Universidade Federal do Parana
Setor de Ciências Exatas
Departamento de Estatística
Programa de Especialização em Data Science e Big Data

Guilherme José Garmatter Rocha

Trading de alta frequência com redes neurais e codificação de séries temporais em imagens

Curitiba
2023

Guilherme José Garmatter Rocha

Trading de alta frequência com redes neurais e codificação de séries temporais em imagens

Monografia apresentada ao Programa de Especialização em *Data Science* e *Big Data* da Universidade Federal do Paraná como requisito parcial para a obtenção do grau de especialista.

Orientador: Prof. Dr. Anderson Ara

Curitiba
2023

Trading em alta frequência com redes neurais e codificação de séries temporais em imagens

Guilherme José Garmatter Rocha[1], Anderson Ara[1]

[1]Departamento de Estatística, Universidade Federal do Paraná Rua Evaristo F. F. da Costa 418, Jardim das Americas, 82590-300, Curitiba, PR, Brasil * †

Resumo: Neste trabalho utilizamos técnicas para conversão de séries temporais em imagens GAF-GASF-MTF em conjunto com um *ensemble* de redes neurais convolucionais (CNN) para obter uma estratégia de arbitragem/*trading* de alta frequência viável. Apesar da acurácia relativamente baixa obtida pelo modelo, que ficou entre 52% e 54% para os dados de teste, podemos observar que a estratégia foi bastante lucrativa durante o período analisado. Também podemos observar que o uso de *thresholds* de decisão podem aumentar a acurácia do modelo, embora não resultem, necessariamente, em um modelo mais lucrativo devido a menor quantidade de operações realizadas no período.

Palavras-chave: codificação de series temporais em imagens, redes neurais convolucionais, CNN, classificação de imagens, *trading*, arbitragem, alta frequência, criptomoedas

Abstract: In this study, we employed techniques for converting time series into GAF-GASF-MTF images and an ensemble of convolutional neural networks (CNN) to develop a viable high-frequency arbitrage/trading strategy. Despite the relatively low accuracy achieved by the model, ranging from 52% to 54% for the test data, we observed that the strategy was highly profitable during the analyzed period. We could also observe that the use of decision thresholds can increase the accuracy of the model, although it does not necessarily lead to a more profitable model due to the lower number of operations performed.

Keywords: time series to image encoding, Convolutional neural networks, CNN, image classification, high frequency trading

1. Introdução

A prática de comprar e vender ativos com fins lucrativos, também chamada de arbitragem ou *trading*, é antiga e pôde ser observada desde o descobrimento do Brasil com as grandes navegações. Com a criação do mercado financeiro e da *internet* essa atividade se tornou ainda mais ágil: sendo possível, hoje, comprar e vender produtos, ações, commodities e outros ativos em questão de segundos fora do horário comercial convencional. Desta forma, tornou-se possível, também, a criação de estratégias de investimento que envolvem a rápida compra e venda destes ativos, se aproveitando de oportunidades que podem ocorrer durante curtos períodos de tempo no mercado financeiro.

O avanço da tecnologia também possibilitou a criação de novas ferramentas para análise de dados. Em particular, modelos que utilizam redes neurais convolucionais (*CNN - Convolutional Neural Networks*), que geralmente são mais utilizadas para problemas de visão computacional, se mostraram interessantes para uso em séries temporais devido a sua capacidade de extrair *features* dos dados automaticamente, reduzindo a necessidade de criação destas por especialistas (GAMBOA, 2017)[1]. Este tipo de rede neural também já foi utilizada com sucesso para identificar oportunidades de compra e venda no mercado financeiro, superando uma estratégia convencional de *buy and hold* (BARRA et al., 2020)[2].

