [1], se trata justamente da determinação de uma estratégia de precificação para um produto a fim de maximizar as vendas. O ambiente nessa abordagem é representado por um modelo paramétrico que relaciona a quantidade de vendas com o aumento de preço, além disso uma série sintética que simula demanda pelo produto é utilizada para representar o estado. Enquanto que as ações são discretizadas em incrementos fixos no preço. Assim, o agente analisa a demanda pelo produto ao longo do tempo e decide se vai aumentar ou diminuir o preço em questão. Os resultados mostram que o Agente treinado com DQN, acaba por encontrar um incremento ideal que aumenta o lucro obtido.

Ainda em [1], a segunda aplicação abordada é a de reposição de produtos por demanda, onde os testes foram realizados com o dataset do kaggle 'Rossmann Store Sales'. Nesse problema, o objetivo é decidir o quanto comprar de um produto para repor o estoque baseado na expectativa de vendas no futuro. Para modelar o estado, o autor utilizou features como a quantidade de vendas no dia anterior, se há ou não promoção ativa e uma representação one-hot do dia da semana. Já o espaço das ações foi limitado a 7 possíveis escolhas referente a quantidade que deverá ser comprada para suprir a demanda. Nesse caso, Pavlyshenko traz uma visão interessante dos resultados ao analisar o comportamento do agente em diferentes dias da semana.

Outra abordagem para o varejo, é tratada no artigo [6], o problema de otimização de estoque, o qual tem como objetivo aprimorar as operações de gerenciamento de estoque da empresa, assegurando o suprimento de produtos conforme a demanda, evitando desperdícios e os riscos associados ao excesso de estoque, ao mesmo tempo em que melhora o fluxo de caixa. Neste artigo, a abordagem Deep Q Network (DQN) é empregada para criar uma política que determine quando e quanto encomendar determinado produto.

Seguindo os princípios do Processo de Decisão de Markov (MDP), o estado é definido como a combinação da posição de estoque (estoque atual mais o pedido futuro) e o

dia da semana. O espaço de ações (a_t) refere-se à quantidade de produtos que devem ser encomendados até o final do dia. A recompensa é calculada através da função r_t, a qual é definida como o mínimo entre a demanda do dia (d_t) e a quantidade de produtos pedidos (i_t), multiplicado pelo preço unitário de venda do produto (p), subtraindo o custo de manutenção por unidade por noite (h) multiplicado pelo indicador I(a_t > 0) (que assume o valor de 1 se a_t > 0 e 0 caso contrário), e subtraindo o custo fixo do pedido (f) acrescido do custo variável do pedido por unidade (v).

Assim, utilizando um conjunto de dados sintéticos, utilizando a DQN, conseguiu obter um lucro de 18,09% a mais que uma política clássica que diz que se o estoque ficar abaixo de um valor s, devemos fazer um pedido S unidades.

Uma dificuldade na modelagem de séries temporais é saber quais pontos do passado são mais úteis para realizar a predição de um ponto futuro, no artigo [2], os autores utilizam RL para tentar resolver esse problema. No trabalho, RL não é aplicado diretamente nas séries temporais, mas sim utilizado para realizar uma busca por hiperparâmetros de uma rede neural que irá melhor se adequar aos dados. O Reinforcement Learning based Dimension and Delay Estimator (RLDDE) é proposto para determinar dois hiperparâmetros: Dimension e Delay. O primeiro está relacionado ao tamanho da entrada da rede neural, enquanto o segundo dita quais pontos da série temporal serão escolhidos como features para realizar a previsão. O algoritmo de RL utilizado é Q-learning, o estado que o agente recebe é determinado pelo desvio padrão que é discretizado em diferentes intervalos e a ação escolhida é formada por dois números representando o Dimension e o Delay. O autor chega em resultados muito bons se comparado com os métodos disponíveis na época da publicação do artigo.

Há uma ampla variedade de métodos disponíveis para lidar com análise de séries temporais, incluindo abordagens baseadas em modelos estatísticos, técnicas de aprendizado de máquina e algoritmos de deep learning. No entanto, a escolha do modelo mais adequado que funcione bem em diferentes horizontes de previsão, ou seja, no curto, médio e longo prazo, pode ser um problema complexo. Assim, os artigos [4] e [5] entram para abordar essa questão.

Diferentes modelos tendem a apresentar melhor desempenho em períodos específicos da série temporal, o que torna a seleção do modelo apropriado uma tarefa complexa. Nesse contexto, a estratégia de Dynamic Ensemble surge como uma abordagem promissora. Essa estratégia envolve a seleção dinâmica dos melhores pesos para a previsão.

Os métodos de Dynamic Ensemble buscam propor políticas adequadas para a previsão de séries temporais probabilísticas, considerando as características de ponderação dos modelos para diferentes horizontes de previsão. Essa abordagem permite uma adaptação flexível, permitindo que os modelos mais apropriados sejam escolhidos de forma dinâmica, dependendo do contexto e das necessidades da previsão. Assim, o objetivo é fazer uma rápida adaptação dos pesos dependendo das características da série temporal, fazendo a pergunta “Qual o melhor modelo para prever o próximo valor da série temporal?”.

Nesse contexto, segundo os artigos, o Reinforcement Learning é utilizado porque consegue explorar melhor o espaço de ações para otimizar a política, além disso abordagens de Model-Free nos permitem aprender uma política puramente a partir dos dados registrados, sem conhecer a complexa dinâmica subjacente do sistema. O modelo utilizado nos artigos [4] e [5] foi um off-policy actor-critic, o DDPG. No artigo [4] ele foi utilizado baseado no Processo de Decisão de Markov, já em [5] ele foi utilizado pelo agente criado que é o RL Based Model Combination (RLMC).

Yu et al., em [3], segue uma linha de pesquisa parecida com os artigos [4] e [5], contudo o objetivo não é selecionar o melhor modelo e sim combinar as previsões feitas por eles. O método proposto é aplicado no campo financeiro para prever o preço das ações da bolsa de valores, um cenário bem complicado devido à alta volatilidade. Este trabalho consiste em estabelecer um pipeline para forecasting robusto, com pré-processamento das séries e integração de modelos de deep learning através do Q-learning. Na parte de pré-processamento, os autores decompõem a séries em vários componentes mais simples através de ferramentas estatísticas a fim de facilitar o aprendizado do modelo. Enquanto que na parte de modelagem, é utilizado 3 modelos de deep learning para fazer a previsão em cada um dos componentes: Deep belief Network (DBN), gated recurrent unit (GRU), long short-term memory network (LSTM).

Nesse cenário, RL é utilizado por meio do Q-learning para melhorar o resultado ao aprender a combinar as previsões dos três modelos de forma eficiente. Assim, o agente recebe como estado os parâmetros ajustados de todos modelos e como ação escolhe os pesos para ponderar cada uma das três previsões feitas em um único valor. Como resultado, o modelo proposto no artigo supera todos os outros métodos testados e atinge bons resultados mesmo nesse cenário desafiador onde há uma grande volatilidade no preço das ações.

Referências

[1] Pavlyshenko, B. M. (2022). Sales time series analytics using deep Q-learning. *arXiv preprint arXiv:2201.02058*.

[2] Liu, F., Quek, C., & Ng, G. S. (2005, July). Neural network model for time series prediction by reinforcement learning. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* (Vol. 2, pp. 809-814). IEEE.

[3] Yu, Y., Lin, Y., Hou, X., & Zhang, X. (2023). Novel optimization approach for realized volatility forecast of stock price index based on deep reinforcement learning model. *Expert Systems with Applications*, 120880

[4]FU, Y.; WU, D.; BOULET, B. Reinforcement Learning Based Dynamic Model Combination for Time Series Forecasting. Proceedings of the AAAI Conference on Artificial Intelligence,

[S. l.], v. 36, n. 6, p. 6639-6647, 2022. DOI: 10.1609/aaai.v36i6.20618. Disponível em: https://ojs.aaai.org/index.php/AAAI/article/view/20618. Acesso em: 25 oct. 2023.

[5]Yuhao Ding and Youngsuk Park and Karthick Gopalswamy and Hilaf Hasson and Bernie Wang and Luke. Dynamic ensemble for probabilistic time-series forecasting via deep reinforcement learning, 2023Disponível em: https://www.amazon.science/publications/dynamic-ensemble-for-probabilistic-time-series-forecasting-via-deep-reinforcement-learning. Acesso em: 25 oct. 2023.

[6]XIE, G. Reinforcement Learning for Inventory Optimization Series I: An RL Model for Single Retailers. Disponível em: <https://towardsdatascience.com/a-reinforcement-learning-based-inventory-control-policy-for-retailers-ac35bc592278>.

APÊNDICE 2

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 9 de nov. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto
Gustavo Barbosa

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

No *stage* 3, o time de Reinforcement Learning (RL) decidiu continuar a pesquisa por cenários de aplicação de RL em séries temporais. Depois de mais busca por artigos, foi encontrado aplicações dessas técnicas principalmente nas áreas do varejo e financeira. Na primeira, o contexto de aplicação foi para o gerenciamento de estoque, já no segundo foi aplicado para gerenciamento de portfólio de ações da bolsa de valores. Uma visão geral desses trabalhos podem ser vistos no documento:

https://docs.google.com/document/d/1JmlZR9NR5saIusiZrdv_DkMD1fZsqhfrrbK36W8ca98/edit?usp=sharing

Além disso, como descrito na parte de planejamento do *gate* 2, foi revisado alguns dos algoritmos usados nos artigos lidos até o momento, um breve resumo das principais características deles foi colocado no documento:

https://docs.google.com/document/d/1oa5gWbkHzEq18Cbe2h3_kkjkSQ_5w_6x7zeW0zCrKEE/edit?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para próxima semana, foi planejado:

- Estudar mais a fundo as métricas e funções de recompensa comumente utilizadas no cenário do mercado financeiro.
- Buscar por frameworks e repositórios/códigos, e datasets para o varejo e o financeiro

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Resumo dos artigos de aprendizado por Reforço e séries temporais

Uma das aplicações de *Reinforcement Learning* (RL) com dados temporais é encontrado no mercado financeiro, mais especificamente no trading. Nesse cenário, especialistas precisam observar os padrões passados da cotação dos ativos para tomar decisões assertivas sobre a compra ou venda a fim de gerar lucro. Dessa forma, com RL, uma abordagem para automatização deste processo é treinamento de um agente que aprenda sobre esse padrões do mercado interagindo com o ambiente capaz de tomar tais decisões.

Millea, em [1], realiza uma revisão bibliográfica sobre o uso de *deep Reinforcement Learning* (DRL) para o trading. Neste trabalho, é feito um apanhado geral sobre como DRL permitiu que os agentes conseguissem aprender mesmo de estados complexos e também em ambientes com um espaço de ação grandes. Millea deixa em evidência duas grandes vertentes em que DRL pode ser utilizado, a primeira é o uso de um único ativo da bolsa de valores e com base na cotação passada, o agente toma a decisão de vender, comprar ou segurar. Já na segunda opção, se trata do gerenciamento de um portfólio de ações, onde o agente decide qual será a proporção de cada ativo na carteira do investidor, ou seja, é um espaço de ações contínuo.

Na linha da segunda opção, o artigo [2] propõe uma forma de lidar com problema de gerenciamento de portfólio a partir do histórico de cotação do ativo bem como outras características como liquidez e dividendos pagos. O autor escolheu para compor o ambiente de aprendizagem 415 ações das 500 presentes no índice S&P 500 cotado na bolsa americana. Cada estado era representado por um tensor de dimensão (d, n, f) onde d é quantidade de dias (pontos das séries), n representa o número de ações e f a quantidade de features (cotação, liquidez, dividendo, etc…). O autor decidiu limitar o agente a escolher no máximo 20 ativos para colocar na carteira, sendo assim, o espaço de ações são pesos contínuos de cada ativo cuja soma resulta em 1. O algoritmo utilizado não é especificado, porém é dito que se trata de um *model-free* e *policy gradient*.

