Técnicas de memória para previsão de retornos de ativos financeiros com alta volatilidade

João Matheus Del Vecchio França Barbosa, Bruno Franco Vieira, Jean Carlo

November 18, 2024

1 Introdução - Análise, tese, tema e estratégia.

1.1 Análise do mercado e dos ativos escolhidos

As criptomoedas se consolidaram como uma nova forma de moeda eletrônica, com impacto significativo nas economias emergentes e no mercado global. Em 2024, o número de hedge funds especializados em criptomoedas alcançou 300, refletindo um aumento notável no interesse institucional por esses ativos [ADICIONAR REFERÊNCIA]. No entanto, a previsão de retornos e a análise de volatilidade no mercado de criptomoedas apresentam desafios únicos, principalmente devido à extrema flutuação de preços e à complexidade inerente desse mercado, características que o diferenciam dos mercados financeiros tradicionais.

Consequentemente, os modelos econométricos tradicionais de previsão de retornos, que utilizam equações lineares, tendem a apresentar desempenho inferior nesse contexto em comparação com modelos baseados em aprendizado de máquina. Isso ocorre porque os modelos lineares não conseguem capturar adequadamente as dependências não lineares entre as variáveis disponíveis e os retornos futuros, ao contrário das técnicas de Machine Learning, que são mais flexíveis, mas também mais suscetíveis a overfitting [ADICIONAR REFERÊNCIA].

Nos últimos anos, metodologias de aprendizado profundo foram aplicadas à previsão de séries temporais no mercado de criptomoedas. Modelos avançados têm combinado técnicas como camadas convolucionais e LSTM (Long Short-Term Memory). As camadas convolucionais são utilizadas para filtrar ruídos em dados complexos e extrair características relevantes, enquanto as LSTMs capturam padrões sequenciais e dependências de curto e longo prazo. No entanto, estudos de Pintelas et al. e Livieris et al. apontam que esses modelos enfrentam dificuldades ao prever os preços das criptomoedas. Os principais desafios identificados são: (1) as séries temporais de criptomoedas frequentemente se assemelham a processos de Random Walk, o que torna a previsão extremamente complexa; e (2) a presença de erros correlacionados e a falta de estacionaridade nos dados. Séries não estacionárias, caracterizadas por alta volatilidade e tendências, dificultam a confiabilidade dos modelos, uma vez que propriedades como média, variância e frequência variam ao longo do tempo.

Além disso, há outra característica importante desse mercado promissor: a alta correlação entre os retornos de seus principais ativos. Conforme demonstrado na etapa de análise de dados deste relatório [FIGURA TAL] e descrito por [COLOCAR REFERÊNCIA], os retornos das criptomoedas apresentam uma dependência e covariância significativamente maiores entre si em comparação com os ativos do mercado de ações tradicional. Essa correlação é especialmente alta com o Bitcoin, que atua como um indicador de tendência para o mercado de criptomoedas.

1.2 Tese e Estratégia

Com base nos estudos e avanços citados, e considerando as ressalvas mencionadas, a tese de nossa estratégia é a seguinte: estruturando-se em uma arquitetura de rede neural baseada em células Long-Short-Term Memory (LSTM) e utilizando a técnica referenciada no livro de Marco Lopez [ADICIONAR REFERÊNCIA], que aplica diferenciação fracionária em séries temporais para maximizar a memória dos dados enquanto preserva a estacionariedade, além de incorporar séries temporais do retorno de múltiplos ativos para prever o comportamento de uma única moeda, propomos um modelo de previsão de retornos de criptomoedas que explora as características de alta volatilidade, complexidade na dependência temporal e alta correlação entre os ativos.

A estratégia de nossa abordagem consiste em desenvolver um modelo que capture padrões de memória de longo e médio prazo da série temporal, tanto na arquitetura da rede neural quanto no tratamento dos dados. Essa capacidade de memória será essencial para melhorar a previsão, aproveitando a informação histórica de forma eficaz.

Um dos principais desafios será evitar o overfitting, dado que nosso modelo, por ser mais flexível, tem maior propensão a se ajustar excessivamente aos dados de treinamento. Para mitigar esse risco, elaboramos um planejamento experimental robusto, que utiliza apenas dados passados para prever valores futuros e aplica técnicas de sampling para evitar a superespecialização.

