UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

RICARDO ADLEY DA SILVA SENA

ANÁLISE DO USO DE TRANSFORMERS NA PREDIÇÃO DE SÉRIES TEMPORAIS

CAMPINA GRANDE - PB

2024

RICARDO ADLEY DA SILVA SENA

ANÁLISE DO USO DE TRANSFORMERS NA PREDIÇÃO DE SÉRIES TEMPORAIS

Trabalho de Conclusão de Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador : Fábio Jorge Almeida Morais

CAMPINA GRANDE - PB

2024

RICARDO ADLEY DA SILVA SENA

ANÁLISE DO USO DE TRANSFORMERS NA PREDIÇÃO DE SÉRIES TEMPORAIS

Trabalho de Conclusão de Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Fábio Jorge Almeida Morais

Orientador – UASC/CEEI/UFCG

Maxwell Guimarães de Oliveira

Examinador – UASC/CEEI/UFCG

Francisco Vilar Brasileiro

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 15 de Maio de 2024.

CAMPINA GRANDE - PB

RESUMO

A arquitetura baseada em Transformers, desenvolvida para resolver problemas de machine learning relacionados ao processamento de linguagem natural, expandiu-se para outras áreas, como a previsão de séries temporais. Diferentes modelos, baseados em regressão ou árvores de decisão, são utilizados nesse campo, como o ARIMA , XGBoost e Prophet, por exemplo. Cada abordagem possui suas especificidades em termos de precisão e eficiência computacional, o que requer estudos para determinar a melhor a ser adotada em diferentes cenários. Nesse contexto, este trabalho visa apresentar resultados comparativos e analíticos de diferentes abordagens para previsão de séries temporais. Para tal, foi realizado um estudo comparativo analisando o uso da arquitetura baseada em Transformers e seu desempenho frente a modelos tradicionais, utilizando dados referentes aos registros de consumo energético da Universidade Federal de Campina Grande (UFCG). A partir dessa análise, foi possível analisar as abordagens em relação à precisão e eficiência computacional, verificando se a complexidade de uma abordagem mais sofisticada, como a dos Transformers, para previsão de séries temporais, se mostra superior em relação às outras abordagens populares utilizadas para esse fim.

Palavras-Chave: Séries temporais. Transformer. Predição. XGBoost. Consumo Energético.

ANALYSIS OF THE USE OF TRANSFORMERS IN TIME SERIES PREDICTION

ABSTRACT

The Transformer-based architecture, developed to tackle natural language processing machine learning problems, has expanded into other areas such as time series forecasting. Various models, based on regression or decision trees, are employed in this field, such as ARIMA, XGBoost, and Prophet, for instance. Each approach has its own specificities in terms of accuracy and computational efficiency, necessitating studies to determine the best one to adopt in different scenarios. In this context, this work aims to present comparative and analytical results of different approaches for time series forecasting. To do so, a comparative study was conducted analyzing the usage of the Transformer-based architecture and its performance against traditional models, using data related to energy consumption records from the Federal University of Campina Grande (UFCG). Through this analysis, it was possible to assess the approaches in terms of accuracy and computational efficiency, determining whether the complexity of a more sophisticated approach like Transformers for time series forecasting proves superior to other popular approaches utilized for this purpose.

Keywords: Time series. Transformer. Prediction. XGBoost. Energy Consumption.

Análise do Uso de Transformers na Predição de Séries Temporais

Ricardo Adley da Silva Sena
ricardo.sena@ccc.ufcg.edu.br
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil

Fabio Jorge Almeida Morais
fabio@computacao.ufcg.edu.br
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil

Resumo

A arquitetura baseada em Transformers, desenvolvida para resolver problemas de machine learning relacionados ao processamento de linguagem natural, expandiu-se para outras áreas, como a previsão de séries temporais. Diferentes modelos, baseados em regressão ou árvores de decisão, são utilizados nesse campo, como o ARIMA , XGBoost e Prophet, por exemplo. Cada abordagem possui suas especificidades em termos de precisão e eficiência computacional, o que requer estudos para determinar a melhor a ser adotada em diferentes cenários. Nesse contexto, este trabalho visa apresentar resultados comparativos e analíticos de diferentes abordagens para previsão de séries temporais. Para tal, foi realizado um estudo comparativo analisando o uso da arquitetura baseada em Transformers e seu desempenho frente a modelos tradicionais, utilizando dados referentes aos registros de consumo energético da Universidade Federal de Campina Grande (UFCG). A partir dessa análise, foi possível analisar as abordagens em relação à precisão e eficiência computacional, verificando se a complexidade de uma abordagem mais sofisticada, como a dos Transformers, para previsão de séries temporais, se mostra superior em relação às outras abordagens populares utilizadas para esse fim.

Keywords

Séries temporais. Transformer. Predição.XGBoost. Consumo Energético.