O agente foi treinado nos dados de 1998 à 2011, realizou validação de 2012 a 2013 e os testes ficaram para o período de 2014 até 2018. O autor obteve resultados promissores se comparado com outras estratégias de gestão de carteira, contudo, apesar do lucro obtido, é ressaltado que de todas as estratégias, a do agente foi aquela mais arriscada passado por altos e baixos expressivos. Além disso, como nenhuma empresa que faliu foi escolhida para compor a base de treino, é bem possível que o comportamento mais arriscado do agente seja devido a esse viés. Portanto, os resultados de [2] são animadores, mas podem não servir para um investidor mais conservador.

Por outro lado Yu, P. et al, em [3], foca também na otimização de portfólio, mas a abordagem nesse caso foi *model-based*. O autor utiliza uma modelagem bem mais elaborada, onde há um modelo de forecasting que é responsável entender a dinâmica do ambiente e realizar a predição do próximo valor da ação. Além disso, no trabalho é

ressaltado a insuficiência na quantidade dos dados para treinar o agente, por isso o autor também propõe usar redes neurais generativas adversárias (GAN) para aprender a gerar amostras sintéticas. Por fim, Yu et al também sugere o treinamento e uso de um modelo que imitasse o comportamento de um especialista para gerar experiências adicionais ao agente. No estado do algoritmo é representado pela saída de um rede LSTM aplicada às 10 horas anteriores de todas as 8 ações escolhidas. Nesta representação também é concatenada com a ação anterior juntamente com a previsão do modelo de forecasting. Assim, como em [2], nesse trabalho o espaço de ação é formado pelos pesos que a gente decide dar a cada ativo da carteira. Os resultados pelo agente bateu o baseline estabelecido, além de que o desempenho dele não foi tão volátil como em [2].

Fora do contexto financeiro, há também aplicações de RL e séries temporais no cenário de varejo também como o gerenciamento de estoque. É nesse sentido que o artigo [4] propõe um sistema de controle de inventário, que visa determinar o nível ideal de estoque para diferentes produtos, equilibrando o custo de atender à demanda do cliente com o custo de manter muito estoque. O sistema ajusta o nível de estoque de forma periódica, adquirindo mais estoque ou removendo o estoque existente conforme necessário. Além disso, lida com alguns problemas como a demanda que é uma variável aleatória com dinâmicas desconhecidas(apresentando sazonalidade, tendência, picos), atrasos na chegada de produtos após um pedido e preocupações com a correspondência de preços por concorrentes.

As features utilizadas para representar o estado foram o nível atual de estoque, as ações anteriores tomadas até o tempo t, características relacionadas à série temporal de demanda histórica ajustada, informações sobre a proximidade de feriados públicos, dados históricos de visualizações da página do produto, características estáticas do produto, como grupo de produtos, recursos baseados em texto da descrição do produto, marca, além de informações econômicas do produto, como preço e custo, lags, entre outros. E para ações é utilizado um espaço unidimensional de ações contínuas para cada produto para representar a demanda.

No artigo, é introduzido o algoritmo *DirectBackpropagation*, que é detalhadamente explicado ao longo do texto. Para avaliar seu desempenho em comparação com outros métodos de RL, foram selecionados os métodos *model-free*, A3C, SAC e ARS. Os resultados obtidos revelam que o algoritmo *DirectBackpropagation* mantém níveis mais elevados de recompensa, ao mesmo tempo em que emite um número menor de pedidos em comparação com os outros métodos. Em seguida, o ARS também apresenta um desempenho notável.Além disso, o artigo destaca a capacidade dos algoritmos de RL em gerar curvas de inventário que se assemelham de maneira razoável aos inventários reais.

Referências

[1] Millea, Adrian. "Deep reinforcement learning for trading—A critical survey." *Data* 6.11 (2021): 119.

[2] Huotari, Tommi, Jyrki Savolainen, and Mikael Collan. "Deep reinforcement learning agent for S&P 500 stock selection." *Axioms* 9.4 (2020): 130.

[3] Yu, Pengqian, et al. "Model-based deep reinforcement learning for dynamic portfolio optimization." *arXiv preprint arXiv:1901.08740* (2019).

[4] Madeka, Dhruv, et al. 'Deep Inventory Management'. arXiv [Cs.LG], 2022, http://arxiv.org/abs/2210.03137. arXiv.

Tabela com os principais conceitos de aprendizado por Reforço dos artigos lidos.

Artigo	Dados	Espaço de estados	Espaço de ações	Recompensa	Desempenho
Huotari T., Savolainen J., Collan M. - 2020	S&P 500	Dimensão: (d, n, f). Índices (f) de várias ações (n) a longo do tempo (d).	Contínuo. Peso de cada ação na carteira	Sharpe ratio	328% de retorno total um período do 1255 dias
Yu P. et al - 2019	8 ativos selecionados	saída do LSTM para 10 horas anteriores de 8 ativos+ ação anterior + forecasting do próximo valor da cotação + índice S&P 500	Contínuo. Peso de cada ação na carteira	logarítimo da taxa de retorno	Em 2 anos, obteve 16% de lucro totalizando $ 80,899 dolars
Madeka-2022	Dados semanais de um único Marketeplace, 104 semanas de treinamento, 19 validação	Nível atual de estoque, as ações anteriores tomadas até o tempo t, características relacionadas à série temporal de demanda, informações sobre a proximidade de feriados, dados históricos de visualizações da página do produto, características estáticas do produto, além de informações econômicas do produto, como preço e custo, além de lags	Contínuo. Quantidade de pedido para cada produto	Fluxo de caixa, receita menos temos de custo	Diminuição do estoque em 12% sem perder receita

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 16 de nov. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

No *stage* 3, foi estudado sobre o uso de Reinforcement Learning (RL) aplicado a séries temporais no contexto do mercado financeiro. Dessa forma, o agente recebe dados temporais históricos da cotação de ações e toma decisões tais como vender, comprar ou segurar as cotas. Dessa forma, durante o *stage* 4, o foco foi dado em estudar melhor as métricas para mensurar o lucro ou risco de um ativo mencionadas no survey lido no *stage* 3.

Segundo o próprio Survey, algumas dessas métricas são úteis no treinamento dos agentes de RL, pois podem ser utilizadas como função de recompensa para guiar o aprendizado. No link abaixo é disponibilizado uma descrição sobre o estudo feito dessas métricas:

https://docs.google.com/document/d/1-xymX7KVVSFVd2N3o1iPpnrDbNlmnz88syhMCVq_ZiY/edit?pli=1

Além disso, com relação à obtenção dos dados do mercado, pode ser realizado através da plataforma do Yahoo Finance com auxílio do pacote do python pandas_reader. Dessa maneira, as principais informações sobre o preço histórico dos ativos também usados pelos papers lidos no *stage* 3 estão disponíveis.

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para o próximo stage é planejado:

- Começar a implementação de algoritmos para arrumar os dados para o modelo.
- Começar a preparar o ambiente de teste (open-ai gym).

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Métricas para avaliar o desempenho do investimento

***Return on Investment* (ROI)**

O ROI é uma métrica bastante usada para calcular de forma rápida a rentabilidade de um investimento [8]. A fórmula é bem simples e se baseia a apenas no valor investido e no valor atual do investimento:

$$\text{ROI} = \frac{\text{Current Value of Investment} - \text{Cost of Investment}}{\text{Cost of Investment}}$$

Apesar de útil, o ROI possui algumas limitações, como não levar em consideração a volatilidade do investimento, ou seja, lucro e quedas no meio caminho. Dessa maneira, é uma métrica que ignora os dados temporais e foca só no ganho atual. Isso é problemático, pois duas ações, por exemplo, podem possuir um mesmo ROI, porém uma pode ter oscilado entre altos e baixos muitas vezes enquanto a outra teve apenas lucros constantes. Portanto, outras métricas mais robustas são necessárias para mensurar o risco e lucro de um ativo financeiro.

Sharpe ratio

O índice de sharpe foi criado pelo economista William F. Sharpe com intuito de comparar o desempenho de ativos semelhantes [1, 2]. A principal vantagem dela é levar em consideração não apenas o retorno obtido, mas também a volatilidade. Além disso, essa métrica compara o desempenho do ativo com um investimento totalmente seguro (sem risco de perda) como é o caso do tesouro direto, por exemplo.

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

Onde,

R_p é o retorno sobre o investimento feito no ativo
R_f é o retorno sobre o investimento livre de risco
σ_p é desvio padrão de R_p

Na fórmula, ao calcular a média, o histórico é levado em consideração, se comparado com cálculo apenas do ROI. Já a volatilidade é inserida por meio do desvio padrão. Dessa forma, é feita uma ponderação entre retorno do investimento e o risco que o mesmo carrega segundo sua oscilação. Assim, valores maiores de sharpe positivos indicam investimento com um bom retorno e seguros.

Existem algumas desvantagens dessa métrica, mascarar a volatilidade do ativo é uma delas, isso porque a oscilação depende da janela de tempo que se escolhe. Por exemplo, em geral, o valor do ativo varia menos anualmente do que mensalmente que, por sua vez, varia menos que diariamente [2]. Além disso, o movimento de preço para cima ou baixo tem o mesmo peso, assim ações em que o valor cresce aos poucos com certa consistência, mas que vez ou outra tem grandes quedas podem não ser detectadas pelo sharpe ratio

Sortino ratio

Sortino ratio, criado por Frank A. Sortino, também é uma forma de comparar dois ativos diferentes. Na cotação do preço do ativo há oscilações negativas que prejudicam o investidor, mas também oscilações positivas que os favorecem. Nesse sentido, o Sortino dá mais peso às mudanças negativas no retorno do ativo do que às positivas [1, 3]. A fórmula é semelhante ao do sharpe, porém o denominador é mudado para o desvio padrão apenas dos retornos que foram negativos.

$$\text{Sortino Ratio} = \frac{R_p - r_f}{\sigma_d}$$

Onde,

R_p é o retorno sobre o investimento feito no ativo
R_f é o retorno sobre o investimento livre de risco
σ_p é desvio padrão dos retornos negativos

Value at risk

É uma métrica que mensura o tamanho do risco de um ativo dentro de uma janela de tempo. Nesse sentido, o *Value at Risk* (VaR) informa uma perda potencial que um ativo pode ter com um certa probabilidade ou margem de confiança [1, 4, 5]. Por exemplo, se o VaR de uma carteira de investimentos é de R$ -1000 com um confiança de 95%, isso significa que há 95% de chance que o investimento tenha uma perda de até R$ -1000 dentro da janela de tempo em que métrica foi calculada.

Existem algumas formas de calcular essa estatística, dentre elas estão o método com dados históricos, variance-covariance e Monte-carlo. O método de dados históricos é o mais simples, pois utiliza os lucros e perdas passadas para realizar a estimativa de VaR. Supondo que temos um ativo com cotação diária com tamanho de M de dias, a fórmula abaixo pode ser utilizada:

$$\text{Value at Risk} = v_m \frac{v_i}{v_{i-1}}$$

Onde,

v_m é cotação mais recente no último dia
v_i é a cotação no dia i
v_{i-1} é cotação no dia i-1

Assim, podemos realizar o cálculo do VaR todos os dias, o que vai determinar o VaR único para representar esse ativo é o nível de confiança. Dessa forma, é desejado um nível de confiança de 90%, basta escolher o VaR que representa o décimo (100 - 90 = 10) percentil do conjunto [4].

O método variance-covariance é um método paramétrico onde é assumido que a distribuição do retorno do investimento segue uma distribuição normal, ou seja, não pode ser aplicado em todos os casos. Já o método do Monte Carlo também é preciso assumir uma distribuição para os dados e o VaR é encontrado por meio de diversas simulações de lucros e perdas baseado na distribuição assumida.