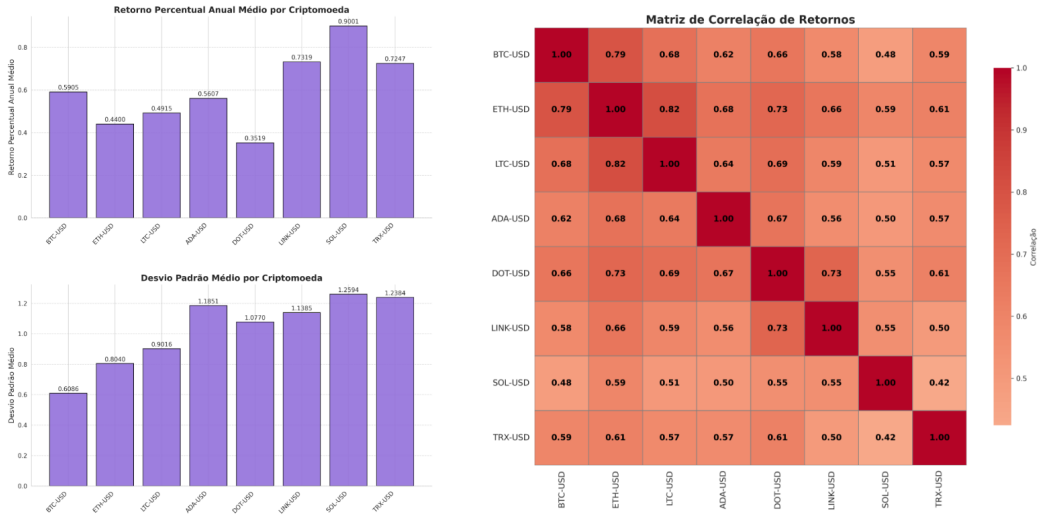

Figure 1: Gráficos de Retorno médio anualizado, Desvio Padrão médio anualizado e Matriz de correlação do portfólio de moedas escolhido

O trading será realizado diariamente, com base nas previsões da rede neural para o retorno esperado do próximo dia de cada uma das 8 criptomoedas selecionadas para o portfólio. A distribuição do capital entre esses ativos buscará maximizar o índice de Sharpe do portfólio final, adotando uma abordagem baseada na teoria de portfólio de Markowitz. Para construir esse portfólio, não apenas o retorno esperado de cada ativo será considerado, mas também a matriz de covariância entre as criptomoedas. Propomos que, enquanto utilizamos técnicas de machine learning para estimar os retornos, a matriz de covariância seja construída com base na covariância populacional estatística dos ativos.

Para comparação dos resultados, utilizaremos como baseline o modelo econométrico de Simple Moving Average (SMA), que assume que o retorno esperado é aproximadamente a média dos retornos dos últimos 30 dias. Esse modelo seleciona para o portfólio apenas as moedas com valor esperado positivo; caso nenhuma moeda apresente retorno positivo, não haverá operação.

Definida a estratégia, nosso robô de trading foi batizado como "LSTMinator: o exterminador de preços futuros", uma referência ao famoso androide da cultura pop, o Exterminador do Futuro (Terminator), combinada com o núcleo da nossa arquitetura de rede neural, a célula LSTM.

Por fim, todo o projeto, que envolveu três etapas principais — Escolha da Arquitetura, Otimização de Hiperparâmetros e Backtest — foi implementado na plataforma Jupyter Notebook, utilizando bibliotecas como Yahoo Finance (para obtenção de dados financeiros), TensorFlow Keras (para modelos de rede neural), NumPy e Pandas (para álgebra linear e manipulação de dados), e PyMoo (para algoritmos genéticos). O versionamento do código foi realizado inteiramente no GitHub.

2 Dados

2.1 Coleta de Dados e Análise exploratória

Os dados coletados da biblioteca do Yahoo Finance consistem em séries temporais do preço de abertura diário das seguintes criptomoedas: Bitcoin (BTC), Ethereum (ETH), Litecoin (LTC), Cardano (ADA), Polkadot (DOT), Chainlink (LINK), Solana (SOL) e TRON (TRX). O período de coleta abrange de 21 de agosto de 2014 a 31 de dezembro de 2023.

Pelas Figuras 1 e 2 podemos observar que, ao comparar o desvio padrão médio das 9 criptomoedas selecionadas com o das 9 ações integrantes do índice SP500, a volatilidade (medida aqui pelo desvio padrão) média das criptomoedas, de 1.02, é significativamente superior à volatilidade média das ações, que é de 0.33. Essa evidência corrobora a análise de mercado feita na introdução, que destaca a alta volatilidade presente no mercado de criptomoedas. Além disso, comparando as Figuras1 com 2 , podemos notar que o retorno percentual médio anualizado dos criptoativos também é consideravelmente maior que o das ações, indicando um mercado com elevado potencial de ganhos.

Por fim, a matriz de correlação entre as criptomoedas apresenta valores mais elevados, especialmente no caso do Bitcoin, que possui, em média, uma correlação de 0.63 com as demais moedas. Esse comportamento não é comumente observado no mercado de ações tradicional, o que reforça a hipótese de utilizar séries temporais de