1 INTRODUÇÃO

O planejamento do consumo energético é essencial para um uso eficiente e, para tal, são utilizados dispositivos que medem o consumo periodicamente e armazenam esses dados para análise posterior. Os dados coletados pelos dispositivos são organizados em um formato de séries temporais, onde para cada intervalo de tempo o valor de uma determinada métrica é salvo [9]. Em muitos contextos, a estimativa de valores futuros dessas métricas é necessária para planejamento e tomadas de decisão. Para isso, são utilizadas abordagens de previsão de séries temporais, do inglês *Time Series forecasting (TSF)*, que historicamente é uma tarefa que possui diferentes aplicações, como estimação de tráfego, gestão de energia e investimentos financeiros [21]. Essas aplicações podem buscar prever resultados para diferentes intervalos de tempo, seja de apenas um dia até anos à frente, mas independentemente do período necessário, a previsão é uma ajuda importante e eficaz para um planejamento eficaz e eficiente [9].

Para a realização dessas tarefas de *TSF*, diferentes ferramentas são utilizadas, modelos estatísticos como o ARIMA[3], algoritmos de Redes Neurais e de redes temporais convolucionais [21]. Essas estratégias são vistas como bem sucedidas na predição de séries temporais. No entanto, existem ainda outras arquiteturas, criadas para contextos de predição em outras formas de registro, como o Transformer [20]. Essa solução foi proposta pela equipe de pesquisa do Google buscando melhorar o desempenho de tarefas envolvendo processamento de linguagem natural e atualmente utilizado em diferentes aplicações bem sucedidas para esse fim, como o modelo BERT [5].

Apesar de criadas inicialmente para o uso em dados textuais, com o passar dos anos, inúmeras arquiteturas usando Transformer foram propostas para avançar a performance e atingir o estado da arte em diferentes tarefas [16]. Com o campo da predição de séries temporais não foi diferente, foram geradas arquiteturas com esse objetivo, e existem melhorias feitas em baixo nível (módulo) e alto nível (arquitetura) de Transformers que buscam otimizar a performance em modelagem de séries temporais [16].

Desta forma, este trabalho pretende analisar o uso de Transformers na predição de séries temporais, utilizando os dados de consumo energético da Universidade Federal de Campina Grande (UFCG)[1], observando métricas de resultados e recursos computacionais, buscando indicar a melhor estratégia de predição para o contexto analisado. Para esse fim, foram considerados dados de consumo de diferentes edificações da UFCG em um histórico de até um ano.

Nas próximas seções, este trabalho apresentará o referencial teórico adotado (Seção 2), a metodologia empregada, incluindo métricas e modelos discutidos (Seção 3), a análise dos dados utilizados no estudo, e suas características (Seção 4), a análise e discussão dos resultados obtidos por meio dos experimentos realizados (Seção 5), seguido das conclusões (Seção 6). Por fim, as seções de agradecimentos (Seção 7) e as referências utilizadas. Todos os dados e implementações utilizados neste estudo estão publicamente disponíveis no *GitHub*[2].

2 REFERENCIAL TEÓRICO

A predição de séries temporais aplica diferentes modelos e possui um longo tempo de estudo. Abordagens estatísticas como o ARIMA, suavização exponencial e modelos estruturais para previsão de séries temporais têm sido utilizadas desde a década de 1970. Em geral, os modelos paramétricos utilizados em métodos estatísticos exigem um conhecimento significativo do domínio para serem construídos [21].

Dessa forma, modelos com maior poder computacional e maior facilidade de uso foram desenvolvidos. Nos últimos anos, diversos Transformers têm sido propostos para avançar significativamente

Trabalho de Conclusão de Curso, Bacharelado em Ciência da Computação, 2024
.

[1] UFCG: https://portal.ufcg.edu.br/
[2] GitHub: https://github.com/ricardoadley/transformer-serie-temporal.git

o estado da arte em diversas tarefas [16]. Isso se dá, pois o mecanismo de atenção dos Transformers os torna uma arquitetura diferenciada em relação aos demais modelos. No caso de dados sequenciais, um mecanismo de atenção *multi-head* pode atender conjuntamente a informações em diferentes posições na sequência, tornando os Transformers uma arquitetura atraente para previsão de séries temporais [1]. A organização de abordagens baseadas em Transformers pode ser vista na Figura 1, onde podemos observar o pipeline da operação que segue desde a entrada dos dados para o pré-processamento, seguido da geração de *embeddings* para os dados e, por fim, o *encoder* e *decoder*, que são definidos conforme as especificações do modelo Transformer adotado.

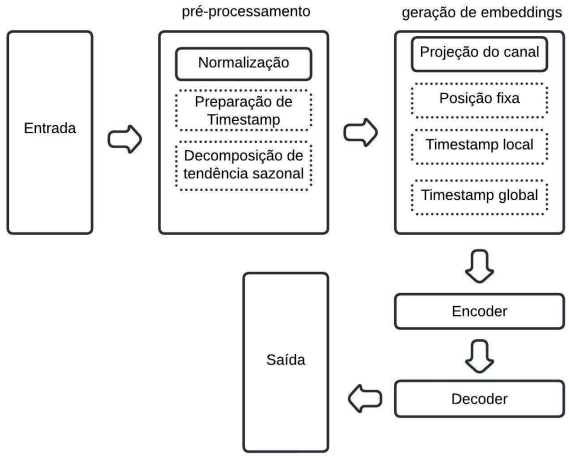

Figura 1: *Pipeline* de execução para uma solução baseada em Transfomer. No pré-processamento e na geração de *embeddings* as operações circuladas por linhas são essenciais e as por pontilhado são opcionais [21].

