IA aplicada ao Mercado Financeiro

Gustavo dos Santos Ferreira
UNIFESP
RA: 140731
São José dos Campos, São Paulo
gsferreira16@unifesp.br

***Abstract*—Em tempos recentes, se vê um crescente interesse da população mundial no mercado de ações financeiras, que envolve grandes fluxos monetários que dependem de fatores praticamente imprevísiveis. Por esse motivo, ser capaz de prever tendências de súbida ou queda nos preços de ações abre a possibilidade de se obter um grande retorno monetário. Outra tendência moderna do interesse das pessoas está no campo de inteligência artificial, mais especificamente de redes neurais, poderosos modelos computacionais com vasto número de aplicações. Este trabalho busca investigar a eficácia de redes neurais artificiais na previsão de ações do mercado financeiro.**

***Index Terms*—redes neurais, mercado financeiro, inteligência artificial.**

I. Introdução e Motivação

Bilhões de dólares são diariamente trocados no mercado financeiro global. Nas últimas décadas também tem sido observado um crescimento exponencial do interesse das mais variadas pessoas em investir nesse mercado. Tal interesse é esperado, uma vez que muitas são as histórias contadas daqueles que obtiveram grande sucesso monetário por meio de seus investimentos. De um ponto de vista sistemático, o mercado financeiro é complexo, evolucionário e não linear [1].

O mercado financeiro é muitas vezes visto como um tipo de aposta, uma vez que é impossível conhecer todos os fatores que levam a variação das muitas ações existentes. Temos então um problema de natureza imprevisível mas que pode ser extremamente lucrativo quando dominado. No passado, pesquisas e algoritmos foram desenvolvidos a fim de observar o comportamento superficial do mercado financeiro, ou seja, as tendências de subida e queda. A esses problemas, observamos técnicas de auto regressão, onde a solução proporcionada é tentar estimar preços futuros com base nas variações estudadas no passado.

É nessa lógica que múltiplos algoritmos estatísticos, como média móvel integrada auto regressiva (ARIMA), heterocedasticidade condicional auto regressiva (ARCH), filtro de Kalman e suavização exponencial foram aplicados ao contexto do mercado financeiro [2]. Com os recentes avanços na tecnologia de unidades gráficas de processamento (GPUs), também se viu grandes avanços nas pesquisas de inteligência artificial, uma vez que o maior poder computacional potencializou descobertas maiores ainda. Como a área que mais vê novas e variadas pesquisas no contexto atual, é de se esperar que suas soluções venham a ser aplicadas aos mais variados campos. Com isso em mente, pesquisadores buscam aplicar técnicas de aprendizado de máquina a fim de estudar o comportamento de ações do mercado financeiro e melhorar as previsões realizadas.

Uma estratégia empregada por meio de técnicas de inteligência artificial é a de auto regressão, sendo capaz de atingir resultados satisfatórios quando comparada às técnicas estatísticas. Outros métodos envolvem o uso de máquinas de vetor suporte (SVMs) ou aprendizado de reforço, para, por exemplo, fazer previsões com base nas perspectivas de autoridades no ramo financeiro ou pessoas importantes [3]. A isso é dado o nome de análise de sentimento. Máquinas de vetor suporte também são muito utilizadas para classificar tendências de súbida ou de queda.

A ambas essas estratégias podemos ver fraquezas que as impedem de alcançar previsões perfeitas. A hipótese dos mercados eficientes rejeita a possibilidade de que o método de auto regressão seja eficaz, por acreditar que os mercados se comportem como uma Caminhada Aleatória, significando ser impossível sua previsão [4]. Porém, analistas técnicos acreditam que os atributos mais importantes já estão refletidos nos preços mais recentes, possibilitando previsão caso tendências de súbida ou queda sejam observadas [4]. Enquanto isso, a análise de sentimento pode ser vítima de perspectivas enviesadas ou opiniões que se baseiam puramente no sucesso passado.

Mesmo com existentes limitações, a evolução de técnicas de inteligência artificial motiva mais e mais estudos a aplicar novos e mais sofisticados modelos no campo financeiro. Por esse motivo, este trabalho se propõe a estudar e desenvolver uma rede neural a ser treinada sobre os dados de ações financeiras e devolver a nós preços previstos que se aproximem do preço real com erro mínimo.

II. Conceitos Fundamentais

A. Mercado Financeiro e Ações

O mercado financeiro pode ser dividido em quatro segmentos: Mercado monetário, mercado de crédito, mercado cambial e mercado de capitais. Para o escopo deste trabalho, é de mais relevância o mercado de capitais. Uma de suas funções mais comuns está em sediar a troca de ativos de renda variável, também chamados de ações. Uma ação representa a menor parcela em que se subdivide o capital de uma sociedade anônima, demonstrando posse de parte do capital da empresa que a emitiu. A posse de um acionista é proporcional ao número de ações que detém [6].

Para obter o principal retorno de seu investimento, o investidor deve revender seu ativo. Para a empresa emissora, somente a abertura de capital e o lançamento das ações no chamado mercado primário gera riquezas. Após serem lançadas, as ações são comercializadas no mercado secundário e somente representam a troca de um agente para outro [6], sem efeito ou retorno à empresa emissora.

No Brasil, a comercialização de ativos deve ser realizada através de intermediação de uma corretora autorizada. A bolsa de valores nacional mais importante é a BM&FBOVESPA, agindo como central das operações realizadas em seus ambientes. O sistema de *home broker* é um dos mais populares utilizados para comércio de ações. Se caracteriza pela conveniência de ser realizado pela internet, permitindo enviar ordens de compra e venda e acompanhamento em tempo real da carteira de ações [6].

B. Aprendizado de Máquina

O conceito de capacidade de aprendizado é um das principais características da inteligência humana. A partir de experiências passadas, podemos moldar nossos comportamentos e expectativas quando nos deparamos com situações semelhantes às que já conhecemos. Da mesma maneira, o aprendizado de máquina se refere ao emprego de técnicas computacionais para simular a experiência adquirida por humanos em computadores. A experiência de uma máquina se dá na forma de dados, assim, o aprendizado de máquina é o conceito de se criar modelos com base nesses dados. À alimentação de um algoritmo com dados é dado o nome de treinamento e ele nos dará um modelo capaz de realizar predições quando se deparar com novas experiências. De [7], "se considerarmos ciência da computação como o campo dos algoritmos, então aprendizado de máquina é o campo de aprender algoritmos."

Quando não basta alimentar dados ao programa e desejamos também que esses dados alcancem um resultado esperado, podemos utilizar de um conjunto de dados de validação para verificar quão perto ou quão longe estão os resultados de nosso programa da verdade que desejamos que ele alcance. Quanto mais longe estiverem, devemos indicar ao programa que deve realizar um grande reajuste para alcançar a verdade. A isto é dado o nome de aprendizado supervisionado, uma vez que ao mesmo tempo que ensinamos ao programa o que existe no mundo, o guiamos para que aprenda a nos devolver observações que se alinhem com o que esperamos. O aprendizado supervisionado é útil quando desejamos que o programa realize uma tarefa de classificação ou regressão. Classificação se refere a relacionar os dados de entrada a um rótulo pré-definido, ou seja, a previsão do programa é discreta, enquanto regressão se refere a relacionar os dados de entrada a um valor contínuo como previsão. Neste trabalho, realizaremos uma tarefa de regressão.