As vantagens do VaR são a sua facilidade de interpretação, pois resume tudo em único número. Além disso, há também a questão da possibilidade da aplicabilidade em diferentes investimentos para comparar os possíveis riscos. Contudo, existem também desvantagens, uma delas é a sensibilidade à janela de tempo escolhida para fazer cálculo, visto que o valor pode ser mascarado por um período não representativo do investimento. As suposições também assumidas erroneamente podem gerar resultados destoantes da realidade.

Turbulence Index

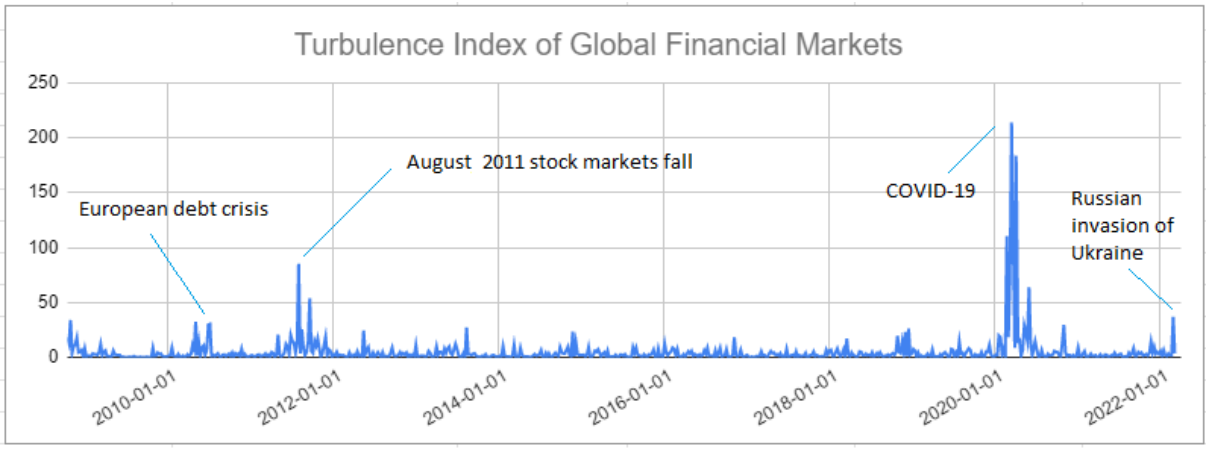
O índice de Turbulência é uma medida calculada a partir dos retornos históricos obtidos por um conjunto de ativos na carteira usada para identificar estresse no mercado financeiro [6]. Dessa forma, valores altos dessa métrica podem indicar possíveis eventos externos ao mercado que estão afetando os retornos dos investimentos seja positivamente ou negativamente. Abaixo está a fórmula utilizada para realizar calculado do *Turbulence Index* [survey]:

$$turbulence_t = (\mathbf{x}_t - \mu)\boldsymbol{\Sigma}^{-1}(\mathbf{x}_t - \mu)^T$$

Onde,

x_t representa os retornos de cada ativo do período t que se deseja analisar
μ é o retorno médio de cada ativo
Σ é a matriz de covariância dos ativos

Na fórmula, tanto μ quanto Σ são calculados em período de referência dos ativos diferente de t que reflita o seu comportamento padrão. Em [6], é feito um experimento interessante onde é calculado o *Turbulence Index* de algumas ações durantes os anos de 2009 à 2022:



É interessante notar que a métrica tem picos justamente em momentos em que houve algum evento ou crise externa que afetou o mercado financeiro. Dessa maneira, essa métrica é um indicativo útil para definir se é ou não um bom momento para investir um grande volume financeiro nos ativos devido a instabilidade do mercado.

***Maximum Drawdown* (MDD)**

Essa métrica é outra forma para calcular o risco de um ativo, ela considera a maior queda percentual no valor do ativo dentro de uma janela de tempo especificada [1, 7]. Ela pode ser calculada por meio da fórmula:

$$MDD = \frac{Trough\ Value - Peak\ Value}{Peak\ Value}$$

Dessa maneira, é considerado na fórmula o maior queda de um pico no gráfico para um vale, antes que um pico maior seja alcançado. A métrica é simples de ser calculada e útil para fazer análise de risco, mas possui limitações como não levar em consideração a frequência das quedas do investimento.

Referências

[1] Millea, Adrian. "Deep reinforcement learning for trading—A critical survey." *Data* 6.11 (2021): 119.

[2] Fernando, J. (n.d.). Sharpe ratio: Definition, formula, and examples. Investopedia. https://www.investopedia.com/terms/s/sharperatio.asp

[3] KENTON, W. (n.d.). Sortino Ratio: Definition, Formula, Calculation, and Example. https://www.investopedia.com/terms/s/sortinoratio.asp

[4] Corporate Finance Institute. *Value at risk (VAR)*. https://corporatefinanceinstitute.com/resources/career-map/sell-side/risk-management/value-at-risk-var

[5] KENTON, W. (n.d.). Understanding Value at Risk (VaR) and How It's Computed. https://www.investopedia.com/terms/v/var.asp

[6] Portfolio Optimizer. The Turbulence Index: Measuring Financial Risk https://portfoliooptimizer.io/blog/the-turbulence-index-measuring-financial-risk/

[7] HAYES, A. (n.d.). Maximum Drawdown (MDD) Defined, With Formula for Calculation. https://www.investopedia.com/terms/m/maximum-drawdown-mdd.asp

[8] Fernando, J. (n.d.). Return on Investment (ROI): How to Calculate It and What It Means. https://www.investopedia.com/terms/r/returnoninvestment.asp

APÊNDICE 3

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 23 de nov. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o Stage 4, foi encontrado uma API do Yahoo Finance que permitia acesso aos dados históricos de preços de vários ativos, inclusive os nacionais. Como o foco do estudo é o aprendizado por reforço, com esses dados, durante o Stage 5, foi feita a implementação de um ambiente onde os agentes que serão desenvolvidos possam atuar.

Para construir esse ambiente foi utilizado como estrutura base o padrão da OpenGym. Dessa forma, a lógica do código visa simular um ambiente de Trading onde observando os preços passados do ativo, os agentes possam interagir ao escolher a porcentagem do patrimônio atual que será investido no ativo com o foco em maximizar o retorno.

Como base para o desenvolvimento, foi utilizado o ambiente Gym Trading Env já implementado e bem completo. Contudo, devido a alguns problemas para executar esse ambiente, o código fonte foi estudado e uma versão mais simples foi feita, disponível em:

https://colab.research.google.com/drive/1v9XjZjvJSYLhcFJ442HpurHoKvPiVuX2?usp=sharing

Uma descrição mais detalhada do ambiente pode ser encontrado no documento:

https://docs.google.com/document/d/14VQprimbpDVYgyBsrwqNKn7-xIdIUSNwj_oOVPZ3kWA/edit?usp=drivesdk

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Como próximos passos para o Stage 6, é planejado:

- Testar algum algoritmo de aprendizado por reforço
- Melhorar o código da implementação do ambiente

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Descrição do Ambiente de simulação

Para o desenvolvimento de um ambiente de aprendizado voltado para Trading no mercado financeiro, a estrutura de classe utilizada para construção do código foi a da OpenAI Gym, como descrito em [1]. Nesse sentido, alguns métodos precisam ser implementados para seguir o padrão de um ambiente Gym, sendo o principal o método *step* ao qual permite que uma ação seja executada no ambiente e é onde toda a lógica de interação do agente com mercado será implementada.

Para implementar o método *step* e outros auxiliares, foi utilizado o ambiente proposto em [3] que segue o padrão do Gym e possui alta fidelidade, pois simula vários aspectos do mercado financeiro a fim de deixar o ambiente mais realista. Alguns desses aspectos são a taxa de transação cobrada pela corretora, operação de Short Selling, Margin trading, além de poder lidar com múltiplos ativos de uma só vez.

Devido a dificuldade para executar o código do ambiente citado disponível no repositório [2], foi feita a implementação de uma versão simplificada do ambiente com apenas os recursos básicos. O código desenvolvido até o momento durante o Stage 5, pode ser encontrado no notebook:
https://colab.research.google.com/drive/1v9XjZjvJSYLhcFJ442HpurHoKvPiVuX2?usp=sharing

No geral, o ambiente implementado permite a interação apenas com 1 ativo, recebendo um valor em reais inicial de patrimônio. Em seguida, a cada interação, é permitido que uma ação seja feita para modificar a porcentagem do patrimônio que será investido no ativo.

Espaço de observações

Em cada registro diário de um ativo há pelo menos 4 informações presentes: a de *open* que é referente a primeiro valor do ativo do dia, a de *high* que é maior valor do ativo no dia, o de *close* é o último valor do ativo no dia e o de *low* que representa o menor valor do ativo no dia. Esse conjunto de informações é conhecido como OHLC. Assim, nas observações que o ambiente retorna estão incluídos:

- Matriz no formato (w, 4), onde as 4 colunas representam o OHLC e w representa a quantidade de dias anteriores que serão utilizados na análise.
- Última ação tomada pelo agente, referente a proporção do portfólio em ações, também conhecido como posição.

Espaço de Ações

Nesta primeira versão do ambiente implementado, o agente irá controlar a proporção do patrimônio que será investido no ativo escolhido. Dessa forma, uma ação do agente é representada por um número decimal dentro do intervalo [0, 1] que indica a porcentagem do patrimônio (posição) que será investido no ativo ao qual o agente está interagindo no ambiente.

Com o formato dessa implementação fica fácil optar por um agente que toma ações contínuas ou aqueles que tomam ações discretas. Pois, basta apenas escolher algumas proporções fixas para serem escolhidas, por exemplo, poderia ser as 3 opções: [0, 0.5, 1].

Recompensas:

As recompensas implementadas até momento foram

- Logaritmo do retorno
- Sharpe ratio
- Sortino ratio

As duas por depender da divisão por desvio padrão, em alguns casos pode haver alguns erros, como divisão por 0. Assim, formas de contornar esse problema serão pensadas nos próximos Stages.

Referência:

[1] https://www.gymlibrary.dev/content/environment_creation/
[2] https://github.com/ClementPerroud/Gym-Trading-Env
[3] https://gym-trading-env.readthedocs.io/en/latest/

APÊNDICE 4

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 30 de nov. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o *Stage* 6, como planejado, foi feito um primeiro teste com um algoritmo de *Reinforcement Learning* (RL) para interagir com o ambiente de Trading desenvolvido no *Stage* 5. Para ajudar nessa tarefa, o *framework* ray-rllib foi utilizado, pois esse pacote do python já disponibiliza diversos algoritmos de RL com implementação confiável. Contudo, para usar o rllib foi necessário fazer algumas alterações e melhorias no código do ambiente de simulação de Trading.

O algoritmo testado foi *Proximal Policy Optimization* (PPO), nesse primeiro teste, por falta de experiência com o rllib, uma versão mais simples foi utilizada apenas com camadas na rede do tipo *Fully Connected*. A ação do Banco do Brasil foi escolhida para esse primeiro experimento, onde as cotações do ativo de 2000 até 2019 foram usados para treinamento e aquelas de 2020 em diante foram usadas para o teste.

Para a entrega dessa semana, foi feita uma atualização no documento disponibilizado na semana anterior, nele foi acrescentado alguns detalhes a mais do ambiente, as melhorias feitas e os primeiros resultados com o framework testado. O link disponível abaixo:

https://docs.google.com/document/d/1LLfzyg331PRq8UonEW4JWfpcdFjIzL-WuEkYij0aCZA/edit?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para os próximos passos é planejado:

- Estudar a documentação do ray-rllib a fim de ver viabilidade de utilizar redes mais complexas. Como: Convolucionais e RNNs.
- Melhorar o código de treinamento

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

Neste gate, o Professor Aldo André Díaz Salazar esteve na banca avaliadora substituindo a Professora Luana.

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Em análise!