Mecanismos de atenção são utilizados em tradução, classificação de imagens ou aprendizado tabular para identificar porções importantes da entrada para cada instância usando a magnitude dos pesos de atenção. Recentemente, eles foram adaptados para séries temporais com motivações de interpretabilidade, utilizando arquiteturas baseadas em LSTM (*Long short-term memory*) [8]. Desta forma, foram propostas muitas variantes do Transformer para lidar com desafios específicos na modelagem de séries temporais e foram aplicadas com sucesso a várias tarefas nesse contexto [16]. Algumas dessas variantes são exploradas nesta pesquisa, sendo elas, o Informer [22], que possui uma aplicação probabilística multivariada, e o Transformer em uma abordagem *vanilla*.

Apesar de diversas variações em sua arquitetura, ainda existem desafios enfrentados pelas aplicações Transformers. Embora seja uma rede geral e universal para modelar dependências de longo alcance, isso também tem um custo, ou seja, muitos dados são necessários para treinar o Transformer a fim de melhorar a generalização e evitar o *overfitting*, ou seja, o ajuste excessivo, dos dados [16].

Dessa forma, os conceitos aqui apresentados contribuem para o objetivo desta pesquisa, onde se deseja avaliar o desempenho do uso de transformers na predição de séries temporais, comparado as abordagens mais tradicionalmente utilizadas e desenvolvidas para esse fim.

3 METODOLOGIA

A pesquisa foi realizada de forma exploratória e qualitativa, utilizando como fonte de dados para os modelos estudados os registros de consumo de energia da Universidade Federal de Campina Grande (UFCG) obtidos através da API de controle do LiteMe[3], entidade que realiza monitoramento do consumo da instituição.

Para os testes os modelos selecionados foram o modelo autorregressivo integrado de médias móveis (ARIMA), Prophet [19], XGBoost [4] e transformers de código aberto para uso em séries temporais, sendo eles o Informer e um transformer *vanilla* com implementação simples de um codificador e decodificador.

Os dados obtidos foram analisados e tratados para uso nos modelos de predição alvo, que serão comparados utilizando os valores obtidos nas métricas de RMSE (Raiz do Erro Quadrático Médio), R^2 score (Coeficiente de determinação), sMAPE (Erro percentual médio simétrico absoluto), além de observado o uso de memória RAM buscando um comparativo de eficiência computacional entre as abordagens.

3.1 Materiais e Métodos

Para treinamento e avaliação dos modelos usando o ambiente de execução jupyter notebook [12] por meio do Google Colaboratory[4] com as configurações de 13 GB de memória RAM, GPU Tesla T4 com 15 GB de memória RAM, processador Intel(R) Xeon(R) CPU 2.00GHz para execução dos modelos.

Durante a execução dos modelos, análise e tratamento dos dados foram utilizados a linguagem de programação python 3 [13], as bibliotecas pytorch [15], pandas [11], seaborn [17] e transformers além da biblioteca darts [7] para aplicação dos modelos ARIMA e Prophet.

3.2 Métricas para análise comparativa

A fim de avaliar o desempenho dos diferentes modelos na tarefa de predição dos dados foram utilizadas as métricas comumente conhecidas para tal fim, sendo elas o RMSE, R^2 e sMAPE. Considerando o formato de séries de dados, as métricas são calculadas a partir das seguintes definições:

- n representa o número total de observações no conjunto de dados.
- y_i é o valor observado na i-ésima observação.
- $\hat{y}_i$ é o valor previsto pelo modelo na i-ésima observação.
- $\bar{y}$ é a média dos valores observados.

3.2.1 RMSE

O RMSE nos dá uma ideia da magnitude dos erros da previsão, tendo o seu valor ideal como 0, ou seja, quanto mais próximo o valor do RMSE estiver de 0 melhor o resultado obtido pelo modelo. O seu cálculo é realizado conforme a fórmula abaixo:

[3]LiteMe: https://ufcg.liteme.com.br/
[4]Google Colaboratory: https://colab.research.google.com/

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

3.2.2 R^2 (R-squared)

Medida estatística que indica o ajuste do modelo aos dados. Um valor próximo a 1 indica um ajuste perfeito. A métrica é calculada da seguinte forma:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}i)^2}{\sum i = 1^n(y_i - \bar{y})^2}$$

3.2.3 sMAPE

Métrica responsável pelo cálculo da média das porcentagens absolutas dos erros entre os valores de Predição e os valores reais. É uma métrica útil em casos onde valores zero estão presentes nos dados. O valor ideal da métrica é o mais próximo de 0 possível, e seu cálculo é realizado usando a fórmula:

$$sMAPE = \frac{100}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2}$$

3.2.4 Uso de memória RAM

A métrica de uso da memória RAM será calculada usando a biblioteca psutil da linguagem Python 3. Os valores serão armazenados durante a execução e o maior valor registrado será definido como o valor da métrica.