C. Autoregressive Integrated Moving Average

Autoregressive Integrated Moving Average (ARIMA) é um modelo utilizado para a análise de séries temporais. Seu uso se dá principalmente na análise de séries temporais não estacionárias, onde a distribuição de probabilidade conjunta não muda durante o tempo [8]. Como funciona de maneira autorregressiva, o modelo aprende com base na própria série temporal que deseja prever. O erro da regressão é baseado em uma combinação linear dos erros passados e do erro presente, uma vez que o algoritmo funciona com base em média móvel. Finalmente, a integração do modelo é aplicada de modo a eliminar a não estacionaridade. Essas funções buscam fazer com que o modelo aprenda os dados o melhor possível [9]. Para conjuntos de dados onde se esperam grandes variações de um instante para outro, podemos adaptar o modelo para realizar uma previsão rolante, onde o conjunto de teste da ARIMA tem somente um elemento em cada instante de tempo. A partir de um conjunto de treino inicial, realizamos a previsão do elemento do conjunto de teste e adicionamos ele a lista de previsões. Então, movemos o elemento do teste para o treino e adicionamos um novo elemento de teste para treinarmos novamente o modelo. Esses passos se repetem até não termos mais elementos de teste.

D. Redes Neurais Artificiais

No campo de inteligência artificial, as redes neurais artificiais, em inglês *artificial neural network* (ANN), são modelos matemáticos onde as unidades, chamadas de neurônios, são funções com entradas e saídas próprias e estão interligadas entre si. Cada uma dessas ligações apresenta um peso, ou seja, um valor numérico, que multiplicará a entrada que recebem e a enviará para o próximo neurônio. As redes neurais de duas camadas, ou seja, uma camada de entrada e uma de saída, são chamadas de *Perceptrons* e são capazes de resolver problemas linearmente separáveis, como as funções lógicas *AND*, *OR* e *NOT* [7]. Para solucionar problemas que não são linearmente separáveis, é necessário aumentar o número de camadas. Às camadas no meio é dado o nome de camadas escondidas, em inglês, *hidden layers*. Às redes neurais com multiplas camadas é dado o nome de *multiple layer perceptron* (MLP).

Para treinar uma rede neural, precisamos observar os resultados que ela obtem a partir das entradas. Calculando a distância dos resultados obtidos aos resultados esperados, obtemos o erro cometido pela rede neural. Existem várias métricas para o cálculo dessa distância, como *Mean Squared Error* (MSE), em português, erro quadrado médio. Então, utilizamos de uma função de retropropagação para ajustar os pesos desde a camada de saída até a camada de entrada com base no erro obtido. Isso permite a rede neural aprender como alcançar as saídas esperadas a partir das entradas que recebe, mas pode torna-la muito engessada ao seu treinamento. O principal desafio do campo é desenvolver modelos que sejam capazes de alcançar bons resultados gerais para aquilo que são treinados e não só reconhecer os dados em que são treinados. Por esse motivo, o chamado *overfitting*, quando um modelo acerta muito dos dados em que é treinado mas não é capaz de fazer o mesmo para dados novos, é algo a ser evitado. Outro desafio é que no presente momento não são conhecidas técnicas que permitam ditar a topologia ideal de uma ANN para cada tipo de problema. Assim, muito do processo de

desenvolvimento de redes neurais é baseado em tentativa e erro a fim de alcançar os melhores resultados.

E. Long Short Term Memory

Long Short Term Memory se refere a um tipo de rede neural recorrente, capaz de utilizar as saídas de seus próprios nós para alimentar suas entradas. Este tipo em específico tem como característica uma célula, que processa a informação recebida em sua entrada, um portão de saída e um portão de esquecimento. O portão de esquecimento é responsável por decidir quais informações manter e quais informações descartar de iterações anteriores durante o aprendizado. Dessa forma, as informações da entrada atual, saída anterior e esquecimento anterior são utilizadas no processamento atual. É da característica de preserver informações a longo prazo, mas também de poder esquecê-las que está rede neural recebe seu nome [10]

III. Trabalhos Relacionados

A. Previsão por meio de Redes Neurais Artificiais

Os autores de [11] verificaram a eficácia de uma ANN aplicada sobre a série temporal da ação NASDAQ. O pré-processamento dos dados é realizado por meio da normalização da série temporal. Para medir o erro, os autores utilizaram da métrica de *Mean Squared Error* (MSE), em português, erro quadrado médio. A rede neural em si utiliza retropropragação do erro para atualizar seus pesos. Os resultados obtidos pelos autores se mostraram bastante próximos dos valores reais em cada período.

No artigo [12] é apresentada uma comparação dos modelos de ANN e Floresta Aleatória quanto a sua eficácia em prever o preço de fechamento de ações. O algoritmo de Floresta Aleatória envolve utilizar os resultados de multiplas Árvores de Decisão, outro tipo de algoritmo de aprendizado, a fim de obter um resultado ou por média ou por voto maioritário. O funcionamento de uma Árvore de Decisão se dá pela criação de regras durante o seu treinamento, com cada regra direcionando a outra até que uma folha seja alcançada, o que nos dará o resultado.

O treinamento descrito em [12] se deu a partir das séries temporais das ações de cinco companhias: Nike, Goldman Sachs, Johnson and Johnson, Pfizer e JP Morgan Chase and Co. A partir dessas séries temporais, foram extraídas as seguintes informações de modo a servirem para alimentar o treinamento dos modelos comparados: Diferença do Preço Máximo e Preço Mínimo, Diferença do Preço de Fechamento e Preço de Abertura, Média Móvel por 7 dias, Média Móvel por 14 dias, Média Móvel por 21 dias, Desvio Padrão dos últimos 7 dias. As métricas utilizadas pelos autores para medir a performance dos modelos foram Raiz do Erro Quadrado Médio (RMSE), Percentual do Erro Médio Absoluto (MAPE) e Erro Enviesado Médio (MBE). Os resultados apresentados pelos autores revelaram que sua Rede Neural teve resultados levemente mais precisos que os da Floresta Aleatória.

B. Máquina de Vetor Suporte para Predição

O trabalho [2] utiliza do algoritmo de *Support Vector Machine* (SVM), em português, Máquina de Vetor Suporte, para prever se uma ação estará em tendência de súbida ou queda em um dado dia, ou seja, um problema de classificação. O treinamento de uma máquina de vetor suporte se dá por encontrar a melhor combinação dos exemplos que recebe como entrada que gere um separador de margem máxima entre duas classes, chamado hiprplano. SVMs podem ser permitidos classificar exemplos de maneira errônea, a fim de alcançar uma melhor precisão geral. Quanto mais distante um dado estiver do hiperplano, mais confiança temos de que esta classificado corretamente.

O alvo deste trabalho foram as mudanças semanais da ação NIKKEI 225. O motivo para escolha dessa ação está em sua característica de representar uma boa parte da indústria japonesa na época (2004). Para descobrir a tendência de súbida ou queda, os autores definiram como t o preço na semana que querem prever e decidiram por utilizar o preço de troca da moeda japonesa (JPY) por dólar na semana t-1 como entrada do treinamento da SVM. Como o mercado dos Estados Unidos tem forte influência sobre o mercado japonês, as condições do Indíce S&P500 na semana t-1 também foram considerados para alimentar o treinamento da SVM.