Uso do RLlib no ambiente de simulação

Em Reinforcement Learning, conceitos como ambiente e agente precisam ser estabelecidos para cada situação em que se deseja resolver um problema com as técnicas da Área [1]. Um esquema bastante comum para exemplificar a interação entre esses dois conceitos é apresentado na figura 1. O ambiente é um componente que visa simular a tarefa a qual deseja-se que o agente resolva, para tal, estabelece as possíveis ações (a) e observações (s) que representam os aspectos do ambiente em um momento específico. Enquanto que o agente é um algoritmo que irá decidir qual ação, dentre aquelas permitidas, será executada em cada interação com o ambiente. Para guiar o aprendizado, o ambiente fornece uma recompensa (r) para indicar o quão boa foi a ação tomada pelo agente no momento em questão. [2]

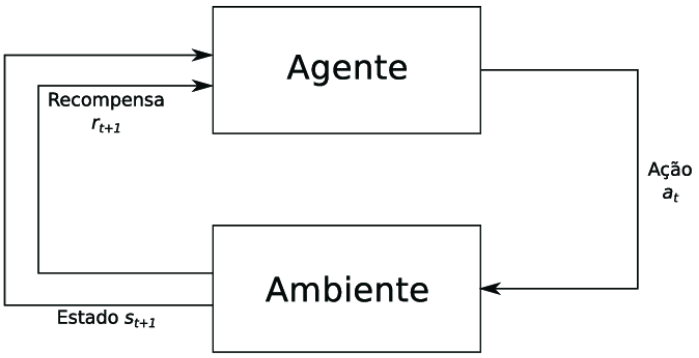


Figura 1: Representação geral dos problemas de RL

1. Ambiente de Trading

No contexto da aplicação no mercado financeiro, um ambiente também precisa ser desenvolvido para possibilitar que os algoritmos de RL (agentes) possam aprender a operar na bolsa de valores. Desta forma, um ambiente que simula uma plataforma de trading foi desenvolvido, como mencionado no Gate 5.

Para facilitar o desenvolvimento, foram consideradas 3 funcionalidades presentes em um plataforma de trading real para ser simulado. As 2 primeiras funcionalidades são a possibilidade do investidor/usuário da plataforma poder realizar operações de compra ou venda um ativo financeiro (ações permitidas pelo ambiente). Já a outra funcionalidade se refere a capacidade da plataforma de mostrar os preços atuais e passados do ativo (observações) ao investidor/usuário. Com isso em mente, essas 3 características foram utilizadas na implementação de um ambiente de simulação simples para contexto de trading por meio do padrão estabelecido pelo Farama Foundation Gymnasium.

Em relação ao agente, os algoritmos de RL, o pacote ray-rllib foi escolhido para ser estudado e utilizado para treinar agentes que irão interagir com o ambiente. Esse framework foi escolhido pois possui implementações confiáveis dos principais algoritmos de RL, além de ser bem conhecido na área. Dessa maneira, o agente irá escolher entre as operações de comprar ou vender ações baseado nos preços passados do ativo a fim de maximizar o lucro obtido (recompensa).

Maiores detalhes do ambiente sobre o formato das observações, ações e recompensas são descritos abaixo:

1.1 Espaço de observações

Em cada registro diário de um ativo há pelo menos 4 informações presentes: a de *open* que é referente a primeiro valor do ativo do dia, a de *high* que é maior valor do ativo no dia, o de *close* é o último valor do ativo no dia e o de *low* que representa o menor valor do ativo no dia. Esse conjunto de informações é conhecido como OHLC. Assim, nas observações que o ambiente retorna estão incluídos:

- Matriz no formato (w, 4), onde as 4 colunas representam o OHLC e w representa a quantidade de dias anteriores que serão utilizados na análise.
- Última ação tomada pelo agente, referente a proporção do portfólio em ações, também conhecido como posição.

Contudo, nesses primeiros testes, apenas o valor de *close* foi considerado para fins de simplicidade. Além disso, a posição atual do agente é acrescentada à observação por meio do one-hot encoding.

1.2 Espaço de Ações

Nesta primeira versão do ambiente implementado, o agente irá controlar a proporção do patrimônio que será investido no ativo escolhido. Dessa forma, uma ação do agente é representada por um número decimal dentro do intervalo [0, 1] que indica a porcentagem do patrimônio (posição) que será investido no ativo ao qual o agente está interagindo no ambiente.

Com o formato dessa implementação fica fácil optar por um agente que toma ações contínuas ou aqueles que tomam ações discretas. Pois, basta apenas escolher algumas proporções fixas para serem escolhidas, por exemplo, poderia ser as 3 opções: [0, 0.5, 1].

Para simplificar os testes iniciais, o ambiente só aceitará executar valores de posição de 0 ou 1. Ou seja, o agente terá que ou investir de uma só vez todo patrimônio no ativo financeiro ou vender os todos ativos que tem na carteira.

1.3 Recompensas:

As recompensas implementadas até momento foram

- Logaritmo do retorno
- Sharpe ratio
- Sortino ratio

As duas por depender da divisão por desvio padrão, em alguns casos pode haver alguns erros, como divisão por 0. Dessa maneira, formas de contornar esse problema serão pensadas nos próximos Stages. Assim, nesse primeiro teste, foi utilizado o logaritmo do retorno.

2. Dados utilizados

Para implementação do ambiente em questão é necessário ter os dados históricos de algum ativo do mercado pelo menos. Por isso, os valores das ações do Banco do Brasil foram obtidos da API do Yahoo Finance referente ao período de 01/01/2000 até 30/12/2019 para realizar o treinamento do agente. Na figura 2, é possível visualizar uma amostra dos dados obtidos dessa API.

Date	Open	High	Low	Close	Adj Close	Volume
2000-01-03	2.800000	2.853333	2.733333	2.733333	1.081363	514800
2000-01-04	2.733333	2.740000	2.603333	2.603333	1.029932	314100
2000-01-05	2.600000	2.633333	2.506666	2.630000	1.040482	478800
2000-01-06	2.630000	2.666666	2.600000	2.666666	1.054989	205200
2000-01-07	2.666666	2.666666	2.606666	2.606666	1.031251	394200
...	...	...	...	...	...	...
2019-12-20	50.950001	51.099998	50.240002	51.060001	39.521938	9537300
2019-12-23	50.959999	52.599998	50.740002	52.599998	40.713940	8803400
2019-12-26	52.480000	53.259998	52.349998	53.200001	41.178364	10031600
2019-12-27	53.349998	53.669998	52.259998	52.970001	41.000343	8633100
2019-12-30	53.099998	53.340000	52.820000	52.820000	40.884228	7839900

Figura 2: Histórico de cotações do Banco do Brasil na bolsa de valores

O ambiente simulação não irá usar todos esses dados de uma só vez, ao invés disso, as amostras serão quebradas em janelas de tamanho fixo de dias. No caso dos testes iniciais feitos esse tamanho será de 15 dias, mesma quantidade usada no artigo [3]. Contudo, a fim de deixar o ambiente mais dinâmico e diferente a cada execução, toda vez que é inicializado, o ambiente escolhe um dia aleatório como início da série temporal. Assim, a partir do dia escolhido, o agente irá interagir durante os próximos 300 dias seguintes, ao completar essa marca o ambiente é reinicializado em outro dia aleatório. Os 300 dias de

interação entre o agente e ambiente é chamado de episódio, e a cada reinicialização do ambiente, um novo episódio começa.

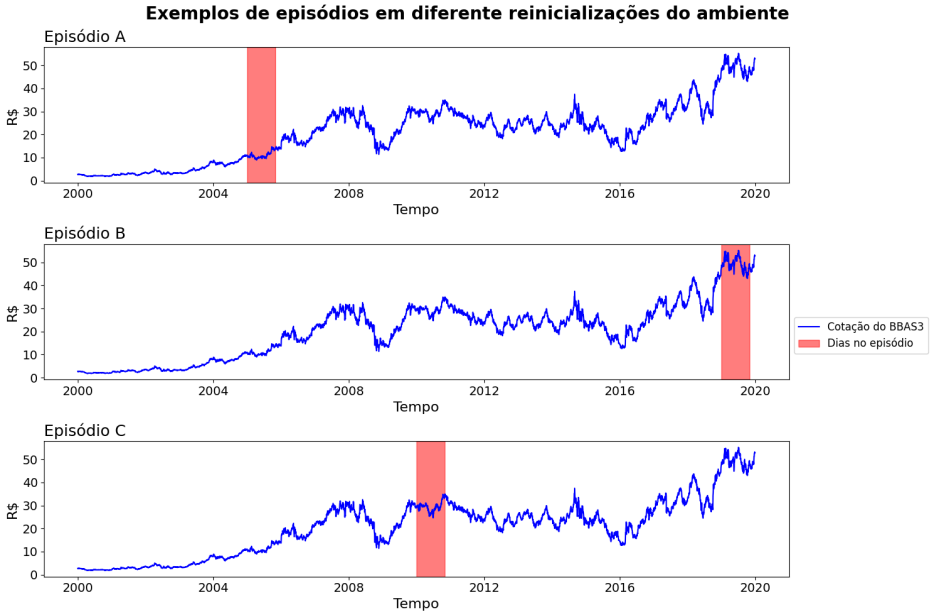


Figura 3: Exemplos de faixas de tempo usadas por diferentes episódios.

3. Resultados com rllib

O algoritmo de RL utilizado no rllib para interagir e aprender com o ambiente de trading foi o *Proximal Policy Optimization* (PPO). A configuração padrão de hiperparâmetro do PPO no framework foi utilizada para treinamento e o critério de parada utilizado foi o limite de 200000 interações do agente com o ambiente. A arquitetura da rede neural é uma *Fully Connected,* mas nos próximos *Stages* será analisada a possibilidade de testar outras opções mais robustas, como redes convolucionais e RNNs dentro do rllib.

Em relação a divisão para treino e teste, os dados de cotação do banco do brasil de 01/01/2000 até 30/12/2019 foram utilizados para o treinamento, já o período a partir de 2020 foi utilizado como teste. Como comparação com o desempenho do agente treinado foi escolhida 2 outras estratégias para gerenciamento da carteira de investimentos. A primeira foi a estratégia ingênua que apenas analisa a média dos últimos 15 dias com relação à cotação atual do ativo, se o preço do ativo estiver maior que a média ele compra, caso esteja menor, os ativos são vendidos. A segunda estratégia é a aleatória, onde é escolhido entre operações de venda e compra a cada dia aleatoriamente. Na figura 4, os resultados podem ser visualizados.

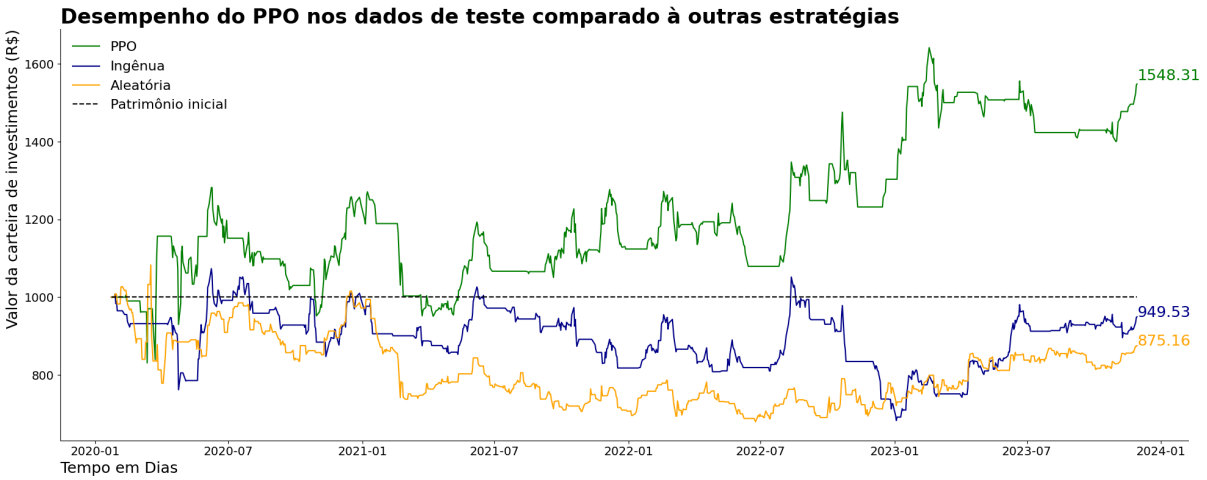


Figura 4: Desempenho da estratégia de investimento desenvolvido pelo PPO

Como pode ser observado, o PPO obteve resultados superiores às outras duas abordagens mais simples implementadas, onde patrimônio saiu de R$1000 reais inicialmente para R$1548, um retorno de 54.8%. Contudo, apesar dos primeiros resultados animadores, vale lembrar que esse experimento com PPO foi feito apenas uma vez, para resultados mais confiáveis é necessário realizar mais execuções. Além disso, antes de tirar as conclusões precipitadas é necessário testar o algoritmo em outras ações mais voláteis, visto que a ação do Banco do Brasil parece não ter variações muito bruscas que possam testar o algoritmo em situações extremas.