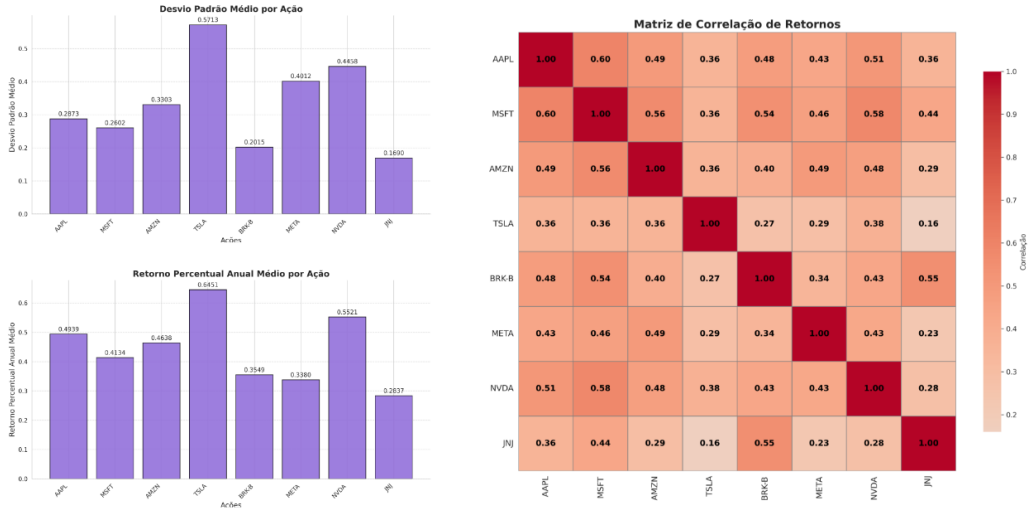

Figure 2: Gráficos de Retorno médio anualizado, Desvio Padrão médio anualizado e Matriz de correlação do portfólio de ações do S&P500

outras criptomoedas além daquela que se pretende prever. Para a previsão do retorno do Bitcoin, utilizamos como série auxiliar a do Ethereum.

A partir dessas séries temporais, criamos as primeiras tabelas, que posteriormente serão tratadas para alimentar a Rede Neural. Nessas tabelas, cada linha representa um dia do nosso espaço amostral, sendo que as duas primeiras colunas correspondem às features de determinada moeda naquele dia. Para cada ativo, as features selecionadas foram o logaritmo natural do preço da moeda específica e o logaritmo natural do preço do Bitcoin. A escolha de utilizar a série do Bitcoin como auxiliar se justifica pelo fato de esta moeda apresentar a maior correlação com as demais, atuando, de certa forma, como um indicativo de tendência de mercado.

Optou-se por não incluir mais features no modelo, pois isso poderia aumentar excessivamente a dimensionalidade dos dados, comprometendo a capacidade preditiva. A terceira coluna representa o valor-alvo (target) de cada observação, que é o retorno logarítmico da moeda no dia seguinte — a variável que desejamos prever.

$$y_t = Retorno_T^{cripto} = ln(\frac{P_t^{cripto}}{P_{t-1}^{cripto}})$$

Na etapa de avaliação das possíveis arquiteturas para a Rede Neural, utilizou-se o período de 1º de junho de 2017 até o final de outubro de 2019, correspondendo a um total de 882 dias. Já para a alocação de portfólio e a realização do backtest, foram considerados os dados de janeiro de 2020 até 31 de dezembro de 2023, abrangendo um total de 1.460 dias.

2.2 Tratamento dos Dados

O tratamento dos dados se resumiu em duas etapas: Diferenciação Fracionária e Normalização.

2.2.1 Diferenciação Fracionária

Com base no capítulo "Fractionally Differentiated Features", a técnica de diferenciação fracionária se baseia no conceito de que, ao utilizarmos a diferença do preço logarítmico, $\ln(P_t) - \ln(P_{t-1})$, em vez do preço logarítmico em si como novos dados da nossa série temporal, realizamos uma diferenciação de primeira ordem. A premissa por trás dessa abordagem é manter a estacionariedade da série temporal, ou seja, garantir que a distribuição da série se mantenha aproximadamente constante ao longo do tempo. No entanto, ao seguir essa estratégia, estaríamos perdendo de forma excessiva e desnecessária a memória da série original.

Como descrito por Lopez, embora a estacionariedade seja uma propriedade necessária para fins inferenciais, raramente, em processamento de sinais, desejamos que toda a memória seja apagada, uma vez que essa memória é a base do poder preditivo do modelo. Por exemplo, modelos de equilíbrio (estacionários) dependem dessa memória para avaliar o quanto o processo de preços se afastou do valor esperado de longo prazo, a fim de gerar previsões mais precisas.

O dilema que enfrentamos é que os retornos são estacionários e sem memória, enquanto os preços possuem memória, mas são não estacionários. Portanto, buscamos uma forma de encontrar um intermediário entre a diferenciação completa, que gera uma série de retornos logarítmicos, e a diferenciação de ordem zero, que resulta na série do preço logarítmico.

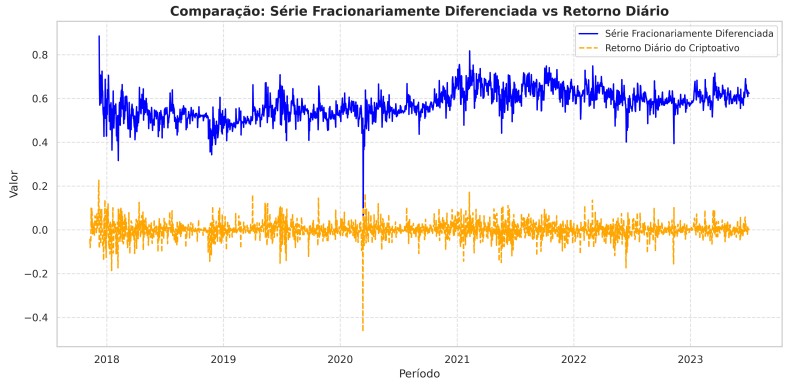

Figure 3: Comparação da série parcialmente fracionária com uma série totalmente diferenciada

A fim de encontrar um meio-termo entre estacionariedade e preservação de memória, vamos aplicar a técnica de diferenciação fracionária.