Para comparar a eficácia seu classificador, os autores utilizaram os algoritmos de Caminhada Aleatória, Análise do Discriminante Linear, Análise do Discriminante Quadrático e Rede Neural com Retropropagação de Elman. Além disso, os autores também optaram por treinar um modelo combinador. Nessa lógica, atribuiram um peso a cada um dos algoritmos que pretendiam comparar em seu trabalho, com cada peso sendo calibrado com base na performance de cada algoritmo, da mesma forma que um treinamento. Em seus resultados finais, os autores demonstraram que sua SVM teve o melhor resultado individual, mas que os algoritmos combinados ponderados alcançavam uma precisão levemente maior que a SVM. Isso se deve a capacidade de um algoritmo cobrir as fraquezas possuidas por outro.

C. Fusão de Técnicas de Aprendizado de Máquina para Previsão do Preço de Ações

O estudo de [4] apresenta a proposta de combinar modelos diferentes em um modelo de dois estágios de modo a alcançar maior precisão em suas previsões. Para realizar comparação cruzada, os autores produziram combinações diferentes. Em todos os modelos criados pelos autores, *Support Vector Regression* (SVR), em português, Regressão por Vetor Suporte, é aplicada como a primeira camada das combinações. As segundas camadas envolveram Rede Neural, Regressão por Vetor Suporte e Floresta Aleatória, sendo chamadas pelos autores como SVR-ANN, SVR-SVR e SVR-RF, respectivamente.

O papel da primeira camada é preparar as entradas que serão recebidas para treinamento da segunda camada. O treinamento envolveu a predição do (t+n)-ésimo dia futuro, onde t é o dia atual e n é o número de dias a frente que desejamos prever. As séries temporais utilizadas foram extraídas a partir dos

preços das ações CNX Nifty e S&P BSE durante 10 anos. As entradas recebidas pela primeira camada envolveram diversas métricas que podem ser extraídas de uma série temporal, como Média Móvel Simples, Média Móvel Exponencial, Momento, entre outras. Mais especificamente, as métricas utilizadas pelos autores são apresentadas em detalhes em [5]. Para medir a performance de cada modelo, as métricas de erro utilizadas pelos autores foram Percentual do Erro Absoluto Médio (MAPE), Erro Absoluto Médio (MAE), Raiz Relativa do Erro Quadrado Médio (rRMSE) e Erro Quadrado Médio (MSE).

Os testes realizados pelos autores envolveram tanto os modelos de dois estágios como seus equivalentes de um estágio, ANN, SVR e RF. Os autores realizaram previsões para valores de n de 1 a 10 dias, de 15 dias e de 30 dias. Os resultados mostraram que na maioria dos casos, os modelos de dois estágios se mostraram mais precisos em suas predições. Como esperado, para valores de n maiores, ou seja, quanto mais longe o dia futuro estiver do dia atual t, maior o erro obtido por todos os modelos, porém, os modelos de dois estágios demonstraram precisão consideravelmente maior nesses casos.

D. Modelo Híbrido de Transformada Wavelet e Rede Neural

[13] apresenta a proposta de decompor as séries temporais comumente utilizadas como entrada de redes neurais em problemas de previsão do preço de ações por meio da transformada *Wavelet*. Essa transformada decompõe sua entrada em Coeficientes de Approximação (de baixa frequência) e Coeficientes Detalhados (de alta frequência) [13]. O intuito de utilizar essas decomposição está em extrair informações escondidas e detalhes temporais da série temporal original [13]. As informações obtidas da decomposição são então utilizadas para alimentar o treinamento de uma rede neural com retropropagação de erro. Ao fim de seu trabalho, os autores comparam seu método com uma rede neural que foi treinada sobre a série temporal não decomposta. Os dados utilizados pelo trabalho foram obtidos a partir do site Yahoo! Finance. As métricas para cálculo de erro foram Coeficiente de Variação, Desvio Médio Absoluto, Raiz do Erro Quadrado Médio e Erro Médio Absoluto. A partir dos resultados obtidos, os autores concluiram que seu método se mostrou mais preciso que o convencional. Por fim, eles sugerem combinar seu método de pré-processamento com outros algoritmos de treinamento em trabalhos futuros.

E. Previsão de ações financeiras através de Máquina de Vetor Suporte

O trabalho apresentado em [1] estuda a interação entre ações estadunidenses e ações internacionais com a finalidade de fazer previsões no mercado de ações. Inicialmente, os pesquisadores realizam um cálculo de correlação para identificar quais ações tendem a subir quando a referência deles, a ação *NASDAQ*, sobe. Identificada a correlação das ações, eles utilizam de uma Máquina de Vetor Suporte (SVM) para realizar um estudo inicial, classificando trechos das séries temporais somente a quanto sua tendência de súbida ou de queda. Uma máquina de vetor suporte é um modelo que utiliza de uma pequena parte de seus dados para identificar um hiperplano capaz de alcançar boa separação do conjunto de dados de treino entre duas classes diferentes, sendo frequentemente uma boa escolha para realizar um aprendizado inicial de um conjunto de dados.

Com a hipótese de que dados numéricos poderiam alcançar uma precisão melhor, os pesquisadores então tentaram comparar seu modelo de SVM com métodos de regressão linear e modelo linear generalizado. Ainda sim, o modelo de SVM se mostrou o mais preciso. Com o objetivo de não só maximizar os lucros do retorno, mas também de minimizar os riscos, os pesquisadores aprimoraram seu algoritmo de SVM introduzindo uma terceira classe que representava dados "neutros", ou seja, aqueles muito próximos da linha divisória e que por esse motivo não eram tão confiáveis. Com o modelo pronto, os pesquisadores montaram uma simulação onde compararam seu modelo a dois *benchmarks*, revelando que seus resultados eram melhores, atingindo 8% de taxa de retorno a cada 50 dias a um baixo risco.

F. Comparações entre diferentes tipos de aprendizado de máquina

O trabalho apresentado em [14] realiza um estudo geral da eficácia de vários tipos de modelos de aprendizado a fim de identificar o melhor que pode ser aplicado a ações financeiras com a finalidade de realizar as melhores previsões. O conjunto de dados utilizado para treinamento dos modelos foi extraído do mercado financeiro iraniano, envolvendo quatro campos: Finanças diversas, petróleo, minerais não-metálicos e metais básicos. Tais dados foram pré-processados de modo a obter várias métricas úteis para o treinamento, como a média ponderada dos n últimos valores e a velocidade de súbida ou queda. Para realizar a medida da eficácia de cada modelo, os pesquisadores utilizaram Erro Percentual Médio Absoluto (MAPE), Erro Absoluto Médio (MAE), Raiz do Erro Médio Quadrado (RMSE) e Erro Médio Quadrado (MSE).

Quanto aos modelos comparados, os pesquisadores escolheram 6 modelos de árvore: Árvore de Decisão, *Bagging*, Floresta Aleatória, *Adaboost*, *Gradient Boosting* e *XGBoost* e 3 modelos de redes neurais: Redes Neurais Artificiais (ANN), Redes Neurais Recorrentes (RNN) e *Long Short Term Memory* (LSTM). Estes dois ultimos tipos de modelos são aqueles que chamamos de aprendizado profundo.

Em geral, todos os modelos tinham 10 entradas que envolviam as métricas previamente obtidas no pré-processamento. Somente os modelos de aprendizado profundo demandaram uma atenção especial de adaptação dos dados. As conclusões dos pesquisadores foram de que os modelos de aprendizado profundo foram os melhores no quesito de se obter boas previsões do mercado financeiro. Em compensação, tais modelos também apresentaram o maior tempo de execução necessário para seu treinamento.

