INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DE SANTA CATARINA

CAMPUS SÃO JOSÉ

GUSTAVO PAULO

APLICAÇÃO DE MODELOS FUNDACIONAIS DE SÉRIES TEMPORAIS PARA A PREVISÃO EM DIFERENTES DOMÍNIOS

SÃO JOSÉ

2025

Gustavo Paulo

Aplicação de modelos fundacionais de séries temporais para a previsão em diferentes domínios

Monografia apresentada ao Curso de Engenharia de Telecomunicações do campus São José do Instituto Federal de Santa Catarina para a obtenção do diploma de Engenheiro de Telecomunicações.

Orientador: Prof. Arliones Stevert Hoeller Junior, Dr.

Coorientador: Prof. Mario de Noronha Neto, Dr.

São José

2025

Gustavo Paulo

Aplicação de modelos fundacionais de séries temporais para a previsão em diferentes domínios

Monografia apresentada ao Instituto Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Telecomunicações.

São José, 8 de dezembro de 2025.

Documento assinado digitalmente
gov.br
ARLIONES STEVERT HOELLER JUNIOR
Data: 12/12/2025 15:54:21-0300
Verifique em https://validar.iti.gov.br

Prof. Arliones Stevert Hoeller Junior, Dr.

Orientador – Instituto Federal de Santa Catarina

Documento assinado digitalmente
gov.br
MARIO DE NORONHA NETO
Data: 12/12/2025 17:41:01-0300
Verifique em https://validar.iti.gov.br

Prof. Mario de Noronha Neto, Dr.

Coorientador – Instituto Federal de Santa Catarina

Documento assinado digitalmente
CLEBER JORGE AMARAL
Data: 17/12/2025 16:07:08-0300
CPF: ***.848.639-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Cleber Jorge Amaral, Dr.

Instituto Federal de Santa Catarina

Documento assinado digitalmente
gov.br
ODILSON TADEU VALLE
Data: 13/12/2025 06:45:01-0300
Verifique em https://validar.iti.gov.br

Prof. Odilson Tadeu Valle, Dr.

Instituto Federal de Santa Catarina

Aos meus pais, ao meu irmão e à minha namorada.

AGRADECIMENTOS

Primeiramente agradeço a Deus, pelas bênçãos que sustentaram esta jornada.

Agradeço com muito amor ao meu pai, Samuel Sérgio Paulo, e à minha mãe, Ester Francisca Ferreira Paulo, pelo apoio incondicional, pelos conselhos e pela base que me deram em todos os aspectos da vida. Agradeço também ao meu irmão, Guilherme Paulo, pela amizade e parceria ao longo de todos os anos. E à minha namorada, Luiza Kuze, pela paciência, pelo incentivo diário, e por estar ao meu lado em cada etapa dessa caminhada, acreditando em mim até mesmo nos dias em que eu não acreditava.

Sou profundamente grato ao meu orientador Arliones Stevert Hoeller Junior e ao meu coorientador Mario de Noronha Neto, por toda a parceria e dedicação não apenas neste Trabalho de Conclusão de Curso, mas também pela confiança que depositaram em mim ao longo do projeto de pesquisa que realizamos juntos. Essa oportunidade foi um divisor de águas na minha trajetória profissional e pessoal, pois foi ali que encontrei meu propósito e meu caminho no campo da Inteligência Artificial.

Também sou grato aos professores, colegas e amigos do IFSC que, de diferentes formas, contribuíram para minha formação ao longo desses anos.

“Inherited Will, the Flow of Time, and the Dreams of the people... These are things that cannot be stopped” (Gol D. Roger)

RESUMO

Diante da evolução dos Time Series Foundation Models (TSFM) e de sua capacidade de capturar dependências temporais complexas sem ajustes específicos, esses modelos vêm transformando pipelines de previsão em diversos domínios. A predição *zero-shot* dos TSFM elimina a etapa de coleta massiva de dados rotulados para treinamento, o que é especialmente vantajoso em aplicações onde os dados são escassos ou custosos de obter. Além disso, esses modelos podem ser re-treinados para especialização em um domínio específico. Este trabalho realiza um levantamento do estado da arte e uma avaliação experimental de TSFM de código aberto em séries temporais provenientes de múltiplos domínios do Monash Time Series Forecasting Repository, um repositório de séries temporais públicas. O desempenho dos modelos é quantificado por métricas como Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) e Mean Absolute Scaled Error (MASE). Os resultados são comparados considerando a previsão de séries temporais em diferentes setores, avaliando sua viabilidade e seu potencial estratégico.

Palavras-chave: TSFM. Séries temporais. Aprendizado profundo. Previsão de séries temporais.