Considere o operador de diferenciação B, o qual, aplicado em uma matriz de atributos X_t, resulta em $B^k X_t = X_{t-k}$. Dessa forma, para $k = 1$, a expressão representa o retorno $(1 - B)^1 X_t = X_t - BX_t = X_t - X_{t-1}$, e para $k = 0$, a série de preços é dada por $(1 - B)^0 X_t = X_t$.

Assim, para valores de k entre 0 e 1, obtemos uma diferenciação fracionária. Aplicando o teorema binomial em sua forma expandida, obtemos a seguinte expressão:

$$(1 - B)^d = 1 - dB + \frac{d(d-1)}{2!}B^2 - \frac{d(d-1)(d-2)}{3!}B^3 + \dots$$

Nos permitindo reescrever o problema como

$$\bar{X}_t = \sum_{k=0}^{\infty} \omega_k X_{t-k}$$

$$\omega = \left\{1, -d, \frac{d(d-1)}{2!}, \frac{d(d-1)(d-2)}{3!}, \dots, (-1)^k \prod_{i=0}^{k-1} \frac{d-i}{k!}, \dots\right\}$$

O valor de w_k pode ser calculado de forma iterativa pela seguinte fórmula:

$$w_k = -w_{k-1} \frac{d - k + 1}{k}$$

Para o tratamento dos nossos dados, considerando a tabela X_t, que é utilizada para prever o retorno no dia t, escolhemos o menor valor de d que garantisse um teste ADF de estacionariedade com p-valor menor que 0,01, com o objetivo de encontrar o intermediário desejado. Além disso, utilizamos um vetor de pesos de tamanho 10 para todas as nossas diferenciações.

As Figuras 3 e 4 representam a comparação de uma diferenciação fracionária para valores de d iguais a 0, 1 e 0.66, sendo este último o valor de d que garante um p-valor menor que 0,01. Os valores do p-valor e do t-statistic para as séries resultantes dos três valores de d estão contemplados na Tabela 1.

Dessa forma, concluímos que a diferenciação total, de fato, exclui a memória de forma exagerada, uma vez que para $d = 0.6$ já obtemos uma estacionariedade satisfatória para a aplicação de modelos de machine learning de forma eficaz. Além disso, nas Figuras 4 e 3, podemos perceber como a série diferenciada com $d = 0.6$ parece apresentar tanto o comportamento de ter uma média constante quanto de seguir os valores de tendência da série de preço original.

Valor de d	ADF t-statistic	ADF p-value
0.6	-3.5067	0.0078
1	-31.1617	0.0000
0	-1.1963	0.6752

Table 1: Resultados do Teste ADF para diferentes valores de d aplicados na Série temporal do BTC-USD até o dia 03 de julho de 2023

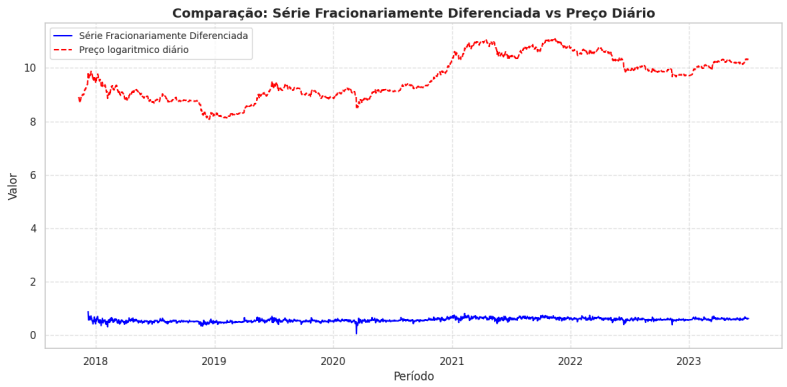

Figure 4: Comparação da série parcialmente fracionária com uma série nada diferenciada

2.2.2 Normalização

A etapa de normalização consistiu em, para cada conjunto de dados que seria enviado para alimentar o modelo, normalizar os dados para média 0 e variância 1, por meio de uma simples substituição:

$$x_{\text{normalizado}} = \frac{x - \mu}{\sigma}$$

onde μ representa a média e σ a variância dos dados originais.

3 Planejamento Experimental

Tanto na etapa de avaliação das possíveis arquiteturas quanto na etapa de backtest, a sequência experimental segue o mesmo modelo:

Suponhamos que estamos no dia i e queremos prever o valor do retorno desse dia. Primeiramente, vamos selecionar os últimos W dias anteriores ao dia i. Cada um desses W dias possui seus próprios atributos, de acordo com a estrutura definida na etapa de Dados, além do valor-alvo, que é o retorno logarítmico do dia.

Assim, trabalhamos com uma tabela de W linhas. A partir dessa tabela inicial, vamos criar γ novos conjuntos de dados, cada um com um número de linhas $n < W$. Esses conjuntos serão gerados por meio da técnica de Bagging, que é um método de amostragem direcionado para reduzir o overfitting do modelo. No Bagging, selecionamos aleatoriamente n linhas do conjunto de dados original para criar um novo conjunto, e esse processo é repetido com reposição até termos γ novas tabelas.