O trabalho apresentado em [15] compara a eficácia de um algoritmo de aprendizado profundo, mais especificamente LSTM, com outras técnicas de aprendizado de máquina. As comparações são feitas com os algoritmos de Floresta

Aleatória, Rede Neural Artificial e Regressão por Vetor Suporte.

Os algoritmos foram aplicados sobre a série temporal de preços de fechamento da *iShares MSCI United Kingdom*. Os resultados que eles obtiveram em seu estudo foi de que o algoritmo LSTM alcançou os melhores resultados, mas que entre os algoritmos que não são de aprendizado profundo, a Regressão por Vetor Suporte se mostrou a mais precisa. Assim, a recomendação dos autores em sua conclusão é que aqueles interessados em fazer predições precisas voltem sua atenção para LSTM, mas que pesquisadores podem encontrar proveito de combinar Regressão por Vetor Suporte com algoritmos genéticos.

G. Variações de LSTM baseadas em Vetores

O trabalho apresentado em [16] busca realizar uma predição através de vetores que contém as informações da série temporal de uma ação. A ideia da criação de vetores é que os autores buscavam combinar as informações de multiplas ações durante o treinamento, enquanto métodos tradicionais costumam focar em uma ação de cada vez, necessitando que sejam retreinados para cada ação individual. Esses vetores são criados não só combinando multiplas ações, mas também convoluindo-as, reduzindo a taxa de erro causada por informações redundantes e inúteis e também o tempo necessário para treinamento da rede neural.

Os autores então propoem duas novas variações do algoritmo LSTM para tratarem os vetores criados. O A primeira, chamada *Embedded Long Short Term Memory* (ELSTM), utiliza de uma camada que pré-processa a matriz dos vetores de entrada de modo a reduzir suas dimensões através de transformação de matriz. Depois, essa matriz reduzida é processada por três camadas LSTM que são efetivamente a rede neural. Após esse processamento, os resultados são emitidos. Três desses circuitos são utilizados pelos autores para calcular tanto o valor futuro da ação ou se está em têndencia de súbida ou de queda.

A segunda variação, chamada de *Automatic Encoder Long Short Term Memory* (AELSTM), realiza a redução de dimensionalidade por meio Máquina de Boltzmann Restritiva Continua. A ideia por trás dessa implementação é aplicar três Máquinas de Boltzmann Restritivas seguidas para realizar a redução da dimensionalidade. Novamente, três LSTM são aplicadas em sequência a serem treinadas sobre as matrizes reduzidas e as saídas também são o valor futuro da ação e sua tendência de súbida ou de queda.

As redes neurais foram treinadas sobre dados do mercado de ações de Shanghai. Os autores verificaram que seus resultados se apresentaram melhores que os de técnicas de predição tradicionais, o que eles em consideram uma vitória também porque o o mercado chinês não está amadurecido. Em geral, mercados amadurecidos podem ser mais facilmente previstos, como dita a literatura [16].

H. Análise de Sentimento

O artigo [17] realiza uma pesquisa envolved análise de sentimento no treinamento de modelos de aprendizado. Utilizando de *crawlers*, os pesquisadores fizeram varreduras na rede social *Twitter* e no site de notícias *Business Insider* com a finalidade de encontrar postagens relevantes a tendência de súbida ou queda de ações. Os pesquisadores utilizaram do pacote Stanford NLP para fazerem a leitura de linguagem natural. O uso do pacote seu deu em sua habilidade superior a outros algoritmos de ler o contexto de uma mensagem e também no quesito de classificar notícias e mensagens quanto a sua positividade em relação a uma dada ação, com as postagens somando de 0 a 4, indo desde extremamente negativas a extremamente positivas, à pontuação total daquele dia.

Também foram extraídas informações da plataforma Yahoo! Finance quanto a obtenção de séries temporais de ações. Os conjuntos de dados criados para treinamento denotavam os preços de cada dia bem como sua tendência de súbida ou de queda, calculada a partir da diferença do preço de fechamento ao preço de abertura daquele dia, e a positividade extraída a partir da análise de sentimento. Os modelos tinham então como objetivo estimar a tendência futura, seja ela de súbida, neutra ou de queda, da ação a partir das informações que recebiam como entrada. 12 modelos de classificação foram escolhidos pelos pesquisadores a fim de realizar comparação cruzada. A fim de diminuir informações redundantes ou irrelevantes, os autores também aplicaram algoritmos de redução de dimensionalidade / seleção de atributos, como *KBest* e *PCA*, a fim de removê-las dos modelos, com comparações sendo realizadas das precisões dos algoritmos antes e depois do tratamento. Em geral, foi observada melhoria na precisão dos modelos após o uso dos algoritmos de tratamento.

A conclusão dos pesquisadores de [17] foi de que a influência das redes sociais e notícias sobre a tendência das ações pesquisadas se deu principalmente 9 dias após sua postagem, com seu efeito sendo menor quanto mais próxima da data da postagem. O algoritmo que atingiu melhor precisão foi o de Floresta Aleatória devido ao problema combinar dados numéricos e dados nominais, a qual este algoritmo se adapta bem.

IV. Objetivo

O principal objetivo deste trabalho é desenvolver uma rede neural artificial que seja capaz de fazer predições aceitavelmente precisas de ações do mercado financeiro e aplicalá sobre ações financeiras reais. Tais predições deverão ser comparadas as séries temporais de suas respectivas ações de modo a verificar quão preciso é o algoritmo desenvolvido.

V. Metodologia Experimental

O programa desenvolvido neste trabalho deverá ser capaz de realizar os seguintes passos gerais: Adquirir séries temporais de ações financeiras a partir de plataformas online, como Yahoo! Finance e Quandl. As ações financeiras escolhidas serão de empresas grandes brasileiras como Petrobras e Bradesco; Pré processar os dados adquiridos de maneira que possam ser interpretados pela rede neural de maneira útil. Em específico, uma maneira possível é criar listas que contenham os preços de

fechamento de dado período de dias (O número específico de dias ainda deverá ser definido, mas poderá ser tornado mais claro com a definição da autocorrelação dos preços de uma ação) ou utilizar de indicadores como as utilizadas em [4] para melhor representar o conjunto de dados em mãos, de qualquer forma, a saída esperada será o preço de fechamento do dia seguinte ao período definido; Treinar e validar uma rede neural e uma LSTM com base nos dados adquiridos e pré processados. É esperado que elas aprendam as tendências de queda e súbida da série e nos devolvam o preço seguinte a esse período; Gerar uma série temporal que corresponda aos valores previstos pela rede neural e compará-la com a série temporal de entrada e também ao modelo ARIMA; Apresentar os resultados obtidos através de gráficos ou tabelas.

Com isso em mente, utilizaremos da linguagem Python para programar o algoritmo. O motivo sendo que possui várias bibliotecas que facilitarão o trabalho e uma grande comunidade voltada para o campo de Inteligência Artificial, o que será útil caso problemas sejam encontrados problemas durante o desenvolvimento. Além do mais, o código será escrito em um Notebook do Google Colab pelo motivo de que essa plataforma permite apresentar os resultados junto do código e os salva entre sessões, permitindo fácil visualização. Dentre as bibliotecas existentes, aquelas que serão úteis ao algoritmo desenvolvido são: yfinance para obtenção de séries temporais de ações financeiras, numpy para manipulações numéricas em geral, pandas para definição de fichas de dados que conterão as séries temporais obtidas por yfinance, pytorch para definição de uma rede neural e matplotlib para plotar os gráficos resultantes. Potencialmente, as bibliotecas sklearn, que poderá utilizada para auxiliar o treinamento da rede neural, e sweetviz, para visualização dos dados, poderão ser inclusas no projeto.