3.3 Modelos utilizados no estudo

Para a execução deste estudo, foram selecionados diferentes modelos com foco em previsão em séries temporais: os Transformers Informer e Vanilla, objeto do estudo, e os modelos ARIMA, XGBoost e Prophet, referências na área de TSF e utilizados no estudo como base comparativa dos resultados.

3.3.1 ARIMA

O modelo é definido por uma combinação de autorregressão (AR), diferenciação (I) e médias móveis (MA). Expresso pela fórmula ARIMA(p,d,q) conforme a seguir.

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \ldots + \phi_p Y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

Onde, os valores p, d e q , são definidos da seguinte forma:

- P é o número de termos autorregressivos (AR) no modelo.
- D é o número de diferenciações necessárias para tornar a série temporal estacionária.
- Q é o número de termos de média móvel (MA) no modelo.

Nesse estudo, o modelo ARIMA, utilizado por meio da biblioteca darts, foi definido como ARIMA(0,1,0) após análise da autocorrelação e autocorrelação parcial dos dados.

3.3.2 XGBoost

Nesse estudo, o modelo XGBoost foi implementado utilizando a biblioteca disponível na *Distributed (Deep) Machine Learning Community* (DMLC) [6]. Durante a implementação, foram extraídas as características dia da semana, dia do mês, trimestre, mês, semana do ano, ano e decorrido da série temporal, buscando melhorar os resultados no treinamento do modelo. Por fim, o modelo foi definido com 100 estimadores e rodadas de parada antecipada de 50 épocas.

3.3.3 Prophet

A implementação do modelo seguiu os padrões da biblioteca darts, garantindo a implementação sem problemas. O parâmetro *future-conv* foi configurado com os valores *cyclic* definidos como *True*, permitindo que o modelo capture padrões cíclicos nos dados, quando apropriado. Além disso, o parâmetro *addlength* foi especificado como o tamanho dos dados de teste, assegurando uma consideração adequada da extensão temporal dos dados para a realização da previsão.

3.3.4 Transformer Vanilla

O modelo foi definido com apenas uma dimensão de entrada, tamanho dos *embeddings* de 64, 4 cabeçalhos de atenção, 2 camadas de codificação e probabilidade de desistência de 0,2. Para o treinamento, o modelo contou com o uso do otimizador Adam [14] com *learning rate* de 0.001 e ReduceLROnPlateau [10] para redução do *learning rate* caso necessário, com paciência de 3 épocas e fator de 0,5. O treinamento ainda contou com *early stopping* observando o valor do *loss* de validação com paciência de 5 épocas.

3.3.5 Informer

O modelo foi definido utilizando o exemplo de uso presente no HuggingFace [18]. A arquitetura selecionada faz uso de atenção esparsa probabilística definida em termos gerais de modo a permitir uma distribuição eficaz de recursos computacionais, concentrando o processamento em áreas específicas de interesse conforme determinado pela probabilidade de relevância, promovendo assim uma melhor adaptação do modelo aos dados e uma execução mais eficiente das tarefas propostas. O treinamento foi realizado por 15 épocas, contando com o otimizador Adam, com taxa de aprendizado de 6×10^{-4} e decaimento de peso de 1×10^{-1}.

3.4 Separação dos dados entre treino e teste

Para os modelos ARIMA, XGBoost, Prophet e Transformer Vanilla, o treinamento ocorreu com os dados até 2023-07-31, e o teste com os dados de 2023-08-01 a 2023-08-17 para os conjuntos de dados do CEEI, CN, SPLAB e CTLABS. Usando os dados do LSD, o treinamento foi realizado de 2022-09-08 a 2023-07-19 e o teste foi realizado a partir de 2023-07-20 até 2023-08-07. Para o Informer, foi seguida a divisão efetuada pelo código da implementação em treino, teste e validação.

4 COLETA E TRATAMENTO DE DADOS

Os *datasets* utilizados no estudo correspondem a 5 edificações distintas da universidade, cada um apresenta sua especificação em relação ao consumo de energia com o *dataset* referente ao bloco do laboratório de sistemas distribuídos (LSD) correspondendo ao maior número de registros como pode ser observado na Tabela 1. A quantidade de registros corresponde aos valores tratados como descrito na subseção 4.1.

Tabela 1: Quantidade de Registros por Base de Dados

Nome da Base de Dados	Quantidade de Registros
LSD	8016
SPLAB	5352
CN	5352
CEEI	5352
CTLABS	5352

Cada registro equivale ao intervalo de 1 hora e foram coletados entre os dias 08/09/2022 e 17/08/2023. A base de dados foi montada seguindo a descrição disponível na Tabela 2.