O código pode ser encontrado no github:
https://github.com/LuisAugusto0205/trading_environment/tree/main

Referências

[1] Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
[2] Data Science Academy. Deep Learning Book, 2022. Disponível em: <https://www.deeplearningbook.com.br/?s=64>.
[3] Huotari, Tommi, Jyrki Savolainen, and Mikael Collan. "Deep reinforcement learning agent for S&P 500 stock selection." *Axioms* 9.4 (2020): 130.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 7 de dez. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Durante o Stage 7, foi continuado o estudo sobre aplicação de Reinforcement Learning em dados temporais das cotações em um ambiente de simulação de trading. Como planejado, o código para treinamento do agente com framework ray-RLlib foi melhorando. A principal vantagem dessa nova versão do código é a facilidade para realizar testes, pois agora é possível mudar diversos parâmetros do código via linha de comando. Além disso, foi dado foco para o estudo da documentação do RLlib para encontrar maneiras de personalizar o modelo dentro da política usada pelo agente. Nesse sentido, a nova versão código permite a customização de tais modelos. Até o momento, um modelo convolucional foi desenvolvido, mas ainda não está finalizado.

Link para o documento de entrega do Stage 7, detalhando melhor o que foi desenvolvido:

https://docs.google.com/document/d/1SMkmrngTV3o1YS4OZVim6QpcqJmmeP6pJpEf03BKFv8/edit?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para os próximos passos é planejado:

- Realizar testes com diferentes ações
- Corrigir problema com modelo convolucional

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

Neste gate, o Professor Aldo André Díaz Salazar esteve na banca avaliadora substituindo a Professora Luana.

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Em análise!

Testes com RLlib e organização do repositório

ray-RLlib

O framework do python ray-RLlib possui implementado diversos algoritmos de Reinforcement Learning que podem ser utilizados para diversos ambientes diferentes que sigam a interface do Gymnasium. Além de permitir a customização do ambiente de simulação e algoritmos de aprendizado, outra vantagem dessa ferramenta é a possibilidade de paralelização, o que agiliza a etapa de treinamento.

Para realizar esse paralelismo, o RLlib utiliza o conceito de WorkerSet (figura 1) para criar um Worker em que cada núcleo do computador Um Worker tem uma cópia do ambiente e do agente, o que permite que ele faça uma simulação separadamente. Dessa forma, é possível coletar amostras para o treinamento mais rápido através das simulações feitas paralelamente em cada núcleo do processador. [1]

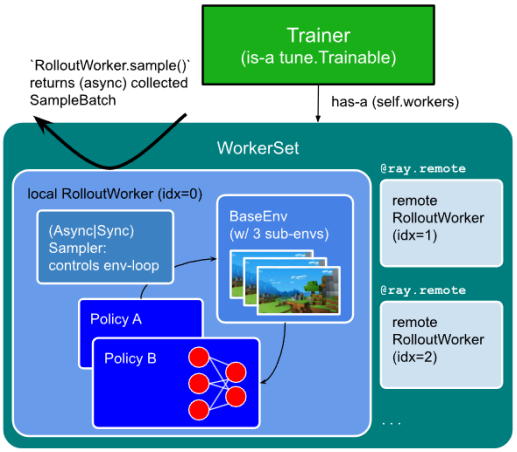


Figura 1: Componentes presentes em experimento do RLlib

Dentro de um RolloutWorker, há sempre uma política (como PPO, DQN, DDPG, entre outras) em que há sempre um modelo responsável por mapear uma observação para uma ação, representado em vermelho na figura 1. O modelo pode ser customizado a fim de seguir alguma arquitetura específica (Fully Connected, Convolutional ou RNN por exemplo). No contexto do ambiente de Trade, a política utilizada foi PPO com uma versão de rede neural totalmente conectada e uma versão convolucional. Contudo, essa última ainda está em desenvolvimento.

Github

O repositório do github do projeto foi atualizado, onde o código de treinamento e avaliação foram melhor estruturados. Nele, há os arquivos TrandEnv.py, main.py e um diretório models. O TrandEnv.py é o arquivo no qual o ambiente de simulação está implementado seguindo os padrões do Farama-Foundation/Gymnasium. Já o arquivo main.py é onde o framework ray-RLlib é aplicado para treinar o agente que irá interagir com o ambiente. O diretório models é onde são armazenados os modelos customizados implementados seguindo os padrões exigidos do RLlib. Há apenas o modelo convolucional, pois o modelo FullyConected já está implementado internamente dentro do RLlib.

Para configurar um experimento, agora é possível customizar os parâmetros do treino por meio da linha comando que são:

- **ticket:** Representa o nome do ativo registrado na plataforma Yahoo Finance os quais os dados serão baixados
- **train_time:** Intervalo de tempo no qual o treinamento será realizado
- **eval_time:** Intervalo de tempo no qual a avaliação será realizada
- **patrimony:** Valor inicial para investir
- **positions:** Lista com possíveis posições na carteira
- **max_steps:** Critério de parada pelo número máximo de iterações com ambiente
- **window_size:** Quantidade de dias anteriores em cada observação
- **epi_len:** Quantidade de passos antes que o ambiente seja reinicializado.
- **n_times:** Quantidade de vezes que experimento será executado
- **model:** Modelo que será utilizado dentro do algoritmo selecionado

Um exemplo de execução por meio da linha de comando seria:

```
python main.py --ticket BBAS3.SA --train_time 2000-01-01/2020-01-01
--eval_time 2020-01-01/2023-12-01 --patrimony 1000 --position 0, 1
--model FullyCon
```

Ao executar o comando acima na pasta principal do projeto, um agente será treinado por meio do algoritmo PPO com um modelo *Fully Connected* para tomar as decisões. O agente será treinado para operar com as ações do Banco do Brasil durante o período de 2000 até 2020, a avaliação será feita a partir de 2020 em dados que o agente nunca viu. O valor que o agente terá que operar é de R$1000 reais como especificado.

Link do repositório:
https://github.com/LuisAugusto0205/trading_environment

Resultados

No trabalho de Théate e Ernst, em [2], foi treinado um agente para apenas com uma ação da bolsa americana semelhante ao que está em desenvolvimento neste projeto da residência. Nesse artigo, os autores fazem experimentos com várias ações da bolsa, treinando um agente separadamente para cada uma delas. Para fins de comparações futuras, foi escolhida a ação americana da empresa Tesla (TSLA) para fazer o teste no código desenvolvido com ray-RLlib. O período de treino e avaliação foram os mesmos do artigo: 2012 a 2017 para treino e 2018 a 2019 para teste.

O comando para rodar o experimento foi:

```
python main.py --ticket TSLA  --train_time 2012-01-01/2017-12-31
--eval_time 2018-01-01/2019-12-31 --positions 0, 1 --max_steps 200000
--n_times 5 --model FullyCon --epi_len 300
```

Portanto, o experimento foi repetido 5 vezes, com modelo *Fully Conected* com política do PPO, onde cada episódio durou 300 passos. Além do mais, as ações (posições) possíveis do agente são vender todos os ativos (0) ou comprar o máximo de ativos que conseguir (1). A média do valor da carteira do Agente durante o período de avaliação pode ser visualizado na figura 2.

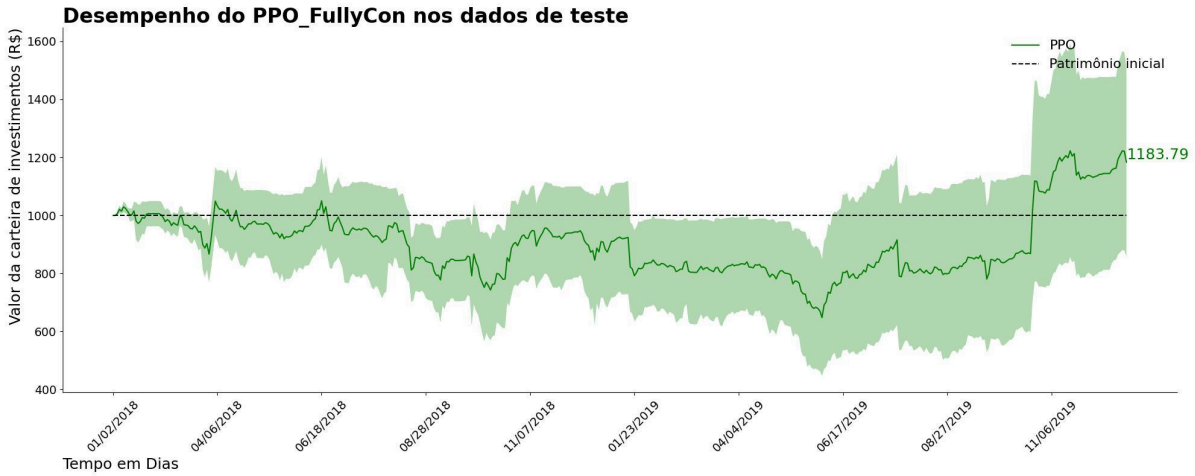


Figura 2: Valor da carteira do agente que operou a ação americana da empresa Tesla.

Como pode ser constatado, o desempenho médio dos nas 5 repetições durante quase todo período de avaliação foi ruim, abaixo do valor inicial da carteira, mas ao final obteve um retorno positivo em 3 dos 5 experimentos realizados. Além disso, o comportamento de escolha de ação do agente que obteve mais lucro pode ser analisado na figura 3.

Figura 3: Ações escolhidas pelo agente durante o período de avaliação, em azul são os momentos em que o agente está com toda carteira investida (ação 1), já o vermelho o agente não está em posse de nenhuma ativo (ação 0).

Analisando a figura 3, é possível perceber que o agente toma algumas decisões que realmente fazem sentido. Por exemplo, ao final (28/06/2019 e 07/11/2019) do período de avaliação o agente resolveu investir quando o preço da ação estava prestes a subir, o que gerou um retorno positivo para o valor da carteira.

Referências

[1]https://docs.ray.io/en/latest/rllib/package_ref/evaluation.html#rolloutworker-reference-docs

[2] Théate, Thibaut, and Damien Ernst. "An application of deep reinforcement learning to algorithmic trading." *Expert Systems with Applications* 173 (2021): 114632.

APÊNDICE 5

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 14 de dez. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Nos últimos Stages não foram obtidos resultados satisfatórios com Reinforcement Learning (RL) aplicados aos dados temporais do mercado financeiro. Por esse motivo, durante a apresentação do gate 7, foram recebidos alguns feedbacks sobre possíveis formas de melhorar o desempenho dos agentes desenvolvidos. Uma das maneiras, citada pelo professor Federson, é a de verificar se os dados passados para o agente tem o conhecimento útil necessário para que o mesmo seja capaz de tomar ações relevantes. Nesse sentido, durante o Stage 8, foi dado enfoque para o estudo de indicadores do mercado financeiro que possam ser utilizados como features de entrada para o agente a fim de facilitar o aprendizado.