Em específico, utilizaremos da biblioteca yfinance para recuperar a série temporal dos preços da Petrobras, representada por PETR4SA. De posse dos dados diários das variações de preços, trabalharemos em cima deles para obter diversos indicadores a serem empregados no treinamento da rede neural, incluindo Média Móvel Simples de 10 dias e Média Móvel Exponencial de 10 dias. Unindo os indicadores aos Preços de Fechamento Ajustados (*Adj Close*), modelaremos eles de modo a servirem de entrada para a rede neural. Em específico, dividiremos os dados em *batches* de 5, com cada um dos *batches* de entrada contendo os preços e indicadores de um único dia e os *batches* de saída contendo o preço do dia posterior. A rede neural será modelada em PyTorch, contendo camadas lineares.

A. Base de Dados

A base de dados inclui a série temporal dos preços das ações da Petrobras (PETR4.SA) do ano de 2022 e de Janeiro do ano de 2023. Cada dia inclui, sem dados faltantes, o preço no momento de abertura para compra (Open), o preço mais alto alcançado no dia (High), o preço mais baixo alcançado no dia (Low), o preço no momento em que a compra foi encerrada (Close), o preço de fechamento ajustado pelo mercado (Adj Close) e o número de ações que foram trocadas naquele dia (Volume). Vale notar que o mercado não opera em fins de semana e feriados, por esse motivo, o número total de dados de dias foi de 271. Para verificar a eficácia dos modelos de rede em generalizar para outras ações, foi também inclusa a série temporal da Vale (VALE3.SA) pelo mesmo período, a ser utilizada como um segundo conjunto de teste após as redes neurais terem sido completamente treinadas.

Para enriquecer as informações presentes na base de dados, populamos-a com diversos indicadores que podem ser adquiridos pelos próprios dados. Esses são indicadores tipicamente usados para estimar qual o melhor momento de realizar trocas no mercado financeiro. Nessa lógica, os indicadores inclusos foram Média Móvel Simples de 10 dias (SMA10), Média Móvel Simples de 14 dias (SMA14), Média Móvel Simples de 21 dias (SMA21), Média Móvel Exponencial de 10 dias (EMA10), K% estocástico de 14 dias (K% 14 days), D% estocástico de 3 dias (D% 3 days), Oscilador de Acumulação e Distribuição (A/D Oscillator), R% de Larry William (Larry William R%), Indíce de Força Relativa (RSI), Divergencia e Convergencia da Média Móvel de 9 dias (MACD e MACD-Signal) e Indice do Canal de *Commodities* (CCI). Também foi incluso como dado de entrada o preço de fechamento ajustado do dia atual.

B. Protocolo de Validação

A validação será realizada por validação cruzada (*cross-validation*). Foram separados os 30 dias finais da série temporal como os dias para teste, que de fato verificarão a capacidade da rede neural de prever novos dados. Os dados da VALE3.SA foram inclusos como um segundo conjunto de teste para testar a generalização dos modelos para ações com alta correlação às da PETR4.SA. Os 30 dias anteriores foram separados como dados para validação, assim, comforme a rede neural é treinada sobre os dados de treino, ela é testada sobre os dados de validação para verificar seus acertos. A melhor rede neural adquirida na fase de validação é então salva para ser aplicada posteriormente nos dados de teste. Finalmente, os dados de treinamento compoem os dias remanescentes. Não se optou por embaralhar os dados pois este é um problema que envolve séries temporais, havendo importancia na ordem dos dados.

Para diminuir o trabalho para treinar as redes neurais, os dados foram reunidos em *batches* de 5, onde cada *batch* contém os dados de 5 dias como sendo a entrada e o preço de fechamento ajustado do próximo dia. Dessa forma, a rede neural recebe os dados de 5 dias, realiza suas previsões e então com base no erro desse processo seus pesos são ajustados, não realizando o ajuste de maneira diária. Os dados de validação e de teste estão arranjados da mesma maneira.

Os modelos escolhidos foram ARIMA por previsão rolante, ANN e LSTM. Todos receberão o preço ”Adj Close” como entrada. Como os indicadores são baseados nos preços dos conjuntos de dados, serão alimentados com base na sua contribuição ao aprendizado de cada modelo. Indicadores que revelarem piorar o aprendizado de um modelo serão desconsiderados para seu treinamento. Os modelos ANN e

LSTM também serão verificados sobre o conjunto de dados da VALE3.SA, servindo como conjunto teste. O modelo ARIMA não será aplicado sobre tal por compor somente dados de teste.

As métricas de avaliação escolhidas foram Erro Quadrado Médio (MSE), Raiz do Erro Quadrado Médio (RMSE) e Erro Absoluto Percentual Médio (MAPE).

C. Pipeline Experimental

A Figura 1 apresenta o pipeline experimental para obtenção dos resultados. Inicialmente, preparamos todo o conjunto de dados, incluindo as funções que geram os indicadores utilizados e também funções para definir quais indicadores serão utilizados por cada modelo. Então, realizamos o treinamento e teste da ARIMA para servir de *benchmark*. Completo o treinamento, verificamos a Rede Neural. Preparamos os dados, incluindo os indicadores criados, como entradas e definimos a estrutura da rede neural. O treinamento envolve encontrar a epóca que gera a melhor rede neural na fase de validação e preserva-la. Finalizado o treinamento e validação, plotamos os gráficos e verificamos se a rede neural está relativamente satisfatória na fase de teste. Se não estiver, ajustamos-a e realizamos esse processo novamente. Quando uma rede neural satisfatória é encontrada, procedemos para a LSTM. A LSTM recebe o mesmo tratamento, podendo receber como entrada indicadores diferentes, a depender de se a afetam positivamente. Finalizada a fase de teste da LSTM e plote de seus gráficos, novamente verificamos se esta relativamente satisfatória, realizando ajustes e repetindo o processo caso não. Finalmente, plotamos um gráfico comparando os três modelos na fase de teste.

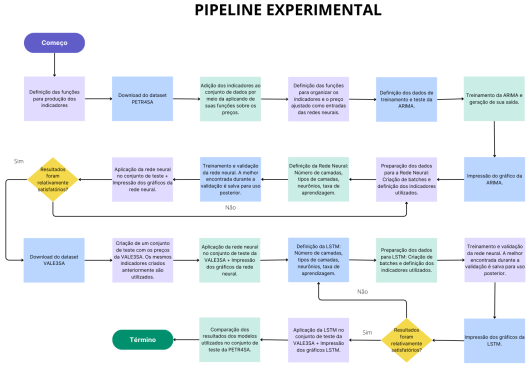

Fig. 1. Pipeline Experimental

VI. Resultados e Discussões

O modelo ARIMA foi alimentado somente com os preços ”Adj Close” passados uma vez que lhe falta a arquitetura de uma rede neural para ajuste de pesos sobre suas entradas. O modelo ANN revelou utilizar melhor os indicadores ”SMA10”, ”SMA14”, ”SMA21” e ”EMA10” durante seu treinamento. Um problema comum encontrado foi definir sua saída como um valor constante quando recebia os outros indicadores como entrada. Uma resposta possível está na pequena arquitetura de camadas ocultas da ANN desenvolvida, dificultando ao modelo se ajustar a um grande número de entradas. Como ”SMA10” e ”EMA10” tem valores próximos de ”Adj Close”, pode-se entender sua maior facilidade em combinar os três para gerar uma saída satisfatória. A LSTM se mostrou capaz de processar todos os indicadores.