Seguindo a literatura, vamos considerar $\gamma = 10$ e $n = 0.9 \times W$.

Em seguida, treinamos γ modelos para cada uma dessas amostras, de forma que o valor previsto será a média dos valores previstos de todas as amostras do conjunto. Isso ajuda a reduzir a chance de overfitting a um conjunto de dados específico.

O problema dessa abordagem é que ela aumenta, ao menos, em 10 vezes o tempo de treinamento, o que é especialmente prejudicial no *backtest*. Por esses motivos de eficiência temporal, a etapa de bagging foi realizada apenas no período dos 100 dias mais recentes durante a etapa de backtest.

Com isso explicado, temos dois principais hiperparâmetros em nosso modelo: o número de dias anteriores considerados como *features* w de cada amostra, e o número de dias anteriores utilizados para treinar o modelo, W. Para o modelo adotado, escolhemos $w = 30$ e $W = 300$. Ou seja, estamos assumindo que o retorno do próximo dia pode ser descrito pelas informações da nossa série temporal dos últimos 30 dias, e também assumimos a hipótese de que a série se comporta de maneira semelhante nos últimos 300 dias.

4 Avaliação de Possíveis Arquiteturas

Foram propostas 5 arquiteturas baseadas em camadas LSTM. Posteriormente, avaliamos o desempenho delas na previsão do retorno percentual das moedas BTC-USD e LTC-USD no período de 01 de junho de 2017 até 31 de outubro de 2019. Além disso, avaliamos o desempenho do modelo baseline como forma de comparação.

As 5 arquiteturas avaliadas foram as seguintes:

Arquitetura 1 e 1* - Arquitetura baseada no trabalho TAL, composta por uma camada LSTM com 1 neurônio, seguida de um dropout de 0.2 e uma camada densa com 1 neurônio. Testamos também a versão 1*, que consiste na mesma arquitetura, mas considerando apenas a série temporal da própria moeda.

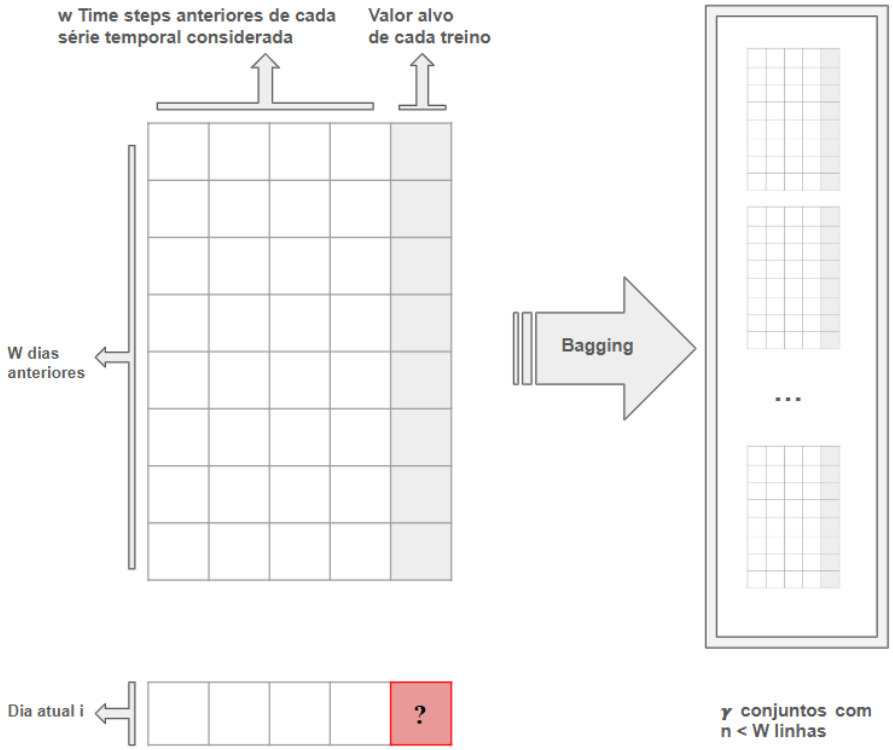

Figure 5: Planejamento Experimental

Arquitetura 2 e 2* - Uma camada LSTM de 64 neurônios, seguida de Dropout 0.3, Batch Normalization, outra camada LSTM com 32 neurônios, Dropout 0.3, e, por fim, uma camada densa com 1 neurônio. Novamente, testamos a versão 2*, que usa apenas a série temporal da própria moeda como *feature*.

Arquitetura 3 - Baseada no trabalho TAL, consiste em uma camada CNN com 15 neurônios, seguida de uma camada de Pooling, uma camada LSTM com 50 neurônios, Batch Normalization, Dropout 0.5, uma camada densa com 128 neurônios, outro Batch Normalization, Dropout 0.2 e, por fim, uma camada densa com 1 neurônio.