Link para o documento de entrega do Stage 8, detalhando melhor os indicadores estudados:

https://docs.google.com/document/d/1U8iu24E52hMCWI4hdKxZs7id9-A3SqWU7nrF_VCbCVY

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para os próximos passos é planejado:

- Implementação das features estudadas
- Correção de erros ou troca de framework utilizado

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Estudo dos indicadores do mercado financeiro

Mean Reversion

É um fenômeno interessante que ocorre com algumas séries temporais que não apresentam tendência muito forte. A ideia é que se o valor atual estiver acima da média histórica, o valor da série irá voltar em algum momento para o valor médio. A mesma ideia vale para quando o valor está abaixo da média [1].

Dessa forma, alguns investidores usam o Mean Reversion para elaborar estratégias automáticas de trade. Caso os valores do preço do ativo assumam uma distribuição normal, uma dessas estratégias consiste em usar o Z-score. Os passos para cálculo são:

1. Primeiro centraliza-se o valores em torno de 0, subtraindo o valor médio dos preços
2. Com os valores centralizados, calcula-se o desvio padrão
3. Divide-se os valores centralizados pelo desvio padrão

Seguindo esses passos é possível obter o Z-score para todas as cotações diárias. Nos dias em que o Z-score está acima de um limiar pré-definido, pode-se tirar algumas conclusões sobre o ativo. Normalmente o limiar utilizado é de 1.5 a 2 e -1.5 a -2, caso o valor do ativo esteja dentro do limite superior é dito que ele está supervalorizado e pode ser um bom momento para venda, enquanto que abaixo do limite inferior indica uma subvalorização e um bom momento para compra.

Momentum

É um indicador do mercado financeiro que indica o quanto o preço de um ativo tende a continuar subindo ou descendo, ou seja, mensura a taxa de aceleração do preço [2]. Esse tipo de indicador permite ao investidor a ter um base de qual comportamento futuro esperar o cotação preço, por isso é pode ser utilizada para tomar decisões de compra ou venda de ativos.

Relative Strength Index (RSI)

É uma dos indicadores de momentum criado por J. Welles Wilder Jr., o RSI varia de 0 a 100. Dessa forma, é definido alguns valores limites dessa métrica que quando ultrapassados podem indicar uma boa oportunidade de compra ou venda. O RSI é calculado utilizando um período passado, normalmente o valor é 15 pontos passados [3]. A fórmula para cálculo do RSI é apresentada abaixo:

$$RSI_{\text{step one}} = 100 - \left[\frac{100}{1 + \frac{\text{Average gain}}{\text{Average loss}}}\right]$$

onde,

Average gain: é média dos ganhos positivos (negativos são zerados)

Average loss: é o valor absoluto da média dos ganhos negativos (positivos são zerados)

Os valores normalmente utilizados como limite inferior ou superior são 30 e 70, respectivamente. Nesse sentido, cotações que superem 70 de RSI são considerados ativos que estão acima do preço verdadeiro e, por isso, se configura uma boa oportunidade para venda. Na lógica inversa, se a cotação em período de tempo especificado cair abaixo de 30 de RSI, o ativo é considerado estar abaixo do seu preço real indicando uma boa oportunidade de compra. É importante ressaltar que esse limites podem ser alterados.

Moving Average Convergence/Divergence (MACD)

É outro indicador do momentum que relaciona duas exponential moving average (EMA) de curto a longo prazo com tamanho de 12 e 26 dias anteriores. O MACD é calculado como diferença entre as médias de curto e longo prazo, além disso uma outra média móvel ponderada é calculada para fins comparação é o Sinal por meio dos 9 dias anteriores [4]. A fórmula utilizada no cálculo é:

$$\text{MACD} = \text{12-Period EMA} - \text{26-Period EMA}$$

A visualização, na figura 1, dessa métrica pode ser feita tanto por gráfico de linhas e a diferença entre o MACD e Sinal é analisado como histograma.

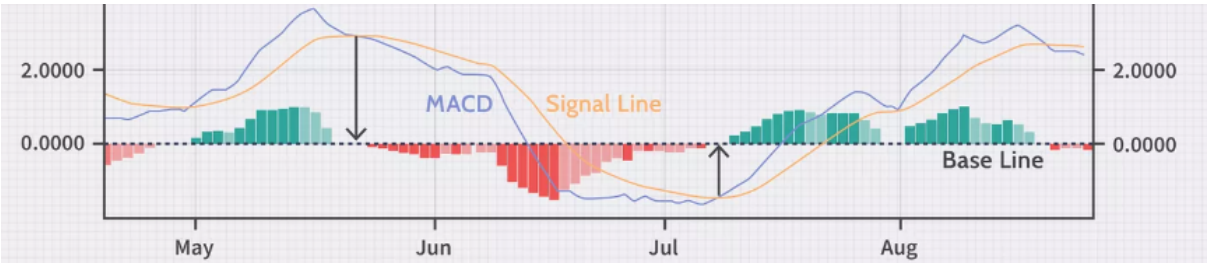


figura 1: Visualização do MACD e Sinal

Nos momentos em que MACD cruza a linha de Sinal pode indicar ou movimento de alta ou de queda. Quando MACD fica menor que o Sinal, o histograma fica vermelho e indica uma tendência de queda. Na situação contrária, o histograma fica verde e indica tendência de alta.

Stochastic Oscillator

Também é um indicador de momentum do mercado financeiro criado por George Lane em 1950. Similar ao RSI varia de 0 a 100 e pode dar insights de momento para realizar a ações compra e venda de um ativo. Contudo, o Stochastic Oscillator é calculado por meio dos valores mais altos e baixos atingidos pela cotação em uma janela de tempo, normalmente 14 dias [5]. Esse indicador pode ser calculado pela fórmula:

$$\%\mathrm{K} = \left(\frac{\mathrm{C} - \mathrm{L14}}{\mathrm{H14} - \mathrm{L14}} \right) \times 100$$

onde,
%K: é valor do indicador Stochastic Oscillator
L14: é o preço mais baixo dos últimos 14 dias
H14: é o preço mais alto dos últimos 14 dias
Valores de %K maiores que 80 indicam que o preço do ativo está acima do seu preço real. Enquanto que valores de %K menores que 20 é um indicativo de que o valor do ativo está abaixo do preço real. Normalmente, esse indicador é comparado com a média móvel simples dos últimos 3 dias.

Referências:

[1] https://www.investopedia.com/terms/m/meanreversion.asp
[2] https://www.investopedia.com/terms/m/momentum.asp
[3] https://www.investopedia.com/terms/r/rsi.asp
[4]https://www.investopedia.com/terms/m/macd.asp#toc-what-is-moving-average-convergencedivergence-macd
[5]https://www.investopedia.com/terms/s/stochasticoscillator.asp#toc-history-of-the-stochastic-oscillator

APÊNDICE 6

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 21 de dez. de 2023

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

No Stage 9, foram feitas alterações no código a fim de melhorar o desempenho dos algoritmos de Reinforcement Learning (RL) aplicados nos dados temporais das cotações de uma ação da bolsa de valores. Para tal, foram implementadas as sugestões feitas durante o Gate 7 e Gate 8 dadas pelo professor Federson e a Luana. As mudanças feitas são:

- Alteração do framework utilizado para treinar os agentes de RL e mudança do algoritmo de Proximal Policy Optimization (PPO) como sugestão da Luana, pois além de difícil otimização, o mesmo não havia dado bons resultados nos testes anteriores feitos. Portanto, agora foi utilizado outras implementações disponibilizadas no Github do algoritmo Deep Q-learning Network (DQN).
- Os indicadores financeiros da bolsa apresentados no Gate 8 e descritos no documento de entrega do Stage 8 foram implementados neste Stage 9. Tal sugestão foi dada pelo professor Federson, onde a ideia central é facilitar o aprendizado do agente de RL ao passar alguns indicadores calculados que os próprios investidores utilizam na hora de operar na bolsa de valores.
- Alteração na função de recompensa utilizada pelo ambiente a fim de melhorar o feedback que o agente recebe pela decisão de compra ou venda tomada a cada momento.

As alterações feitas se mostraram muito positivas, visto que nos experimentos executados os agentes treinados conseguiram um desempenho melhor se comparado a testes feitos anteriormente. Os detalhes das mudanças e treinamento dos agentes podem ser vistos no documento gerado durante o Stage 9:

https://docs.google.com/document/d/19_-1rKqj3pC0yDMQ8h0MO-qoVwA6F6PaANcou4nea3c/edit?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Para os próximos passos é planejado:

- Testar outros algoritmos de RL como DDPG e ARS.
- Pensar em formas de melhorar ainda mais a função de recompensa.

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Teste com *Deep Q-learning* com os indicadores

Como apresentado no gate 7, às tentativas de aplicar Reinforcement Learning (RL) em dados de séries temporais do mercado financeiro não foram bem sucedidas até o momento do gate 7. Nesse sentido, foram apresentados alguns feedbacks pelos professores sobre possíveis estratégias para melhorar os resultados ruins dos agentes de RL. Dentre as sugestões dadas, estavam a de mudar o formato dos dados passados para o agente, onde a ideia é calcular indicadores utilizados pelos investidores humanos em cima dos dados do ativo financeiro ao invés de passar os dados brutos das últimas cotações de preço.

Seguindo a sugestão dada, no gate 8, foi estudado e apresentado algumas métricas do mercado financeiro, porém não haviam sido implementadas no ambiente até o momento dessa apresentação. Além disso, no final desse mesmo gate, outras sugestões foram dadas, como mudar o framework RLlib utilizado devido às dificuldades para resolver os problemas enfrentados com o aprendizado dos agentes testados. Dessa forma, durante o Stage 9 foi utilizado uma implementação de algoritmos Deep Q-learning Network (DQN) presentes no repositório do Github:

https://github.com/udacity/deep-reinforcement-learning/tree/master

O link para github do projeto com implementação das alterações no ambiente de trade e dos agentes está no repositório:

https://github.com/LuisAugusto0205/trading_environment

Features

Durante o Stage 9 foi realizado a implementação dos indicadores citados na apresentação do gate 8 comumente utilizado como heurísticas pelos investidores da bolsa para tomarem as suas decisões, mais detalhes sobre eles podem obtidos no documento Semana 8. Em resumo, foram 5 indicadores no total, sendo eles:

- **Mean Reversion (MR):** Assume a ideia de que a série temporal tende a voltar à média e mensura o tanto que o preço atual está longe da média. O cálculo envolve as 14 cotações passadas do ativo. Os valores absolutos acima de 1.5 podem indicar um bom momento de compra ou venda.
- **Relative Strength Index (RSI):** Forma de mensurar se o ativo está acima ou abaixo do preço ideal, o valor varia entre 0 e 100. Normalmente, valores abaixo de 30 indicam valor do ativo abaixo do preço ideal e acima de 70 indica que o preço está super valorizado. O cálculo envolve uma janela de 14 cotações passadas.
- **Moving Average Convergence/Divergence (MACD):** Podem indicar movimento de alta ou baixa do preço do ativo, calculado por meio das médias de 12 e 26 cotações passadas. Essa métrica é usada junto com a Signal MACD.

- **Signal MACD:** É a média ponderada dos últimos 9 valores do MACD calculados. Se o valor do Signal é maior que o MACD atual, pode indicar tendência de baixa, enquanto se Signal for menor, pode indicar tendência de alta.
- **Oscillator Stochastic (K):** Assim como RSI, também indica se o ativo está acima ou abaixo do preço ideal. Varia também entre 0 e 100. Geralmente, valores maiores que 80 indicam supervalorização e abaixo de 20 indica subvalorização. O cálculo foi feito com as últimas 14 cotações.