A. ARIMA

A Figura 2 apresenta uma comparação dos resultados do modelo ARIMA com a verdade. Podemos observar que a previsão aparenta estar atrasada em relação ao valor real. Disso interpretamos que o modelo está realizando pequenas previsões com base no que conhece do estado atual, tendo dificuldades em de fato prever o valor futuro. Quanto as metricas do modelo, apresentou MSE de 0.328, RMSE de 0.572 e MAPE de 0.022.

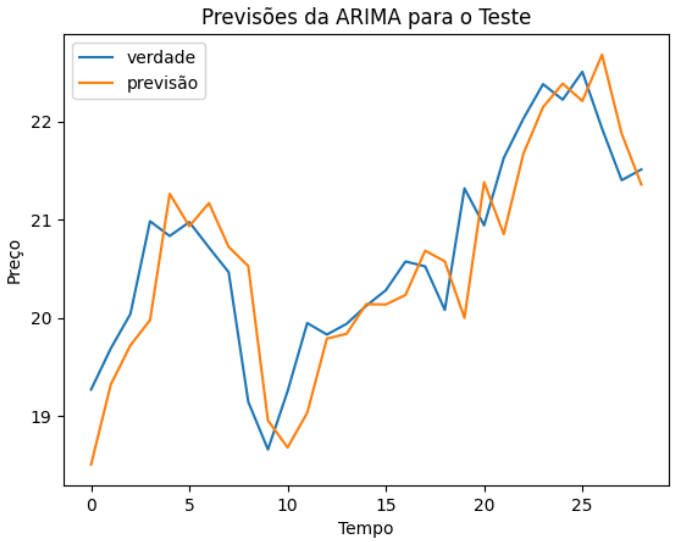

Fig. 2. Resultados do Modelo ARIMA para o Dataset de Teste

B. MLP

A Figura 3 apresenta os resultados do melhor modelo MLP obtido a partir da validação cruzada e os compara com os valores esperados. A arquitetura escolhida para obter os resultados envolveu uma camada ocultas com 40 nós (8 x Número de Entradas) e duas camadas ocultas com 20 nós (4 x Número de Entradas) cada. O otimizador utilizado foi ADAM, provido pela biblioteca PyTorch. A métrica utilizada para cálculo do Erro foi MSELoss, também provida pela biblioteca PyTorch, porém, foi testada a métrica MeanAbsolutePercentageError, provida pela biblioteca torchmetrics, mas seus resultados foram inferiores durante os testes. As métricas obtidas foram MSE de 0.220, RMSE de 0.469 e MAPE de 0.018 para o conjunto teste. O gráfico obtido é bastante similar ao da ARIMA, mas mais próximo da verdade como podemos verificar a partir das métricas. O modelo apresenta problemas em prever o valor exato, mas podemos tirar certo valor de sua previsão de tendências de súbida ou de queda.

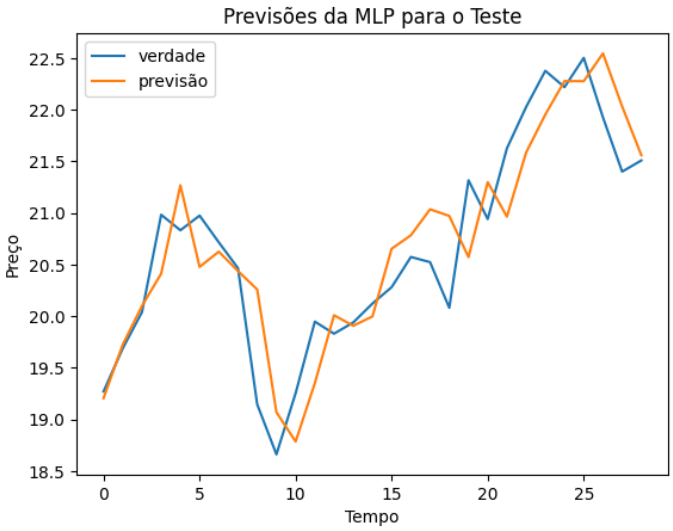

Fig. 3. Resultados do Modelo MLP para o Dataset de Teste

A Figura 4 apresenta os resultados do modelo MLP para o conjunto completo dos dados. Isso nos permite observar como o melhor modelo encontrado para a validação sacrifica sua precisão em prever os dados de treino para obter melhor precisão em prever dados em que não foi treinado.

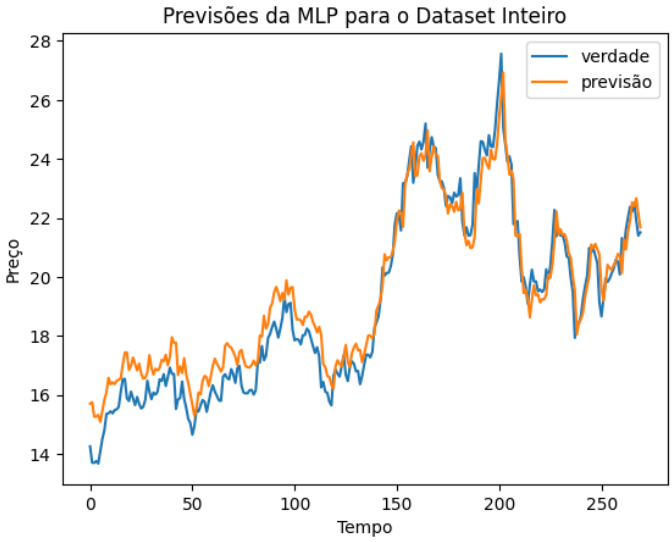

Fig. 4. Resultados do Modelo MLP para o Dataset Completo

Finalmente, aplicamos o modelo treinado sobre as ações de uma empresa distinta, a Vale. Apesar disso, pode ser verificado que o modelo consegue gerar resultados muito similares aos reais, porém erra ao tentar estimar o valor exato, alcançando um valor bastante inferior. Sabe-se que os preços da Vale tem alta correlação com os preços da Petrobras, o que pode ser um motivo pelo qual as tendências de súbida ou de queda puderam ser captadas pelo modelo treinado. Os resultados são apresentados na Figura 5.