Tabela 2: Campos do *Dataset* Utilizado no Estudo

Coluna	Descrição
`date`	data e hora do registro
`name`	nome do bloco referente a origem do registro
`totalw`	consumo de watts no momento do registro

4.1 Ausências de registros

Durante a análise dos dados foi constatado que em alguns momentos houve falha na coleta dos registros de consumo dos blocos, sendo o maior período em que essa falha ocorreu de forma contínua correspondente a 30 dias e o menor período a 5 horas. Dessa forma, foi necessário a adoção de tratamento para os dados faltantes do *dataset* a fim de completar a série temporal.

Para o preenchimento dos dados, foi utilizada a estratégia de preenchimento para frente, do inglês *forward fill*, onde o último dado registrado é repetido até que aconteça um novo registro no banco. Dessa forma, o consumo energético é preservado como o último em registro pelo sistema.

4.2 Características das séries temporais usadas no estudo

Observando como o consumo de watts por cada bloco, podemos notar se a série temporal apresenta sazonalidade (padrão) e também se existem *outliers* de grande destaque nos registros, que podem acabar por causar problemas durante o treinamento dos modelos. Como podemos notar na Figura 2, na série temporal do CEEI, durante o período de estudo, observamos picos acentuados e uma variabilidade considerável nos valores. Especificamente entre fevereiro de 2023 e julho de 2023, identificamos picos de consumo intercalados com períodos de baixa atividade, padrão que pode ser notado nas bases de dados do CN, SPLAB e CTLABS. Apenas nos dados do LSD não observamos semelhanças entre os registros com outras bases.

Podemos observar ainda que os dados do LSD apresentam uma sazonalidade durante todo o período de registro, com a ocorrência de apenas dois *outliers* de valores baixos que ocorreram entre maio e julho de 2023. Ocorrências de *outliers* como essa não são observadas nos registros presentes nas demais bases utilizadas no estudo.

Observando a Figura 2, é notável ainda que nos registros referentes ao LSD é onde se observa os maiores valores de consumo, junto das maiores oscilações com valores próximos a 0 até maiores de 40000 watts. Nos demais registros, apenas o CN se aproxima desses valores, com registros máximos próximos a 30000 watts.

Por fim, as bases do CEEI, CN, SPLAB e CTLABS, embora tenham um range de valores diferente, compartilham um padrão de registro muito semelhante e apresentam o mesmo intervalo de registro, entre janeiro e agosto de 2023, enquanto a base de dados do LSD se destaca por suas características mais diferenciadas e um tempo de registro maior, ocorrendo entre setembro de 2023 e agosto de 2023.

4.3 Tratamento de outliers

Para realizar a ação de tratamento dos *outliers* dos registros e, dessa forma, deixá-los em uma qualidade melhor para o uso no treinamento dos modelos, podemos utilizar um método explícito de remoção, sendo ele o realizado por meio do uso dos quartis. Usando uma regra de alcance interquartil, onde os valores de corte são:

$$limiteSuperior = Q1 - 1.5 \times (Q3 - Q1)$$
$$limiteInferior = Q3 + 1.5 \times (Q3 - Q1)$$

Onde $Q1$ e $Q3$ correspondem ao primeiro e terceiro quartis, respectivamente[2].

O tratamento consistiu na definição de *outliers* por meio do método explícito e na substituição desses valores, seguindo a regra de que, se o valor identificado estiver dentro dos limites, permanece inalterado. Caso esteja abaixo do limite inferior, é substituído pelo próprio limite inferior; caso esteja acima do limite superior, é substituído pelo próprio limite superior. Essa abordagem foi empregada com o intuito de minimizar o impacto na qualidade dos dados ao remover os *outliers*.

5 ANÁLISES E RESULTADOS

Nesta seção, serão discutidos os resultados dos experimentos executados com cada uma das 5 bases de dados, observando o desempenho dos modelos conforme as métricas definidas. É importante destacar que, em relação ao tempo de treinamento, apenas o modelo Informer apresentou um tempo longo significativo (cerca de 2 horas para cada base de dados) e necessitou de processamento em GPU, com as características descritas na seção 3.1.

Na subseção 5.2, será discutido de forma geral o desempenho dos modelos em relação ao valor máximo de memória RAM utilizada durante o treinamento com as diferentes bases de dados.