Além disso, existem novas técnicas para conversão de séries temporais em imagens. Estas técnicas tem o poder de permitir que modelos de visão computacional tradicionais, como as redes neurais convolucionais, possam ser utilizadas para reconhecer, classificar e aprender estruturas e padrões nos dados visualmente. Em particular, a conversão de séries temporais

*guilherme.garmatter@outlook.com
†ara@ufpr.br

em images GAF (*Gramian Angular Fields*) e MTF (*Markov Transition Fields*) se mostrou interessante: alcançando resultados competitivos quando comparados a outros modelos do estado da arte para análise de séries temporais.[3]

Apesar do avanço da tecnologia, das ferramentas e técnicas para predição de séries temporais, a predição do comportamento dos preços no mercado financeiro continua sendo bastante desafiadora. Isto ocorre, pois, o mercado financeiro é afetado por diversos fatores exógenos e de difícil predição como decisões políticas, eventos macroeconômicos mundiais, expectativas dos investidores etc.(ZHANG; WU, 2009)[4]. Este problema pode acarretar em modelos com baixa capacidade preditiva.

Desta forma, exploramos neste artigo a ideia de que pode ser possível aproveitar pequenas vantagens em relação ao mercado utilizando uma estratégia de arbitragem/*trading* de alta frequência, onde o investidor realiza diversas operações de compra e venda durante curtos períodos de tempo. Uma boa analogia para esta estratégia seria a ideia de realizar apostas consecutivas em posse de um dado ou moeda viciada: onde quanto mais vezes se aposta, maior o retorno esperado.

Para isto, aplicamos técnicas de conversão de séries temporais em imagens em conjunto com um *ensemble* de redes neurais convolucionais inspiradas no trabalho de Barra (2020)[2], que obteve relativo sucesso ao prever os movimentos para o índice S&P500, para classificação de momentos de compra e venda no mercado de criptoativos.

2. Conjunto de dados

Para este trabalho, foram utilizados dados de preço e volume negociados em pequenos intervalos de tempo. Estes dados foram obtidos gratuitamente da base de dados da corretora de criptomoedas Binance, que é a maior corretora de criptomoedas do mundo em termos de volume negociado[6]. Os dados foram retirados para o *ticker* BTCUSDT, que corresponde ao ativo Bitcoin precificado na *stablecoin* Tether, para os *timeframes* (intervalos de tempo) de 1, 5, 15, 30 e 60 minutos. O Bitcoin é a maior criptomoeda em termos de capitalização de mercado, enquanto a *stablecoin* Tether possui valor vinculado ao dólar e é o ativo com o maior volume de negociações e liquidez no mercado de criptomoedas[7]. Este *ticker* em específico foi selecionado devido a sua alta liquidez, o que é essencial para que se possa fazer operações de compra e venda em alta frequência sem que os preços se alterem de maneira significativa. A estrutura da base de dados para cada *timeframe* e suas variáveis disponíveis são apresentadas na Tabela 1.

Tabela 1: Descrição das variáveis que compõem a base de dados da Binance

Variável	Descrição
Close time	Data/hora do fim do período.
Open time	Data/hora do início do período.
Open	Preço de abertura.
High	Maior preço para o período.
Low	Menor preço para o período.
Close	Preço de fechamento.
Volume	Volume negociado no período medido em Bitcoin.
Quote asset volume	Volume negociado no período medido em Tether.
Number of trades	Número de operações para o período.
Taker buy base asset volume	Volume de compras a mercado no período medido em Bitcoin.
Taker buy quote asset volume	Volume de compras a mercado no período medido em Tether.

A base de dados corresponde ao período de 2017-08-17 até 2022-12-31 e possui 2.826.480 linhas para o *timeframe* de 1 minuto. Cada linha é transformada em uma janela de dados com informações dos últimos 20 períodos para todos os *timeframes* analisados. Das 11 variáveis disponíveis, utilizamos apenas as referentes ao preço de fechamento e volume negociado medido em Bitcoin. Isto ocorre devido a limitações de memória e de tempo disponível para o treinamento do modelo. Desta maneira as imagens geradas com base nos dados ocuparão um espaço suficientemente pequeno na memória para que o modelo itere sobre elas rapidamente. Vale lembrar que o processo de transformação dos dados em imagens GAF, GASF e MTF aumentará consideravelmente a quantidade de pontos de dados disponíveis para cada período analisado. Ao utilizarmos dados dos últimos 20 períodos de preços de fechamento, por exemplo, teremos uma imagem de tamanho 20x20x3 para cada *timeframe* utilizado como resultado.