No formato anterior, quando o agente de RL precisa-se tomar uma decisão de compra ou venda do ativo em um dia qualquer, era passado para ele apenas os valores brutos das últimas cotações (tamanho da janela) desse ativo para suportar a decisão. Agora, há um passo intermediário, onde antes dos valores brutos chegarem no agente, as métricas apresentadas são calculadas com as cotações dentro dessa janela. Nenhum desses indicadores são utilizados isoladamente por investidores da bolsa para tomar decisões, mas sim o conjunto delas os ajudam a tomar ações mais acertadas. Portanto, o agente precisa aprender uma metaheurística que relacione esses indicadores para suportar as suas decisões a fim de gerar lucro com operações feitas.

Função de recompensa

Em relação a função de recompensa, duas abordagens foram testadas, a primeira utiliza o tamanho do lucro obtido no investimento mensurado através da métrica de retorno. O cálculo dessa forma de recompensa utiliza o patrimônio total que é o valor total em reais que o agente tem para investir, somando com o valor que ele já possui investido no ativo financeiro. Assim, um agente que tem R$500 e 10 ações de um ativo que custa com R$100 possui um patrimônio total de 500 + 10 x 100 = 1500 reais.

O cálculo do retorno da primeira abordagem de recompensa é feito com o valor do patrimônio total atual (P_i) no dia i e do patrimônio total anterior (P_{i-1}) no dia i-1, calculado pela fórmula: $\frac{P_i - P_{i-1}}{P_{i-1}}$. Dessa maneira, um agente que tenha no primeiro dia R$1000 de patrimônio total, no segundo dia passe a ter R$950 e no terceiro dia fique com R$1015, receberá uma recompensa de -0.05 no segundo dia e de 0.015 no terceiro dia.

A segunda abordagem, ao invés do retorno, tenta analisar a coerência da decisão tomada pelo agente com a tendência do mercado. As cotações de um ativo financeiro sofrem períodos de alta e de baixa, a detecção de qual desses períodos o mercado está passando foi implementada por uma análise simples ao verificar a ordenação das últimas 3 cotações do ativo. Por exemplo, se as cotações do ativo em três dias consecutivos foram R$14, R$15 e R$18 (ordem crescente), então é assumido que o mercado está em alta, em contrapartida se os valores forem R$15, R$ 14.5 e R$13 (ordem decrescente), é assumido que o mercado está em baixa.

Nesse sentido, é esperado que quando o valor da cotação comece a subir (em alta) o agente compre o ativo para vender por um preço maior no futuro, já quando a cotação começar a diminuir o agente deve vender os ativos para não ter prejuízo ou garantir o lucro obtido. Portanto, se o mercado entra em alta e a decisão do agente é de comprar ou se o mercado está em baixa e o agente decide vender, a recompensa será de 1. Por outro lado, se o mercado entra em alta e o agente vende ou se o mercado entra em baixa e o agente compra, a recompensa será -1. Por fim, se o mercado não estiver nem em alta nem em baixa, a recompensa é 0 independente da ação.

Resultados:

Para realizar experimentos com o algoritmo DQN, foi escolhida a ação da empresa APPLE (AAPL) no período de 2012 a 2023, como pode ser visto na figura 1. O período 01/2012 à 12/2017 foi usado no treinamento e 2018 em diante foi usado para teste.

Figura 1: Cotações das ações da APPLE

O conceito de episódio nesse ambiente se trata de 300 dias seguidos começando a partir de uma data escolhida aleatoriamente no conjunto de treino em que o agente terá que tomar uma sequência de decisões de compra ou venda. Dessa forma, todos os testes foram realizados com 2000 episódios totalizando 600000 interações do agente com o ambiente da bolsa de valores.

Primeiro foi testado o algoritmo do DQN com a recompensa baseada no retorno diário. Na figura 2, é possível analisar em quais dias o agente optou por deixar o capital 100% investido na bolsa e em quais ele optou por vender todos os investimentos. Ao analisar o comportamento, é possível perceber que as decisões do agente não parecem seguir uma lógica esperada, onde para gerar lucro espera-se comprar o ativo num valor mais baixo para vender em valor mais alto.

Figura 2: Comportamento do agente DQN treinado com função de recompensa baseado no retorno obtido

O segundo experimento foi feito também com DQN, porém a função de recompensa foi trocada por aquela cujo agente é punido ou recompensado de acordo com a decisão tomada e a tendência das cotações. Na figura 3, é mostrado as decisões tomadas por esse agente frente a cotação do ativo. Perceba que dessa vez, na maioria das vezes, quando a cotação do ativo começa a cair, o agente resolve tirar todo o investimento no ativo, enquanto que quando ela começa a subir ele volta a investir.

Figura 3: Comportamento do agente DQN treinado com função de recompensa baseado na tendência das cotações

Para analisar se o algoritmo do DQN iria chegar consistentemente a valores semelhantes de capital ao final do período de teste, o experimento foi repetido com as mesmas configurações 5 vezes. Na figura 4, é possível analisar como o capital financeiro dos 5 agentes evoluem ao decorrer do período de teste, percebe-se que há 3 deles em que

o comportamento foi bastante semelhante e chegam a valores parecidos no final. Contudo, o agente do experimento 1 se destacou consideradamente dos demais ao obter quase o dobro de lucro final, já o experimento 5 o agente tem uma receita reduzida bruscamente se comparado aos demais. Portanto, vemos que o algoritmo do DQN consegue aprender nesse ambiente, porém seu desempenho em alguns casos apresenta certa variação.

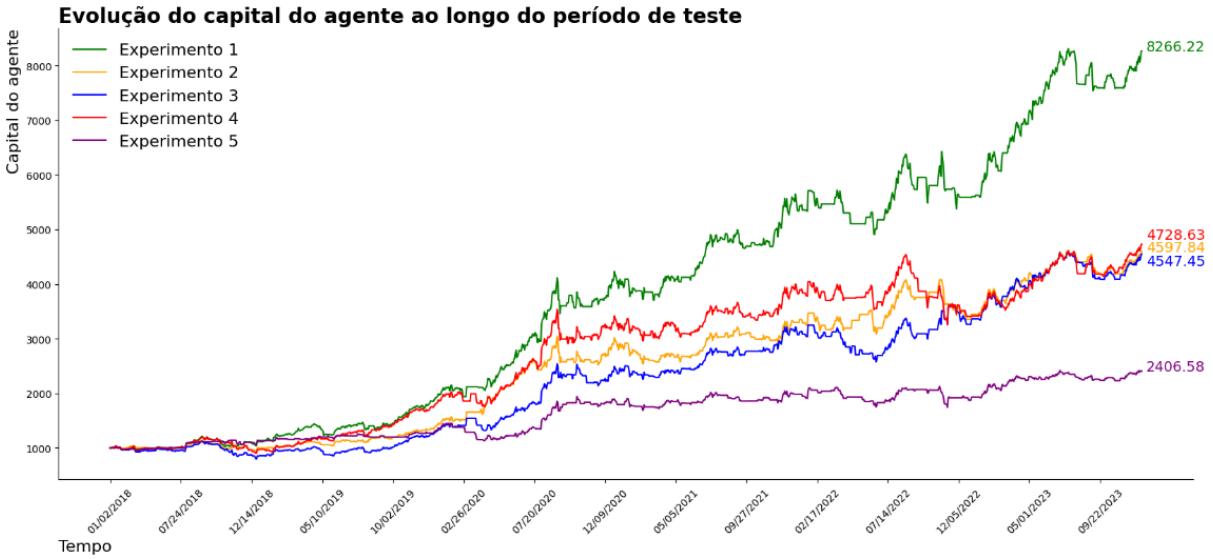


Figura 4: histórico do capital financeiro obtido pelo agente com os investimentos durante o período de treino

Em relação a comparação do desempenho das duas funções de recompensas testadas, no geral, percebe-se que a função que analisa a coerência da tendência de alta ou baixa dos valores e a ação tomada pelo agente obteve melhores resultados que função que analisava somente o retorno do investimento. Tal comparação é evidenciada na figura 5, onde há a comparação da evolução do capital dos dois agentes, além da inclusão também de um baseline que é obtido ao investir todo o capital no primeiro dia (01/01/2018) e somente vender no último dia (19/12/2023). Dentre os 5 agentes treinados com a recompensa com tendência, foi selecionado aquele com melhor desempenho.

Figura 5: Comparação entre baseline e as duas recompensas com DQN

Conclusão

Por meio dos experimentos feitos, nota-se que a troca do framework RLlib e do PPO pela outra implementação do DQN se mostrou uma boa escolha, pois é visível que dessa vez o agente de fato conseguiu aprender um comportamento melhor se comparado com os testes dos Stages anteriores. Além disso, o agente parece ter aprendido a utilizar os indicadores financeiros calculados de forma a tomar decisões mais coerentes e conseguir, em alguns casos, gerar um lucro considerável com as operações de compra e venda. Por fim, a mudança na função de recompensa mostrou ter grande impacto no desempenho final do agente e pode ser um caminho promissor para melhorar ainda mais os resultados no futuro.

Termo de Aceite de Entrega

Objetivo deste documento

Este documento faz parte do Processo da disciplina Residência em IA e tem como objetivo formalizar o aceite da entrega considerando o planejado e o realizado para o período.

Data da Reunião ("gate") de aprovação: 11 de jan. de 2024

Participantes da Entrega [matriculados em Residência em IA]:

Luís Augusto

Entrega: [descrever a ENTREGA: requisitos e produtos gerados: links para textos, códigos, vídeos etc.]

Para o último Stage, o foco foi melhorar os resultados a aprendizado por reforço no ambiente de simulação da bolsa de valores. A ação utilizada foi a da empresa Apple, onde de 2005 até 2020 para treino e 2020 e 2024 para teste. Os novos experimentos realizados incluem principalmente alterações na função de recompensa utilizada e, além do DQN, outros algoritmos foram testados.

Segue o link abaixo com maiores detalhes sobre os experimentos realizados;

https://docs.google.com/document/d/1voeiOLx7WI-SAj0vCIiULPCgqEOGq4WMeaWgwOyfIQk/edit?usp=sharing

Planejamento: [descrever o que pretende fazer para realizar a próxima ENTREGA]

Observação: [caso precise fazer alguma observação, de qualquer "natureza"]

ACEITE DA ENTREGA:

CEDRIC LUIZ DE CARVALHO: Go!

LUANA GUEDES BARROS MARTINS: Go!

Teste com vários algoritmos de Reforço e funções de recompensa

No stage 10, mais experimentos foram realizados com aprendizado por reforço no ambiente de Trading do mercado financeiro. O foco foi alterar a função de recompensa para analisar o seu impacto no desempenho dos algoritmos já usados como também em novos. Os experimentos foram realizados com a ação da empresa APPLE cujo ticket na bolsa é AAPL, o período dos dados de treinamento abrange os anos de 2005 até 2019 e os anos de 2020 até 2023 foram usados como teste.

Em relação ao ambiente, as demais configurações continuam iguais às do Stage 9, onde o espaço de observações é construído com os indicadores utilizados por investidores da bolsa como Reversão à Média, Índice de força relativa, Oscilador estocástico entre outros já apresentados nos Stages anteriores Semana 8. Já o espaço de ações também permanece inalterado com as opções apenas de investir todo o capital (100%) ou não investir nada (0%). Além dessas configurações, o tamanho dos episódios são fixos em 300 dias de operações e a janela para os cálculos dos indicadores é de 15 dias.

Até nos stages anteriores, um agente treinado era avaliado em todo o período de teste de um única vez. Nesse Stage 10, foi decidido adotar a abordagem episódica para o período de teste também. Assim, um dia é escolhido aleatoriamente entre o período de 2020 a 2023 e então a simulação começa a partir daí pelos próximos 300 dias. Dessa forma, a maneira como o agente é avaliado no teste é semelhante a como foi treinado, além disso é possível testar o agente em situações diferentes a cada episódio.