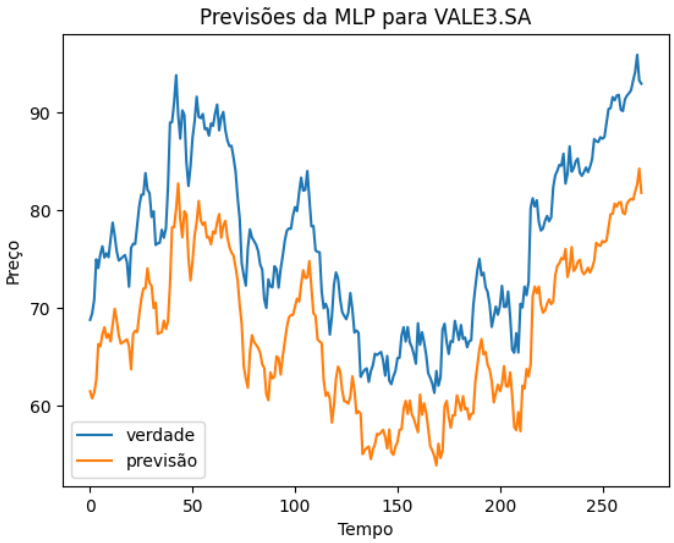

Fig. 5. Resultados do Modelo MLP para o Dataset da Vale

C. *LSTM*

A Figura 6 apresenta os resultados do modelo LSTM pós treinamento para o conjunto de teste. A arquitetura que optamos por manter após os testes foi de 3 camadas LSTM com 325 nós (25 x Número de Entradas) em suas camadas ocultas. Naturalmente, foi observado um treinamento muito mais lento dado não só o número de nós como a maior complexidade das camadas empregadas. O otimizador utilizado foi ADAM, provido pela biblioteca PyTorch. A métrica utilizada para cálculo do Erro foi MeanAbsolutePercentageError, provida pela biblioteca torchmetrics, porém, foi testada a métrica MeanAbsolutePercentageError, provida pela biblioteca PyTorch, mas seus resultados foram inferiores durante os testes. Um possível motivo está no uso de indicadores normalizados como entradas do modelo LSTM. Pelo gráfico da Figura 6, podemos observar que o modelo consegue se aproximar das tendências de súbida ou de queda, mas falha em obter resultados próximos aos valores da verdade. As métricas obtidas foram MSE de 1.0344, RMSE de 1.0171 e MAPE de 0.0438. Um dos possíveis motivos, uma vez que este é um dos melhores modelos descritos na literatura, pode ser o tamanho pequeno do conjunto de dados utilizado. O poder da LSTM está em memorizar saídas e entradas passadas e ajustar seus pesos a fim de determinar sua utilidade em prever valores futuros. Com um conjunto pequeno, ela não pôde identificar todas as variações que poderia encontrar em valores novos. O número grande de entradas e nós serviu para contrabalancear o conjunto relativamente pequeno para o modelo. Durante os testes foi observado que utilizar o mesmo número de entradas que a MLP não alcançava resultados muito bons, mas quando foi aplicado um maior número de entradas, eles melhoraram.

A Figura 7 apresenta os resultados do modelo LSTM para todo o conjunto de dados. Podemos observar que diferente da MLP, o melhor modelo consegue se manter próximo dos

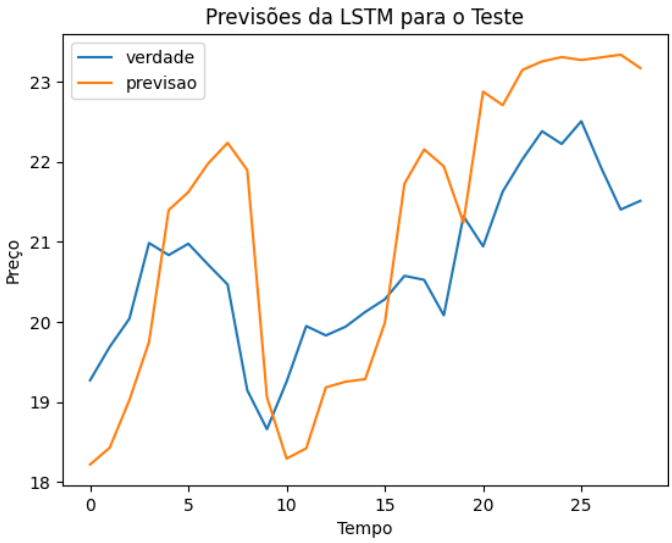

Fig. 6. Resultados do Modelo LSTM para o Dataset de Teste

valores do treino. Uma possível explicação está em sua capacidade de memória, reconhecendo os dados em que foi treinado, mas não os replicando perfeitamente dado que ajustou os pesos conforme sua necessidade em prever os dados de validação. Também pode se notar uma resistência do modelo em replicar os picos presentes nos dados. Isso pode se dar novamente a sua utilização de memória na previsão de dados novos, percebendo o pico como um evento que não duraria.

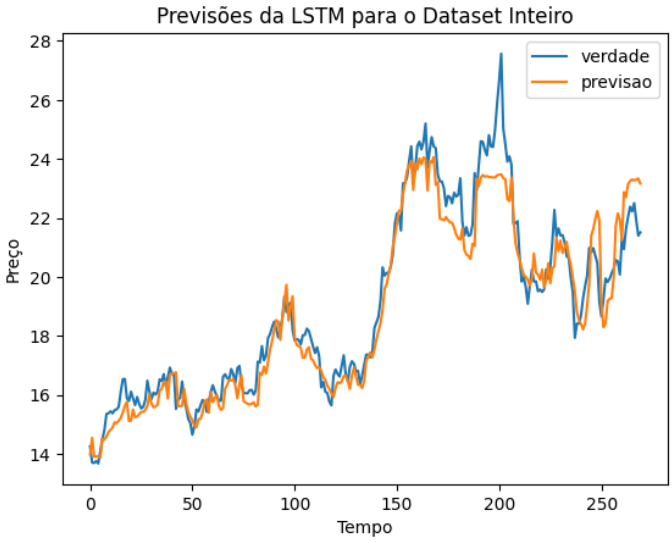

Fig. 7. Resultados do Modelo LSTM para o Dataset Completo

Finalmente, a Figura 8 apresenta os resultados do modelo LSTM em prever o conjunto teste da Vale. Podemos perceber que o modelo falha completamente na previsão dos dados. Os valores previstos se alinham aos preços próximos de 20 encontrados no conjunto da Petrobras, significando que o modelo definiu um limite superior para os dados que espera encontrar. Isso coincide com o comportamento demonstrado pelo modelo quando analisamos o Dataset Completo, uma vez que não replicou os picos encontrados. Dada a presença desse limite superior, o modelo pode estar definindo todo o conjunto de dados como um momento de pico, impedindo sua aproximação dos valores reais. Uma maneira de circumvir esses resultados teria sido normalizar os dados dos conjuntos, melhorando a capacidade do modelo em generalizar.

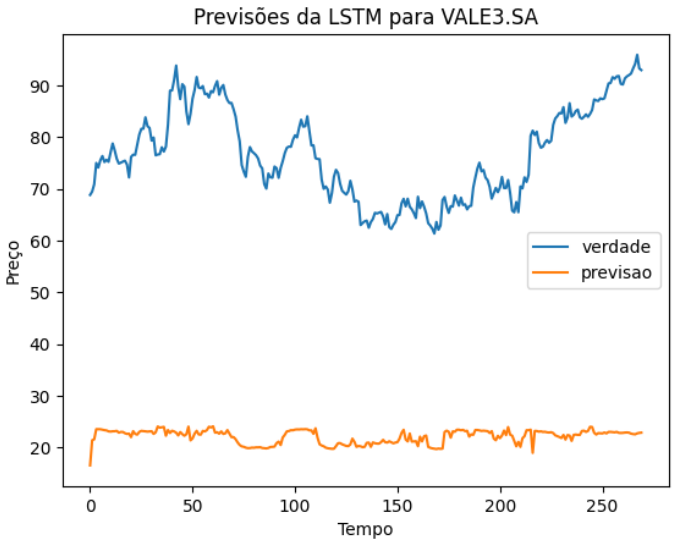

Fig. 8. Resultados do Modelo LSTM para o Dataset da Vale

A Tabela 9 apresenta uma tabela comparando os resultados das métricas para cada modelo. A Figura 10 apresenta um gráfico comparando os resultados de cada modelo para o conjunto de teste a verdade daquele conjunto.