2.1. Recursos Computacionais

Para execução deste trabalho foi utilizado primariamente o *software* Anaconda 2023.03-1[8], que inclui ferramentas para programação como o Python[9], Jupyter [10] e outras bibliotecas utilizadas para tratamento dos dados como Pandas[11] e NumPy[12]. Além disso, foram utilizadas as bibliotecas Pyts[14] para conversão dos dados em imagens GAF, GASF e MTF, TensorFlow [15] para modelagem das redes neurais convolucionais e Sklearn[13] para o cálculo de métricas. O ambiente computacional utilizado neste estudo foi um Desktop

Windows 11 com 32GB de RAM, processador Ryzen 5600X e uma GPU RTX 2060 com 6GB de VRAM.

3. Metodologia

3.1. Preparação dos dados

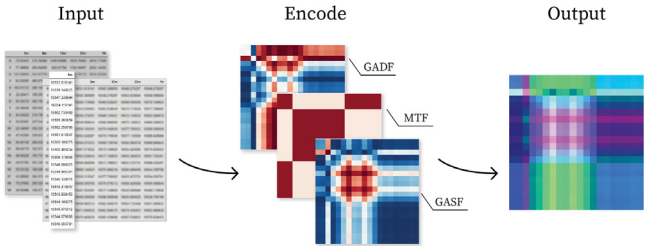

Figura 1: Transformação de uma coluna de dados em imagem. Fonte: Elaborado pelo autor

Em um primeiro momento transformamos as janelas de dados em imagens GAF-GASF-MTF utilizando o método descrito por (WANG; OATES, 2015)[3]. Este método visa utilizar 3 tipos diferentes de codificação de séries temporais em imagens a fim de se obter os benefícios que cada tipo de codificação possui para representar os dados e alimentar o modelo. Assim, cada variável da janela é transformada em um *Gramian Angular Field* (GAF), *Gramian Summation Field* (GASF) e *Markov Transition Field* (MTF). Posteriormente, unimos estas imagens para formar uma única imagem com 3 canais (RGB), sendo cada canal correspondente a um tipo de imagem gerada. Para isto utilizamos as funções GramianAngularField e MarkovTransitionField da biblioteca Pyts[14]. Para criação das imagens MTF utilizamos 3 quantis, pois o uso de uma maior quantidade não era possível em parte dos dados devido ao baixo número de observações e/ou repetições de valores no período analisado.

Após a transformação da série temporal, empilhamos as imagens de preço e volume de mesmo *timeframe* verticalmente e juntamos com os dados de outros *timeframes* horizontalmente. O processo de união das imagens e sua disposição pode ser visualizada por meio da Figura 2. O processo em sua totalidade resulta em imagens de tamanho 40x100 divididas em 3 canais. As imagens foram classificadas em momentos de compra e venda verificando se o preço de fechamento no minuto seguinte foi maior ou menor do que no último minuto. Se o preço foi maior, a imagem foi classificada como compra (long), enquanto se menor, foi classificada como venda (short). Caso o preço tenha permanecido inalterado ou algum dado estivesse

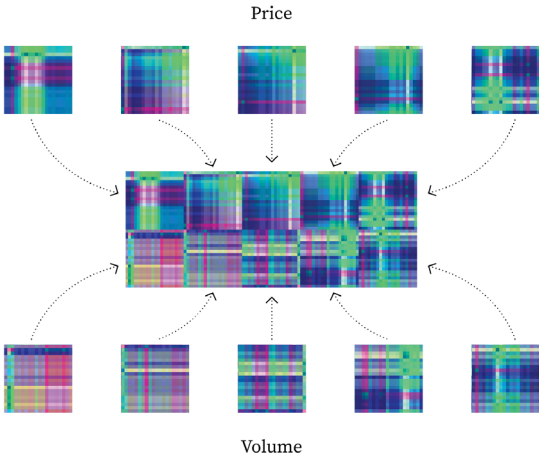

Figura 2: União das imagens. Fonte: Elaborado pelo autor

faltando a imagem foi descartada. Ao final deste processo, haviam 2.692.788 imagens, sendo que 1.348.330 (50,07%) foram classificadas como compra e 1.344.458 (49,93%) como venda.