Funções de Recompensa

No stage 9, foram mostrados testes com as funções de recompensa de retorno e de tendência, a primeira utiliza o lucro percentual do agente de um dia para o outro, já o segundo foca em usar o histórico de cotações para estimar um período de baixa ou alta e com isso determinar se a ação tomada pelo modelo é ou não coerente. Informações mais detalhadas podem ser obtidas no documento entregue na Semana 9.

Uma dificuldade durante o treino foi analisar se os agentes treinados de fato aprenderam estratégias relevantes para operar na bolsa de valores, por isso foi pensado em um baseline para comparar o desempenho do agente no decorrer do treinamento. O baseline é construído simulando um investidor que compra o ativo no primeiro dia e só vende as ações na carteira de investimentos no último dia. É esperado que os agentes treinados consigam um desempenho melhor que baseline, visto que haverá oportunidades de vender o ativo a um preço mais alto e comprá-lo a um preço mais baixo para gerar lucro.

Nas tentativas de treinamento, foi observado que os agentes que treinavam com as funções de tendência e retorno conseguiam melhorar as recompensas, porém isso não se refletia na comparação com o baseline. Nesse sentido, percebe-se que maximizar essas recompensas não refletem necessariamente em uma boa estratégia de investimentos.

Assim, foi necessário pensar em outras funções de recompensa para que o agente aprendesse estratégias melhores do que simplesmente comprar e segurar as ações o máximo possível, como é no caso do baseline.

Outras duas funções de recompensa foram criadas para realizar novos experimentos. A primeira delas foi utilizar diretamente o baseline para dar feedback ao agente se a ação que ele executou é boa ou ruim, isso é feito por meio da razão entre o patrimônio obtido pelo agente e o patrimônio obtido pelo baseline até o momento. Dessa forma, se a razão for maior que 1, a estratégia do agente é melhor que a do baseline e deve ser recompensada, caso contrário, a estratégia é pior que o baseline e deve ser punida.

No geral, com essa função de recompensa, é possível notar que o agente ao longo do treinamento começa pior que o baseline e vai melhorando até conseguir resultados consistentemente melhores. Apesar disso, é levantada a hipótese de que com essa abordagem ocorra algum tipo de overfitting no treinamento visto que, nos episódios de teste (2020 até 2023), o desempenho do agente caiu consideravelmente e ficou abaixo do baseline.

A outra função de recompensa criada, foi uma semelhante a de tendência anteriormente citada, porém foi usado a cotação do próximo dia e não as cotações anteriores. Dessa maneira, a decisão do agente de comprar ou vender era analisada juntamente com a cotação atual e a do próximo dia, assim, se o agente decidiu vender e a cotação irá cair ou se o agente decidiu comprar e a ação irá subir ele é recompensado, e em qualquer outro cenário ele é punido.

Para esse último mecanismo de recompensa o baseline foi superado consistentemente depois de uma certa quantidade de episódios no treinamento. É interessante notar que, dessa vez, o agente atingiu esse feito mesmo sem uma comparação direta com o baseline ter sido usado na função de recompensa. No contexto dos dados de teste, o desempenho cai um pouco, como esperado, mas ainda sim a estratégia desenvolvida pelo agente supera o baseline na maioria dos episódios. Por exemplo, no DQN treinado com essa função, de 1000 episódios feitos nos dados de teste 76 foram melhores que o baseline e 24 piores.

Algoritmos

	Q-learning	DQN	DDPG	ARS	A2C
Recompensa Média	20.82	16.78	7.73	11.64	10.98
Desvio padrão recompensa	14.10	11.40	5.88	8.10	8.51
% de episódios melhor que o baseline	66%	71%	13%	0%	0%

média diferença baseline	71.29	6.21	-25	-1	-1
Desvio Padrão diferença baseline	270.01	195	56.91	0	0

Tabela 1: Comparação entre o desempenho dos algoritmos em 1000 episódios dos dados de teste.

Nas primeiras semanas da residência que foram feitos testes com aprendizado por reforço, as observações do ambiente que eram passadas para o agente eram puramente os dados brutos das cotações passadas. Essa abordagem inicial jogava para o agente todo o trabalho de aprender a como relacionar os dados brutos das cotações para tomar a melhor decisão. A fim de facilitar o treinamento do agente, indicadores do mercado financeiro calculados a partir dos dados brutos passam a ser utilizados como observação, assim o agente já recebe features que investidores reais usam para tomar as decisões de investimento. Tal modificação fez o desempenho do agente melhorar e aprender estratégias de forma mais fácil para operar na bolsa.

Inicialmente, esses indicadores são dados numéricos, porém em cada um deles existem alguns limiares que indicam situações em que se deve prestar mais atenção para tomar a decisão. Assim, seguindo essa linha de raciocínio, um experimento foi feito com a discretização dessas features de acordos com os limiares comumente utilizados por investidores. Os detalhes da discretização podem ser visualizados abaixo:

- **Mean reversion:** foram criadas duas features booleanas a partir dessa, uma indica se o valor superou o limiar de 1.5 e outra se o valor desceu abaixo de -1.5.
- **Relative Strength Index:** Varia de 0 a 100 e duas features foram criadas, uma delas indica se o valor passou de 70 e a outra se o valor está abaixo de 30.
- **Moving Average Convergence Divergence (MACD) e Signal:** A partir desses dois indicadores uma nova features booleana foi criada para verificar se o MACD ultrapassou ou o Signal.
- **Stochastic Oscillator:** Varia entre 0 e 100, duas novas features são criadas uma indica se o valor passou do limiar de 80 e outra se está abaixo de 20.

Com essa discretização das features numéricas em boleanas de 0 ou 1, o espaço de observação se reduz bastante, com isso foi possível pensar na aplicação de um algoritmo mais clássico que não utiliza redes neurais, o Q-learning. A ideia desse algoritmo é criar uma tabela que mapeia a recompensa esperada por cada ação em cada um dos estados possíveis. A otimização desse algoritmo é feita pelas equações de Bellman de tal forma que com decorrer dos episódios de treinamento o tabela vai estabilizando os seus valores.[1]

O Q-learning foi testado no ambiente de trading e conseguiu otimizar a função de recompensa de tendência que usa o valor da cotação do dia seguinte. Como pode ser visto na tabela 1, esse algoritmo foi melhor que baseline em 66% dos casos, mas os resultados não são muitos consistentes visto que o desvio padrão é consideravelmente alto. Além

disso, na figura 1 é possível notar que tanto a recompensa quanto a comparação com baseline estabilizaram logo no início do treinamento.

O melhor agente obtido em todos os testes foi o Deep Q-Network(DQN), ele foi testado no ambiente com features numéricas e com todas as recompensas individualmente e até combinações entre elas. A abordagem que atingiu melhor resultado foi a recompensa com tendência com a cotação do próximo dia, onde o agente superou o baseline cerca de 70% das vezes nos episódios do período de teste, como visto na tabela 1. Contudo, comparado ao Q-learning com $71.29 em média acima do baseline, o DQN atingiu apenas $6.21 acima do baseline. Isso pode indicar que apesar do DQN superar o baseline mais vezes, quando o Q-learning supera o baseline é em uma escala muito maior que o DQN.

Apesar do resultado no teste, o DQN ficou abaixo dos demais modelos no treino, contudo diferente do Q-learning e do DDPG, na figura 1, nota-se que o gráfico de recompensa e de comparação com baseline esteve aumentando durante todo o treinamento e possivelmente subiria mais com um período de tempo de treino maior.

O Deep Deterministic Policy Gradient (DDPG) é diferente dos demais, visto que este algoritmo só funciona em simulações onde o espaço de ações é contínuo. Dessa forma, foi preciso adaptar o ambiente para permitir que, ao invés apenas escolher entre investir 0% ou 100% do patrimônio, o agente agora poderá escolher investir uma porcentagem arbitrária. Feitas essas alterações mencionadas o agente foi treinado, mas mesmo depois de 6000 mil episódios de treinamento, o agente não conseguiu superar o baseline de forma consistente. Nos dados de teste, na tabela 1, é observado o seu péssimo desempenho em relação ao baseline, onde o DDPG foi melhor apenas 13% dos episódios. Na figura 1, percebe-se que o desempenho do DDPG caiu em relação à recompensa e manteve-se parelho ao baseline ao longo do treino .

As implementações utilizadas dos algoritmos DQN e DDPG nos experimentos foram do repositório da Udacity [2].

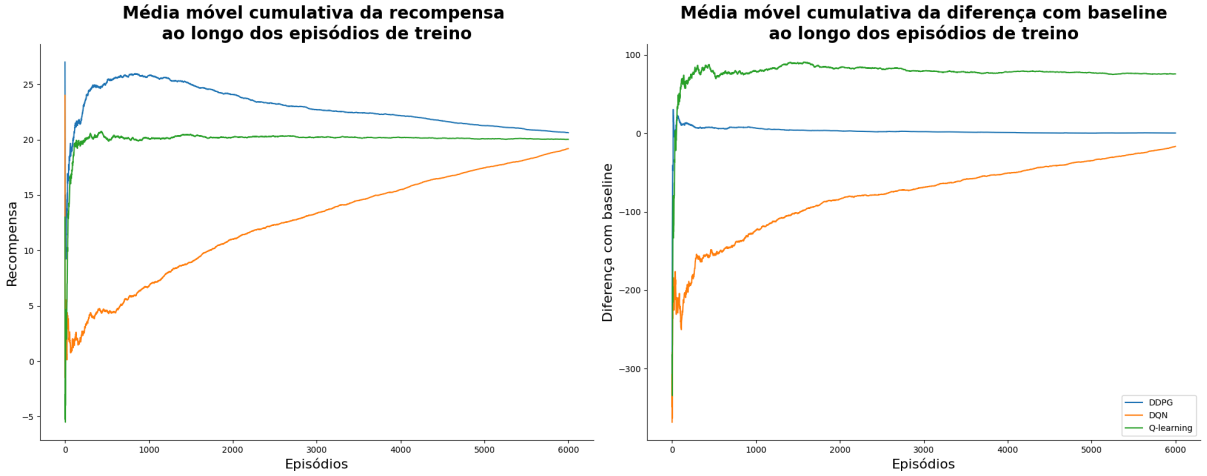


Figura 1: Evolução do desempenho do DDPG durante o treino

Com o Stable Baselines, uma biblioteca que traz implementações confiáveis de alguns algoritmos de IA, foi feita a tentativa de treinar outros dois agentes com os algoritmos ARS e A2C [3, 4]. As recompensas obtidas até são otimizadas e aumentam no decorrer do treino, contudo assim como acontecia no RLlib, no teste os agente optam por justamente a estratégia do baseline, como pode ser observado na tabela 1. Ou seja, os agentes compram as ações logo no primeiro dia e nunca as vendem.

O código execução dos experimentos está no github:
https://github.com/LuisAugusto0205/trading_environment

Conclusão

As funções de recompensa tiveram um impacto grande no treinamento dos agentes, onde a melhor delas se mostrou ser a de tendência que usa a cotação do dia atual e do próximo para analisar a coerência das ações tomadas pelo agente. Os algoritmos do DQN e Q-learning conseguiram superar o baseline nos testes, porém o mesmo não acontece com o DDPG, já o ARS e A2C aprenderam a estratégia de investimento do próprio baseline. Apesar do bom desempenho do DQN e do Q-learning, ambos apresentam uma grande variação no lucro obtido a cada episódio comparado ao baseline.

Referência:

[1]https://www.deeplearningbook.com.br/algoritmo-de-agente-baseado-em-ia-com-reinforcement-learning-q-learning/
[2]https://github.com/udacity/deep-reinforcement-learning
[3]https://stable-baselines3.readthedocs.io/en/master/modules/a2c.html
[4]https://sb3-contrib.readthedocs.io/en/master/modules/ars.html