5.1 Métricas de Desempenho

Observando os resultados obtidos nas diferentes bases de dados, disponíveis na Tabela 3, é evidente que o Transformer *Vanilla* se destaca como o modelo com melhor desempenho. Esse modelo

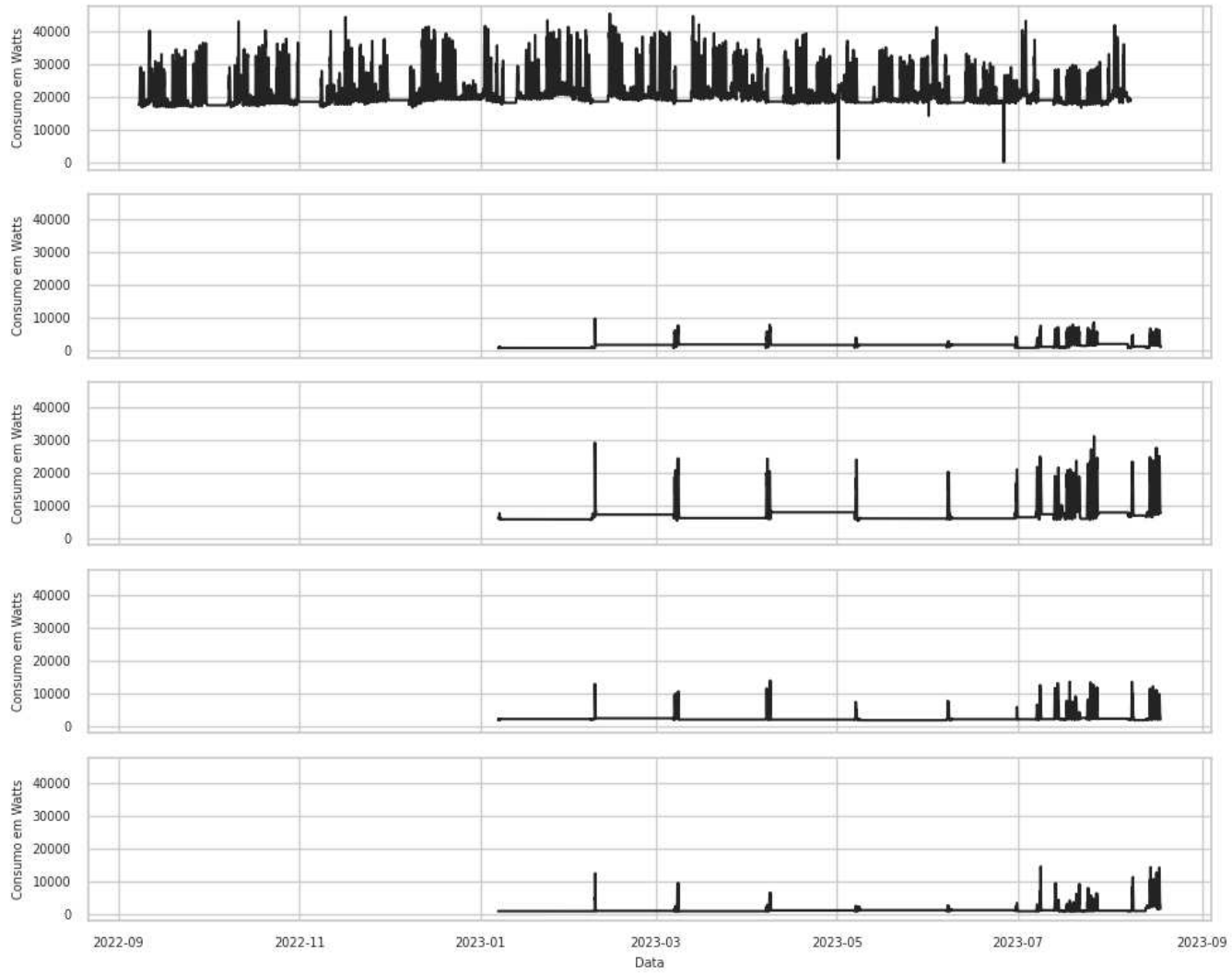

Figura 2: Registro de Consumo em Watts em Intervalos de Uma Hora pelos Edifícios Utilizados no Estudo.

supera tanto o Prophet quanto o ARIMA na maioria das métricas analisadas, especialmente em termos de R^2 e RMSE. Em contraste, o Informer, embora apresente resultados competitivos em algumas métricas, consome consideravelmente mais memória RAM do que os outros modelos.

Pode-se notar essa qualidade de resultado apresentada pelo Transformer *Vanilla* nos testes realizados na base de dados do CEEI, onde ele continuou a demonstrar resultados superiores aos demais modelos e alcançou seu melhor desempenho em todos os testes. Em contraste, os demais modelos, com exceção do Informer, obtiveram resultados piores em termos de R^2 quando comparados aos resultados nos dados do LSD, indicando uma menor adequação aos dados. Destaca-se, em particular, o desempenho do ARIMA, que obteve um valor de R^2 negativo, o que sugere que o modelo não convergiu e, consequentemente, não conseguiu gerar uma predição válida.

Em relação aos testes realizados na base de dados CN, pode-se observar o melhor desempenho do Informer, apresentando uma boa competição com os resultados do Transformer *Vanilla* em termos de sMAPE, mas não se apresentando melhor nas demais métricas apesar de superar os demais modelos analisados nessa base. Quando observamos os resultados de R^2, na Tabela 5.2, apenas as abordagens baseadas em Transformers (*Vanilla* e Informer) não apresentaram resultados negativos.