Também dividimos nossos dados em 80% para treino, 10% para validação e 10% para teste. Os dados de treino e validação correspondem ao período de 08/2017 até 06/2022 minuto a minuto e foram embaralhados. Já os dados de teste correspondem ao período entre 06/2022 até 12/2022 na mesma frequência. Esta separação foi feita para que pudéssemos testar o modelo em um cenário de uso no mundo real: onde treinamos com os dados do passado para tentar prever seu comportamento no futuro.

3.2. *Ensemble* de CNN's

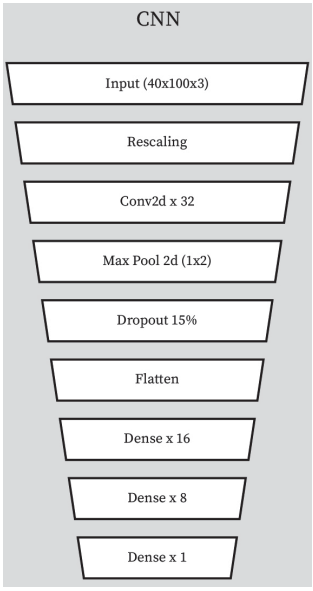

Figura 3: Modelo utilizado. Fonte: Elaborado pelo autor

O modelo convolucional utilizado corresponde a uma rede VGG-16[5] simplificada e utilizando apenas uma convolução antes da camada densa. Testes iniciais mostraram que modelos com menos camadas/convoluções não só convergiam mais rapidamente como evitavam problemas de *overfitting*/sobreajuste e diminuíram o tempo necessário para o treino do modelo consideravelmente. Os dados também foram reescalados para valores entre 0 e 1 através de uma camada de reescalonamento logo após a entrada. O modelo e suas camadas podem ser visualizadas por meio da Figura 3.

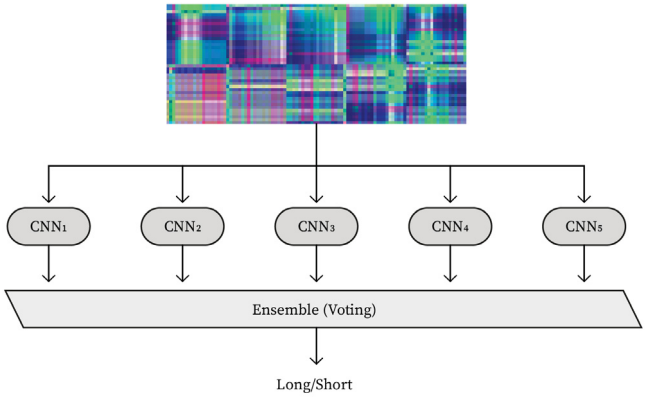

Figura 4: Modelo *ensemble* utilizado. Fonte: Elaborado pelo autor

O modelo utilizado por Barra(2020)[2] originalmente utiliza um *ensemble* com 10 CNN's. Este número de modelos se mostrou inviável para este trabalho devido a maior quantidade de dados disponíveis para o treinamento e limitações de tempo/*hardware*. Desta forma, optamos por utilizar apenas 5 CNN's em nosso modelo.

O processo de treinamento passa 10 vezes pelo conjunto de dados de treino (10 *epochs*) e dura aproximadamente 5 horas. Para este processo foi utilizado o otimizador Adagrad com uma *learning rate* de 0,1 em conjunto com uma função de perda de entropia cruzada binária. Ao final das 10 passagens pelos dados mantivemos apenas o modelo que obteve a melhor acurácia para os dados de validação. Desta forma evitamos um possível sobreajuste/*overfit*: onde o modelo se especializa apenas em prever os dados de treino e perde poder de generalização.

Após o treinamento das 5 CNN's, aplicamos o *ensemble* por meio de uma votação simples: onde a classificação que obtiver a maioria dos votos será a resultante do modelo. Nesta etapa também realizamos um processo de *thresholding*, que definirá qual o voto de cada modelo com base em quão confiante ele está na predição realizada. Desta forma, uma predição só será considerada como um voto para realizar uma operação de compra(*long*) ou venda(*short*) caso a confiança da predição esteja acima do *threshold* definido. Caso contrário, o resultado do modelo é considerado como um voto para não operar(ou *hold*) naquele período.