Para todos os modelos, utilizamos o otimizador RMSprop com um learning rate de 1×10^{-4}, e um método de parada baseado na perda no conjunto de validação. Além disso, os valores de $W = 300$ e $w = 14$ foram definidos.

Antes de apresentar os resultados, definimos as métricas utilizadas para a avaliação dos modelos:

Erro Absoluto Médio (MAE): $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$ onde y_i é o valor real e $\hat{y}_i$ é o valor predito. Precisão: $\text{Pre} = \frac{\text{VP}}{\text{VP+FP}}$ onde VP é o número de verdadeiros positivos e FP é o número de falsos positivos. Acurácia: $\text{Acc} = \frac{\text{VP+VN}}{\text{Total}}$, onde VN é o número de verdadeiros negativos e Total é o número total de observações. Sensibilidade: $\text{Sen} = \frac{\text{VP}}{\text{VP+FN}}$, onde FN é o número de falsos negativos. Especificidade: $\text{Spe} = \frac{\text{VN}}{\text{VN+FP}}$

Os resultados comparativos entre o Baseline e diferentes arquiteturas estão apresentados na tabela abaixo:

	MAE		**Precisão**		**Acurácia**		**Sensibilidade**		**Especificidade**	
Modelo	**BTC**	**LTC**	**BTC**	**LTC**	**BTC**	**LTC**	**BTC**	**LTC**	**BTC**	**LTC**
Baseline	0,03143	0,04399	0,56329	0,46660	0,52828	0,49434	0,55970	0,46890	0,49015	0,51828
Arquitetura 1*	0,03073	0,04236	0,55152	0,48104	0,51750	0,50965	0,57353	0,48798	0,45185	0,52903
Arquitetura 1	0,03052	0,04268	0,55346	0,50127	0,51760	0,52890	0,55462	0,47596	0,47407	0,57634
Arquitetura 2*	0,03039	0,04206	0,53846	0,46499	0,50796	0,48409	0,61895	0,62410	0,37778	0,35914
Arquitetura 2	0,03060	0,04201	0,52347	0,47419	0,49200	0,49970	0,61053	0,59615	0,34975	0,40860
Arquitetura 3	0,03000	0,04230	0,58046	0,48141	0,55550	0,50796	0,63790	0,56145	0,45926	0,46025

Table 2: Resultados comparativos entre o baseline e diferentes arquiteturas.

Podemos observar que, entre as Arquiteturas 1 e 2 e suas respectivas versões com asterisco, a Acurácia e a Precisão para o LTC-USD foram consideravelmente melhores. Isso pode indicar que a adição da série temporal do BTC como *feature* é especialmente eficaz para o desempenho de outras moedas. Além disso, apenas a Arquitetura 3 obteve resultados superiores tanto em MAE quanto em Precisão e Acurácia em relação ao Baseline. Vale ressaltar

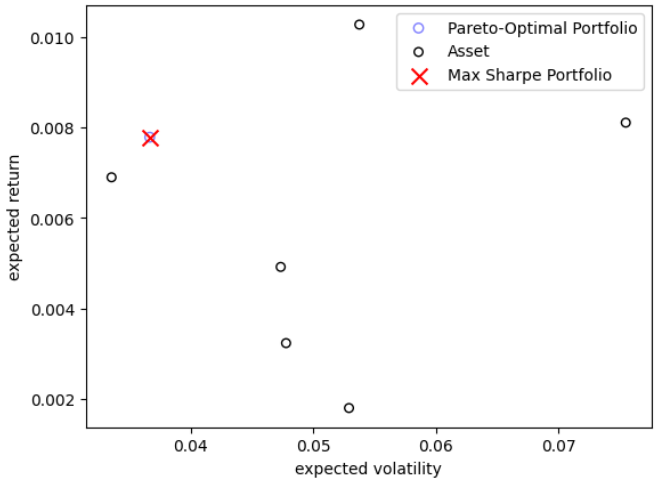

Figure 6: Exemplo de um Portfólio Ótimo gerado para um dia de trade

que, dado que o caráter do nosso modelo é baseado na compra de ações que apresentem um valor de retorno positivo, a métrica de Precisão é particularmente importante em comparação com as outras.

Dessa forma, a Arquitetura 3 se destaca como a mais satisfatória e será a escolhida para a estratégia, embora os resultados possam ser imprecisos devido aos valores próximos entre as arquiteturas, ao fato de o período de teste não ser necessariamente representativo dos períodos futuros e à sensibilidade desses modelos ao ruído. Outra alternativa viável seria a Arquitetura 1. No entanto, para esta estratégia, optamos por adotar modelos mais flexíveis, embora isso nos deixe mais suscetíveis a um maior overfitting.

5 Construção de Portfólio

Dado um dia i, nosso modelo gera um vetor contendo os valores esperados de retorno de cada moeda para o próximo dia. A distribuição do capital no portfólio será feita com base no portfólio que maximiza o Índice de Sharpe, considerando uma taxa de retorno livre de risco anual de 10%.

O Índice de Sharpe é dado pela seguinte equação:

$$S = \frac{E[R_p] - R_f}{\sigma_p}$$

Onde $E[R_p]$ é o retorno esperado de um determinado portfólio ω, R_f é a taxa livre de risco, e σ_p é o desvio padrão (ou volatilidade) do portfólio. Para calcular os pesos do portfólio ótimo de cada dia, utilizamos uma técnica de algoritmo genético implementada pela biblioteca `pymoo`, aliada às equações de construção de portfólio de Markowitz.