Os resultados na base de dados SPLAB e CTLABS seguiram uma tendência semelhante às anteriores, com o Transformer *Vanilla* mostrando-se novamente como a melhor opção em termos de R^2 e RMSE. O Informer, por sua vez, teve um desempenho comparativamente bom em termos de sMAPE, mas ainda foi superado pelo Transformer *Vanilla* e continuou consumindo mais recursos de memória que os demais modelos analisados.

Tabela 3: Métricas Obtidas para Cada Modelo nas Bases de Dados Utilizadas no Estudo.

Base de Dados	Modelo	R^2	RMSE	sMAPE	RAM
LSD	ARIMA	0.34	2942.19	10.91	1.17GB
	XGBoost	0.16	2202.66	11.96	1.17GB
	Prophet	0.41	2763.55	9.59	1.12GB
	Informer	-0.22	5146.68	0.17	9.54GB
	Transformer Vanilla	0.82	1486.47	3.49	1.66GB
CEEI	ARIMA	-0.49	638.18	31.27	1.13GB
	XGBoost	0.02	515.21	30.38	0.65GB
	Prophet	0.0	522.99	30.07	1.17GB
	Informer	0.0	21.85	0.01	7.20GB
	Transformer Vanilla	0.91	158.29	6.05	1.47GB
CN	ARIMA	-0.06	834.50	8.89	1.18GB
	XGBoost	-0.01	815.68	8.38	0.54GB
	Prophet	-1.07	1167.93	12.41	1.12GB
	Informer	0.0	787.06	0.12	7.07GB
	Transformer Vanilla	0.82	342.51	2.04	1.45GB
SPLAB	ARIMA	-0.13	276.90	10.14	1.16GB
	XGBoost	-0.01	261.73	10.14	0.64GB
	Prophet	-1.21	386.35	14.16	1.13GB
	Informer	0.0	135.84	0.07	7.20GB
	Transformer Vanilla	0.90	80.96	2.39	1.44GB
CTLABS	ARIMA	-0.16	341.87	21.87	1.11GB
	XGBoost	0.02	312.59	21.31	0.61GB
	Prophet	-0.14	338.82	23.33	1.17GB
	Informer	0.0	86.59	0.08	7.20GB
	Transformer Vanilla	0.96	60.76	2.43	1.45GB

O Transformer *Vanilla* demonstrou ser a escolha mais consistente e eficaz em todas as bases de dados analisadas, como se pode ver na Tabela 3, proporcionando resultados superiores em termos de R^2 e RMSE, além de exigir menos recursos computacionais em comparação com o Informer, único modelo a apresentar resultados competitivos com o *Vanilla* ao observarmos sMAPE e RMSE.

5.2 Consumo de Memória RAM

Como observado nas análises de resultados nas bases de dados do estudo, o modelo Prophet apresentou um dos menores valores de memória RAM utilizada durante o treinamento entre os modelos analisados e, com base nos resultados dispostos na Tabela , para momentos em que a memória RAM é crucial para a execução, ele se torna uma boa opção para os dados do LSD. No entanto, ao considerarmos um contexto geral de resultados, o Transformer *Vanilla* se mostra como a melhor solução custo-benefício entre as analisadas, consumindo menos de 2GB de memória RAM em todos os casos analisados e apresentando bons resultados em todos, ao contrário do Prophet, que apresenta R^2 negativo em 3 dos 5 casos analisados.

Por outro lado, o modelo Informer não se mostra uma boa opção quando observamos essa métrica. Além do maior tempo de treinamento e da necessidade do uso de GPU para comportar a quantidade de RAM utilizada, que chega a ser 9 vezes maior do que a dos demais modelos, o modelo não apresentou resultados gerais tão bons quanto o Transformer *Vanilla*. Em uma situação com mais dados e, consequentemente, uma solução que necessita de maior complexidade, o modelo Informer pode se apresentar como uma boa opção, desde que se disponha dos recursos computacionais necessários para sua execução.

6 CONCLUSÕES

Neste trabalho, foi realizado um estudo sobre o uso de modelos baseados em transformers para a predição de séries temporais. Comparamos o desempenho em termos de adequação aos dados, desempenho da predição e uso de memória RAM com os modelos tradicionais ARIMA, XGBoost e Prophet. Foram utilizados dados sobre o consumo energético de 5 edifícios da UFCG, disponibilizados pelo LiteMe e processados antes do uso no treinamento dos modelos.

Com os resultados e análises aqui apresentados, podemos observar os benefícios e desvantagens do uso de transformers em previsões de séries temporais. Utilizando-se das métricas definidas e dos tratamentos de dados utilizados, foi possível comparar de forma equitativa os diferentes modelos disponíveis para previsão de série temporal e, com isso, foi observado que os transformers, apesar de utilizados em sua arquitetura mais simples, se apresentam como uma excelente opção para a tarefa, enquanto em suas

arquiteturas mais complexas podem encontrar impedimentos na disponibilidade de recursos computacionais e quantidade de dados.