3.3. Avaliação dos resultados

Para avaliar os resultados do modelo, utilizamos métricas comuns a problemas de classificação binária de imagens como acurácia, precisão, sensibilidade, f1-score e *area under curve* (AUC) para cada *threshold* analisado. Também medimos o percentual de vezes em que o modelo escolheu operar (% cobertura) e lucro obtido com uma estratégia de investimento que utiliza as predições do modelo como base. A estratégia de investimento utilizada pressupõe um capital inicial de 1.000 dólares, onde a compra/venda é realizada pelo preço de fechamento do minuto anterior e a posição comprada ou vendida é desfeita pelo preço de fechamento do minuto seguinte. Não foram avaliados os possíveis custos de transação e/ou efeitos que a execução das operações poderiam ocasionar nos preços.

4. Resultados e discussão

Após rodar o modelo para o conjunto de dados de teste, que corresponde ao período entre 06/2022 até 12/2022 minuto a minuto, obtivemos os resultados mostrados na Tabela 2.

Tabela 2: Resultados para o modelo nos dados de teste

Acurácia	Precisão	Sensibilidade	f1-score	AUC	Cobertura	threshold	Lucro ($)
0.518	0.521	**0.466**	**0.492**	0.518	**1.00**	0.50	**1760.92**
0.524	0.527	0.457	0.489	0.524	0.70	0.51	1461.65
0.529	0.531	0.449	0.487	0.528	0.47	0.52	888.34
0.533	0.536	0.436	0.481	0.532	0.31	0.53	344.29
0.537	0.540	0.417	0.470	0.535	0.21	0.54	192.64
0.541	**0.543**	0.352	0.427	**0.536**	0.13	0.55	52.78
0.545	0.538	0.189	0.279	0.523	0.01	0.60	14.34

Conforme esperado, devido a dificuldade em se prever o comportamento dos preços no mercado financeiro, obtivemos uma acurácia próxima a 50% para todos os *thresholds* utilizados. Esta acurácia foi maior quanto maior o nível de confiança (*threshold*) utilizado para tomada de decisão de compra/venda. O mesmo ocorre para o AUC, mas não para a sensibilidade e f1-score. Isto ocorre pois quanto maior o *threshold* de decisão menor a quantidade de operações realizadas e, consequentemente, menor a sensibilidade e f1-score.

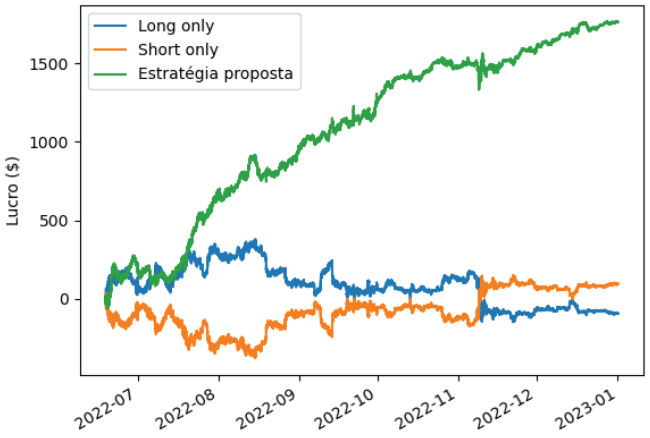

Figura 5: Comparação de estratégias. Fonte: Elaborado pelo autor

O lucro obtido foi maior para o *threshold* de 50%, mesmo com a acurácia sendo a menor neste caso. Isto pode ser explicado pelo maior número de operações realizadas nesta estratégia, que é medida pela porcentagem de cobertura e é a maior para este caso. Isto evidencia que pode ser mais lucrativo utilizar uma estratégia de arbitragem/*trading* de alta frequência quando possuímos pequenas vantagem em relação ao mercado.