Comparações para o Conjunto de Teste

Modelo	MSE	RMSE	MAPE
ARIMA	0.327631	0.57239	0.0219345
ANN	0.220129	0.469179	0.0182466
LSTM	1.64679	1.28327	0.0559542

Comparações para o Conjunto da Vale

Modelo	MSE	RMSE	MAPE
ANN	88.3368	9.39876	0.119283
LSTM	3039.18	55.1288	0.708229

Fig. 9. Tabela de Comparações

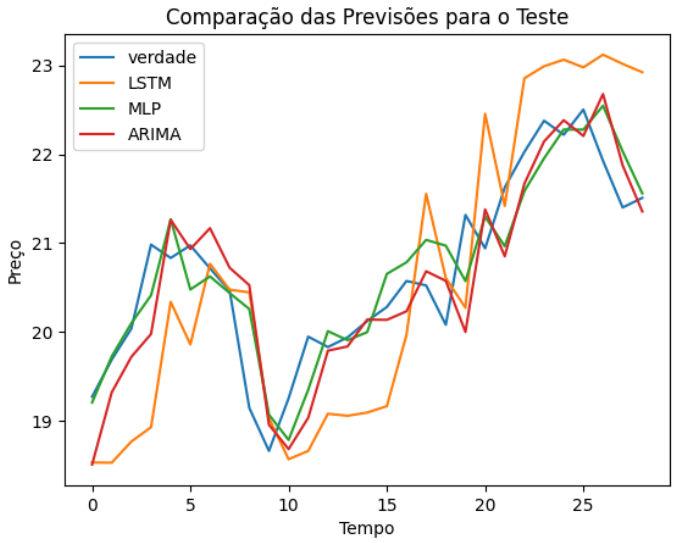

Fig. 10. Gráfico de Comparações

VII. Conclusões e Trabalhos Futuros

O maior sucesso foi obtido pela MLP, que apresentou resultados decentes tanto para o conjunto teste da PETR4.SA e da VALE3.SA, sendo melhor que o modelo ARIMA, usado como benchmark. A LSTM, apesar de ser o melhor modelo segundo a literatura, apresentou resultados piores para ambos os conjuntos testes. Um possível motivo está na dependência da LSTM por ter grandes quantias de dados para gerar bons resultados. A principal vantagem desse modelo está na sua capacidade de memorizar e selecionar quais os melhores resultados passados utilizar na previsão de novos resultados. Dessa forma, ao se deparar com dados completamente novos de uma série na qual não foi treinada, no caso, a Vale, o modelo tenta utilizar os dados da Petrobras para prever novos valores, o que resultam em valores de 20, próximos dos da PETR4.SA, mas muito longes daqueles da VALE3.SA. Enquanto isso, a MLP depende somente dos dados mais recentes que obtem, como o preço atual e médias móveis, permitindo a ela realizar previsões próximas. Dessa forma, podemos concluir que caso este problema envolvesse um número maior de anos ou ações em seu treinamento, poderia ser possível a LSTM obter melhores resultados que as outras ações.

Ainda sim, nenhum dos modelos obteve resultados que permitam uma previsão perfeita do futuro, somente aproximações baseadas no estado presente. Como já era conhecido, é um problema talvez impossível obter modelos computacionais que permitam prever o valor futuro exato, mas pode ser um exercício mais produtivo tentar prever quando haverá súbida ou queda. Uma melhoria relativamente fácil de ser realizada que poderia melhorar os resultados gerais obtidos seria a normalização dos conjuntos utilizados, uma vez que a LSTM falhou em se aproximar dos dados da Vale, aparentando ter definindo um limite superior que os dados poderiam alcançar. Um possível trabalho futuro seria realizar uma classificação de se a ação terá aumento ou decrescimento de seu preço em um futuro próximo. Também permanece a possibilidade de treinar a LSTM em um grande número de outras séries temporais, o que demandaria LSTMs com arquiteturas maiores e maior poder computacional.

References

[1] S. Shunrong, J. Haomiao, Z. Tongda "Stock market forecasting using machine learning algoritm," Department of Electrical Engineering, Stanford University, Stanford, CA, 2012.

[2] W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," Computers & Operations Research, vol. 32, no. 10, pp. 2513–2522, Oct. 2005, doi: https://doi.org/10.1016/j.cor.2004.03.016.

[3] M. Leippold, Q. Wang, and W. Zhou, "Machine learning in the Chinese stock market," Journal of Financial Economics, Aug. 2021, doi: https://doi.org/10.1016/j.jfineco.2021.08.017.

[4] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock market index using fusion of machine learning techniques," Expert Systems with Applications, vol. 42, no. 4, pp. 2162–2172, Mar. 2015, doi: https://doi.org/10.1016/j.eswa.2014.10.031.

[5] M. Moein Aldin, H. Dehghan Dehnavi, and S. Entezari, "Evaluating the Employment of Technical Indicators in Predicting Stock Price Index Variations Using Artificial Neural Networks (Case Study: Tehran Stock Exchange)," International Journal of Business and Management, vol. 7, no. 15, Jul. 2012, doi: https://doi.org/10.5539/ijbm.v7n15p25.

[6] G. Oliveira, M. Pacheco "Mercado financeiro objetivo e profissional," 3ª ed, Editora Fundamento Educacional, pp 20, pp 137-144, pp 154-156, 2017.

[7] Z.-H. Zhou, "Machine learning." Gateway East, Singapore: Springer, 2021.

[8] "Stationary process," Wikipedia, Apr. 20, 2021. https://en.wikipedia.org/wiki/Stationary_process (acesso em Jun. 17, 2023).

[9] "Autoregressive integrated moving average," Wikipedia, Mar. 15, 2023. https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average (acesso em Jun. 17, 2023).

[10] "Long short-term memory," Wikipedia, Nov. 22, 2018. https://en.wikipedia.org/wiki/Long_short-term_memory (acesso em Jun. 19, 2023).

[11] Y. Yetis, H. Kaplan, and M. Jamshidi, "Stock market prediction by using artificial neural network," 2014 World Automation Congress (WAC), Aug. 2014, doi: https://doi.org/10.1109/wac.2014.6936118.

[12] M. Vijh, D. Chandola, V. A. Tikkiwal, and A. Kumar, "Stock Closing Price Prediction using Machine Learning Techniques," Procedia Computer Science, vol. 167, no. 167, pp. 599–606, 2020, doi: https://doi.org/10.1016/j.procs.2020.03.326.

[13] S. Kumar Chandar, M. Sumathi, and S. N. Sivanandam, "Prediction of Stock Market Price using Hybrid of Wavelet Transform and Artificial Neural Network," Indian Journal of Science and Technology, vol. 9, no. 8, Feb. 2016, doi: https://doi.org/10.17485/ijst/2016/v9i8/87905.

[14] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, E. Salwana, and S. S., "Deep Learning for Stock Market Prediction," Entropy, vol. 22, no. 8, p. 840, Jul. 2020.

[15] M. Nikou, G. Mansourfar, and J. Bagherzadeh, "Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms," Intelligent Systems in Accounting, Finance and Management, Dec. 2019, doi: https://doi.org/10.1002/isaf.1459.

[16] X. Pang, Y. Zhou, P. Wang, W. Lin, and V. Chang, "An innovative neural network approach for stock market prediction," The Journal of Supercomputing, vol. 76, no. 3, pp. 2098–2118, 2018. doi:10.1007/s11227-017-2228-y

[17] W. Khan, M. A. Ghazanfar, M. A. Azam, A. Karami, K. H. Alyoubi, and A. S. Alfakeeh, "Stock market prediction using machine learning classifiers and social media, news," Journal of Ambient Intelligence and Humanized Computing, Mar. 2020, doi: https://doi.org/10.1007/s12652-020-01839-w.