Vetor x como sendo o vetor contendo os pesos de cada ação no portfólio:

$$X = [w_1, w_2, ..., w_i, ..., w_N]^T$$

O vetor m contem o retorno percentual esperado de cada cripto:

$$M = [m_1, ..m_i, ..., m_N]^T$$

O valor esperado do retorno percentual do portfólio será:

$$E[r_P] = E[\sum_{i=1}^{N} w_i r_i] = \sum_{i=1}^{N} w_i m_i = \langle X, M \rangle$$

Além disso, o risco desse nosso portfólio é:

$$s_P = \sqrt{\sum w_i^2 s_i^2 + \sum \sum_{i \neq j} 2 w_i w_j Cov_{i,j}} = \sqrt{X^T * Cov * X}$$

A imagem 6 representa, em vermelho (X), um exemplo de portfólio ótimo para um dia de trading utilizado no backtest. Observe que o valor esperado utilizado no algoritmo é exatamente o valor previsto pela Rede Neural.

Devido ao custo computacional de se construir um portfólio para cada dia do backtest, o método utilizado como Baseline não se baseou em uma construção de portfólio como o descrito acima. Em vez disso, escolheu-se simplesmente a moeda com o maior valor esperado previsto. Embora essa simplificação reduza um pouco a

Criptomoeda	Real Subidas (%)	Predito Subidas (%)	MAE	Acurácia (Acc)
BTC-USD	1092 (51.51%)	1235 (58.25%)	0.02319	0.4938
ETH-USD	1094 (51.60%)	1223 (57.69%)	0.02981	0.4871
LTC-USD	1089 (51.37%)	1079 (50.90%)	0.03265	0.5012
ADA-USD	1075 (50.73%)	1036 (48.87%)	0.03541	0.5234
DOT-USD	720 (49.93%)	640 (44.38%)	0.03691	0.5584
LINK-USD	1091 (51.46%)	1197 (56.46%)	0.04151	0.4516
SOL-USD	789 (50.13%)	890 (56.54%)	0.04777	0.5034
TRX-USD	1133 (53.47%)	1250 (58.96%)	0.02860	0.4701

Table 3: BackTest LSTMinator

Criptomoeda	Sensibilidade (Sen)	Especificidade (Spe)	Precisão (Pre)
BTC-USD	0.6026	0.6026	0.5328
ETH-USD	0.5905	0.5905	0.5282
LTC-USD	0.5124	0.5124	0.5171
ADA-USD	0.5005	0.5005	0.5193
DOT-USD	0.4611	0.4611	0.5188
LINK-USD	0.5500	0.5500	0.5013
SOL-USD	0.5894	0.5894	0.5225
TRX-USD	0.5914	0.5914	0.5364

Table 4: Métricas adicionais para BackTest LSTMinator.

relevância de alguns resultados obtidos, ela também abre espaço para futuros estudos e melhorias na estratégia proposta.

Por fim, tanto para o LSTMinator quanto para o Baseline, caso o valor esperado previsto do portfólio seja inferior a um valor de corte previamente determinado, o modelo não realizará compras naquele intervalo. Essa medida serve como uma estratégia conservadora, protegendo o modelo contra pequenas flutuações em dias nos quais ele não possui "certeza" sobre as previsões.

6 Backtest

Realizaremos o backtest de nossa estratégia no período de 1º de dezembro de 2019 até 31 de dezembro de 2023. Devido à grande quantidade de amostras e ao fato de que será necessário criar uma nova rede neural para cada moeda e para cada dia, a complexidade temporal do treinamento se tornou extremamente alta, mesmo com a aplicação de técnicas de paralelismo para processar todas as moedas simultaneamente.

Os resultados da aplicação da arquitetura escolhida, juntamente com o modelo de portfólio e o tratamento de dados descrito nas seções anteriores, estão ilustrados na Figura 8 e nas Tabelas 3 e 4. Para fins de comparação, os resultados do modelo baseline podem ser visualizados na Figura 9 e na Tabela 5.

A precisão média da nossa estratégia se apresentou superior à da maioria dos ativos do nosso portfólio, mantendo-se consistentemente acima de 50%, o que é altamente desejável para um modelo de compra como o nosso. Além disso, observamos que o Erro Absoluto Médio (MAE) do LSTMinator foi sempre inferior ao do Baseline para todas as moedas analisadas.

Dessa forma, ao avaliarmos as métricas propostas, a estratégia do LSTMinator se mostrou superior ao modelo econométrico em todos os quesitos.

Em relação ao lucro cumulativo durante o período de trading, o LSTMinator apresentou um crescimento moderado nos primeiros anos, embora sempre positivo e sem sinais de quedas bruscas ou episódios negativos. No entanto, no início de 2024, justamente quando o erro absoluto do LSTMinator foi o menor de todos, a curva de lucro cresceu consideravelmente. Contudo, não podemos afirmar com certeza que este comportamento seja constante, e o mais razoável é considerar que seja algo específico deste conjunto de dados. Mesmo com essa observação, o retorno ainda foi bastante interessante, especialmente considerando que não aplicamos a técnica de bagging em todo o backtest. Se tivéssemos utilizado o bagging de forma integral, o overfitting seria reduzido, o que poderia ter melhorado ainda mais os resultados.