Por fim, na Figura 5, comparamos a performance do nosso modelo utilizando o *threshold* de decisão de 0.50 contra a de operar apenas comprado (*long only*), também chamado de *buy and hold*, ou apenas vendido (*short only*) durante o período de testes. Para esta comparação desconsideramos custos de transação e supomos um capital inicial de $1000, que é reinvestido em sua totalidade minuto a minuto com base nas previsões do modelo. Os resultados foram bastante positivos.

5. Comentários finais

Neste trabalho utilizamos técnicas para conversão de séries temporais em imagens e um *ensemble* de redes neurais convolucionais (CNN) para tentar obter uma estratégia de arbitragem/*trading* de alta frequência viável. Apesar da acurácia relativamente baixa obtida pelo modelo, que ficou entre 52% e 54% para os dados de teste, podemos observar que a estratégia foi bastante lucrativa durante o período analisado devido a alta frequência em que foi aplicada. Ressaltamos, também, que a predição do comportamento dos preços no mercado financeiro é bastante desafiadora devido a quantidade de variáveis exógenas ao modelo e que este resultado mostrou uma capacidade preditiva interessante para a aplicação proposta.

Também observamos que o uso de *thresholds* de decisão maiores podem aumentar a acurácia do modelo. Este tipo de filtro para tomada de decisão não resulta necessariamente, no entanto, em modelos mais lucrativos. Isto ocorre devido a menor quantidade de operações realizadas.

Para trabalhos futuros, a combinação de diferentes modelos como LSTM ou classificadores SVM aliados a uma camada de convolução CNN podem se mostrar úteis a fim de se obter melhores resultados a partir das imagens geradas. Também pode ser interessante alterar o tamanho das janelas e/ou utilizar dados de outros ativos e índices do mercado relevantes para geração das imagens.

Agradecimentos

Gostaria de agradecer ao Professor Orientador Anderson Ara por toda ajuda e contribuição nesse artigo. Não posso deixar de agradecer também os meus colegas do grupo de estudos de SVM, pelas inúmeras dicas e por me ajudarem durante todo o processo de desenvolvimento.

Referências

[1] GAMBOA, J. C. B. Deep Learning for Time-Series Analysis. arXiv, , 7 jan. 2017. Disponível em: <http://arxiv.org/abs/1701.01887>. Acesso em: 28 nov. 2022

[2] BARRA, S. et al. Deep learning and time series-to-image encoding for financial forecasting. IEEE/CAA Journal of Automatica Sinica, v. 7, n. 3, p. 683–692, maio 2020.

[3] WANG, Z.; OATES, T. Imaging Time-Series to Improve Classification and Imputation. 1 jun. 2015.

[4] ZHANG, Y.; WU, L. Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network. Expert Systems with Applications, v. 36, n. 5, p. 8849–8854, jul. 2009.

[5] SIMONYAN, K.; ZISSERMAN, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv, , 10 abr. 2015. Disponível em: <http://arxiv.org/abs/1409.1556>. Acesso em: 19 ago. 2023

[6] As Principais Corretoras de Criptomoedas Classificadas Por Volume. Disponível em: <https://coinmarketcap.com/pt-br/rankings/exchanges/>. Acesso em: 17 jun. 2023.

[7] Cryptocurrency Prices, Charts And Market Capitalizations. Disponível em: <https://coinmarketcap.com/>. Acesso em: 17 jun. 2023.
[8] Anaconda 2023.03-1 - Software disponível em: https://docs.continuum.io/free/anaconda/install/
[9] Python 3.10.9 - Software disponível em: https://www.python.org/downloads/release/python-3109/
[10] Jupyter - Software disponível em: https://jupyter.org/install
[11] Pandas - Biblioteca disponível em: https://pandas.pydata.org/docs/getting_started/install.html
[12] NumPy - Biblioteca disponível em: https://numpy.org/install/
[13] Sklearn - Biblioteca disponível em: https://scikit-learn.org/stable/install.html
[14] Pyts - Biblioteca disponível em: https://pypi.org/project/pyts/
[15] Tensorflow - Biblioteca disponível em: https://www.tensorflow.org/install?hl=pt-br