Os avanços na área, principalmente em busca da economia de recursos, podem tornar outras arquiteturas baseadas em Transformers, principalmente os pré-treinados, uma opção emergente para sanar os desafios aqui apresentados para o uso de recursos, tornando-se um bom direcionamento para os próximos estudos na área. A utilização em bases de dados maiores também é interessante para observar se os resultados obtidos pelos demais modelos se repetem.

Por fim, o objetivo do presente trabalho de levantar boas opções para o contexto analisado, não só em relação ao desempenho na tarefa, mas também observando o seu custo computacional, foi atingido. Como levantado, a arquitetura baseada em Transformer simples aqui denotada como Transformer *Vanilla* se apresentou como a melhor opção custo-benefício para o contexto analisado.

7 Agradecimentos

Gostaria de agradecer primeiramente à minha mãe, Angelina Ângela, que sempre me incentivou nos estudos e me apoiou em minha jornada. Sem ela ao meu lado, eu nunca teria chegado tão longe, e à minha tia Jeane Tavares por tornar minha moradia em Campina Grande possível e, consequentemente, minha graduação. Agradeço ainda aos meus avós e todos os meus familiares pelo constante apoio durante toda a minha jornada.

Agradeço também aos meus amigos de graduação, Iele Passos, Antonio Neto, Natália André, Ana Truta, Ekarani Teles e Brenda Alves. Nas constantes ondas da vida, vocês foram sempre meus portos seguros e minhas melhores companhias, independentemente da dificuldade enfrentada.

Por fim, agradeço ao meu excelente orientador, Fábio Morais, por sua paciência e orientações que possibilitaram a realização deste trabalho de conclusão de curso. Agradeço também a todos os colaboradores do Laboratório de Sistemas Distribuídos, minha primeira casa na graduação, em especial à Cleide, por ser sempre uma energia positiva em nossas vidas.

Referências

[1] Ahmadi, A., Daccache, A., Sadegh, M., and Snyder, R. L. Statistical and deep learning models for reference evapotranspiration time series forecasting: A comparison of accuracy, complexity, and data efficiency. *Computers and Electronics in Agriculture 215* (2023), 108424.

[2] Bali, T. G., Engle, R. F., and Murray, S. *Empirical Asset Pricing: The Cross Section of Stock Returns.* John Wiley & Sons, 2016.

[3] Box, G., and Jenkins, G. *Time Series Analysis: Forecasting and Control.* Holden-Day series in time series analysis and digital processing. Holden-Day, 1976.

[4] Chen, T., and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Aug. 2016), KDD '16, ACM, pp. 785–794.

[5] Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Distributed (Deep) Machine Learning Community. Xgboost documentation. https://xgboost.readthedocs.io/, 2024. Acessado em 14 de março de 2024.

[7] Herzen, J., Lässig, F., Piazzetta, S. G., Neuer, T., Tafti, L., Raille, G., Van Pottelbergh, T., Pasieka, M., Skrodzki, A., Huguenin, N., et al. Darts: User-friendly modern machine learning for time series. *Journal of Machine Learning Research 23*, 124 (2022), 1–6.

[8] Hochreiter, S., and Schmidhuber, J. Long Short-Term Memory. *Neural Computation 9*, 8 (11 1997), 1735–1780.

[9] Hyndman, R. J., and Athanasopoulos, G. *Forecasting: Principles and Practice.* OTexts, Melbourne, 2021.

[10] Keras. Reducelronplateau callback. https://keras.io/api/callbacks/reduce_lr_on_plateau/, 2024.

[11] Pandas Development Team. Pandas documentation. https://pandas.pydata.org/pandas-docs/stable/, 2024. Acessado em 14 de março de 2024.

[12] Project Jupyter. Jupyter documentation. https://jupyter-notebook.readthedocs.io/en/latest/, 2024. Acessado em 14 de março de 2024.

[13] Python Software Foundation. Python 3 documentation. https://docs.python.org/3/, 2024. Acessado em 14 de março de 2024.

[14] PyTorch. Adam optimizer. https://pytorch.org/docs/stable/generated/torch.optim.Adam.html, 2024.

[15] PyTorch Contributors. Pytorch documentation. https://pytorch.org/docs/, 2024. Acessado em 14 de março de 2024.

[16] Qingsong, W., Tian, Z., Chaoli, Z., Weiqi, C., Ziqing, M., Junchi, Y., and Liang, S. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125* (2022).

[17] Seaborn Development Team. Seaborn documentation. https://seaborn.pydata.org/, 2024. Acessado em 14 de março de 2024.

[18] Simhayev, E., Rogge, N., and Rasul, K. Informer: Beyond efficient transformer for time series forecasting, 2024.

[19] Taylor, S. J., and Letham, B. Forecasting at scale. *PeerJ Preprints 5* (2017), e3190v2.

[20] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems* (2017), vol. 30.

[21] Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? In *AAAI conference on artificial intelligence* (2022), vol. 37, pp. 11121–11128.

[22] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting, 2021.