Adicionalmente, ao compararmos com a estratégia de Média Móvel Simples (SMA), observamos que o lucro no modelo LSTM foi muito menos volátil, ao mesmo tempo que consistentemente maior. Vale ressaltar, no entanto,

Figure 8: Backtest LSTMinator

Figure 9: Backtest Baseline

que o modelo Baseline não contou com uma construção robusta de portfólio, o que pode ter afetado negativamente os resultados dessa estratégia. No entanto, a análise dos indicadores apresentados na Tabela 5 revela que, mesmo sem uma construção de portfólio sofisticada, o LSTMinator ainda superou o Baseline em termos de precisão e erro absoluto, destacando sua superioridade.

Por fim, é importante observar que, embora o lucro da nossa estratégia seja expressivamente alto (cerca de 1200% ao final do período), estamos lidando com ativos que, por si só, passaram por uma valorização significativa durante esse período, o que implica que quase qualquer estratégia de investimento teria gerado lucros consideráveis. Para exemplificar, o Bitcoin, por exemplo, teve um aumento percentual de 535,8% durante o intervalo de tempo do nosso backtest. Assim, uma estratégia simples de **buy and hold** (comprar e manter) também teria gerado um lucro substancial. No entanto, o modelo LSTMinator foi capaz de superar esse retorno, demonstrando a eficácia da estratégia proposta.

7 Conclusão e Trabalhos Futuros

A estratégia baseada em técnicas de memória de longo prazo para redes neurais na previsão de valores futuros de séries temporais obteve bons resultados no período de backtesting, superando consideravelmente o modelo utilizado como baseline e gerando lucros superiores aos de estratégias simples de **buy and hold** durante o mesmo intervalo. Esses resultados indicam que o estudo e a investigação de técnicas de memorização no modelo — tanto pelo tratamento das features quanto pela arquitetura da rede neural — são áreas promissoras e que merecem ser exploradas de forma mais aprofundada.

Entretanto, é importante destacar as limitações dessa estratégia: o possível overfitting nos dados de treinamento

Table 5: Backtes para o Baseline

Crypto	MAE	Acc	Sen	Spe	Pre
BTC-USD	0.0241	0.4887	0.5648	0.4547	0.5160
ETH-USD	0.0320	0.4785	0.5591	0.4449	0.5189
LTC-USD	0.0350	0.4731	0.5000	0.4715	0.5013
ADA-USD	0.0378	0.5075	0.5302	0.4919	0.5215
DOT-USD	0.0405	0.4815	0.4629	0.5000	0.4803
LINK-USD	0.0438	0.4458	0.5176	0.4299	0.4976
SOL-USD	0.0523	0.4996	0.5289	0.4851	0.5078
TRX-USD	0.0306	0.4552	0.5534	0.4142	0.5334

devido à alta flexibilidade do modelo, a complexidade temporal envolvida no backtesting, a escolha subótima dos hiperparâmetros e o portfólio limitado de criptoativos selecionados.

Essas limitações, porém, não representam obstáculos intransponíveis para a aplicação da estratégia, mas sim indicam oportunidades para o desenvolvimento de mais pesquisas em torno deste tema e das abordagens propostas.

Como sugestões para trabalhos futuros, recomendamos as seguintes iniciativas: aplicar a técnica de bagging em todo o dataframe para produzir resultados mais confiáveis; comparar os resultados obtidos com modelos econométricos que também utilizam uma construção de portfólio de Markowitz; otimizar os hiperparâmetros para avaliar o desempenho do modelo em diferentes configurações; e realizar um estudo mais aprofundado sobre a etapa de diferenciação fracionária nas séries temporais.

References

[1]Bouri, E., Lau, C.K.M., Lucey, B., Roubaud, D. (2019). Trading volume and the predictability of return and volatility in the cryptocurrency market. Financial Research Letters, 29, 340–346.

[2]Bouri, E., Roubaud, D., Shahzad, S.J.H. (2020). Do bitcoin and other cryptocurrencies jump together? Quarterly Review of Economics and Finance, 76, 396–409.

[3]Bouri, E., Vo, X.V., Saeed, T. (2020). Return equicorrelation in the cryptocurrency market: analysis and determinants. Finance Research Letters, 101497. https://doi.org/10.1016/j.frl.2020.101497.

[4]Cheng, J., Dong, L., Lapata, M. (2016). Long short-term memory networks for machine reading. arXiv preprint arXiv:1601.06733.

[5]Fang, F., Ventre, C., Basios, M., Kanthan, L., Martinez-Rego, D., Wu, F. (2024). Cryptocurrency trading: a comprehensive survey.

[6]Alessandretti, L., ElBahrawy, A., Aiello, L. M., Baronchelli, A. (2018). Anticipating cryptocurrency prices using machine learning. Scientific Reports, 8(1), 12044. https://doi.org/10.1038/s41598-018-30256-9.