ABSTRACT

In view of the evolution of TSFM and their ability to capture complex temporal dependencies without specific adjustments, these models have been transforming forecasting pipelines across various domains. The *zero-shot* prediction capability of TSFM eliminates the need for massive labeled datasets for training, which is particularly advantageous in applications where data are scarce or costly to obtain. Additionally, these models can be fine-tuned to specialize in specific domains. This work presents a survey of the state of the art and an experimental evaluation of open-source TSFM on time series from multiple domains of the Monash Time Series Forecasting Repository, a public repository of time series data. Model performance is quantified using metrics such as MSE, MAE, MAPE and MASE. The results are compared across different sectors, assessing the feasibility and strategic potential of TSFM-based forecasting.

Keywords: TSFM. Time series. Deep learning. Time series forecasting.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DL Deep Learning.

LSTM Long Short-Term Memory.

MAE Mean Absolute Error.

MAPE Mean Absolute Percentage Error.

MASE Mean Absolute Scaled Error.

ML Machine Learning.

MSE Mean Squared Error.

RNA Rede Neural Artificial.

RNN Rede Neural Recorrente.

SMAPE Symmetric Mean Absolute Percentage Error.

TSFM Time Series Foundation Models.

TTM TinyTimeMixer.

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos, com a presença cada vez maior da tecnologia na vida das pessoas, consolidou-se um ambiente digital em que dados são gerados continuamente e em volumes cada vez maiores. Esse aumento expressivo de dados não apenas amplia os desafios de armazenamento e processamento, mas, sobretudo, impulsiona novas tendências analíticas voltadas à captura de padrões temporais em dados sequenciais, fundamentais em áreas como finanças, saúde, clima, etc (LIANG et al., 2024).

Diante dessa explosão de dados, os avanços do *deep learning* têm se mostrado revolucionários na modelagem de séries temporais. Arquiteturas como *transformer* emergiram como ferramentas potentes para lidar com dependências temporais complexas e variáveis exógenas, superando limitações dos métodos estatísticos tradicionais (CHEN et al., 2023).

Essa evolução culminou em uma nova fronteira de pesquisa: os Time Series Foundation Models (TSFM), inspirados nos modelos fundacionais que transformaram a visão computacional e o processamento de linguagem natural. Esses modelos representam uma mudança de paradigma ao permitir predições *zero-shot* e *few-shot* a partir de pré-treinamento em larga escala, com notável generalização entre domínios e frequências temporais (LIANG et al., 2024).

A proposta desse projeto é aplicar e comparar diferentes TSFM de código aberto para predição de séries temporais em múltiplos domínios, avaliando o desempenho dos principais modelos atuais por meio de métricas como Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) e Mean Absolute Scaled Error (MASE). Espera-se demonstrar que os TSFM podem ser modelos ágeis e eficientes em contextos onde os dados rotulados são escassos ou custosos de obter, alcançando resultados concretos e contribuindo diretamente para a compreensão do potencial e limitações desses modelos.

1.1 OBJETIVO GERAL

O objetivo principal deste trabalho é avaliar diversos modelos de TSFM para predição de séries temporais em múltiplos domínios.

1.2 OBJETIVOS ESPECÍFICOS

- Realizar uma pesquisa do estado da arte sobre TSFM.
- Coletar dados de séries temporais de diferentes domínios disponíveis no repositório Monash.

- Selecionar modelos de TSFM abertos para a realização de testes.
- Realizar testes com previsões *zero-shot* de diversos modelos na base de dados pré-processada.
- Realizar o fine-tuning de pelo menos um modelo TSFM em um dos conjuntos de dados escolhidos.
- Desenvolver uma interface web simples que permita selecionar modelos TSFM e realizar previsões.

1.3 ORGANIZAÇÃO DO TEXTO

O texto está organizado da seguinte forma: O Capítulo 1 introduz o trabalho e seus objetivos. No Capítulo 2 há uma revisão de literatura sobre os temas relacionados ao trabalho. No Capítulo 3 é apresentada a metodologia utilizada para atingir os objetivos definidos. No Capítulo 4 são apresentados os resultados obtidos. No Capítulo 5 são apresentadas as conclusões sobre este trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 SÉRIES TEMPORAIS

Uma série temporal é uma coleção de observações ordenadas no tempo, comum em diversas áreas como economia, meteorologia, energia, etc. Ao contrário de conjuntos de dados independentes, os valores em uma série temporal apresentam dependência temporal, ou seja, o valor atual está frequentemente relacionado a valores passados. Em geral, séries temporais apresentam 4 principais padrões estruturais, que são tendência (mudanças de longo prazo), sazonalidade (flutuações regulares), ciclo (flutuações maiores que um ano) e ruído (variações imprevisíveis), o que influencia diretamente as abordagens de modelagem e previsão (REIS, 2023).

2.1.1 Univariáveis e multivariáveis

Uma série temporal pode ser formalmente representada como uma sequência de observações ordenadas no tempo, em que cada observação corresponde a um vetor de dimensão D. Quando $D = 1$, ou seja, cada ponto no tempo é composto por um único valor escalar, tem-se uma série temporal univariada. Um exemplo típico seria o monitoramento horário do preço de energia elétrica ao longo de vários dias (LIANG et al., 2024).

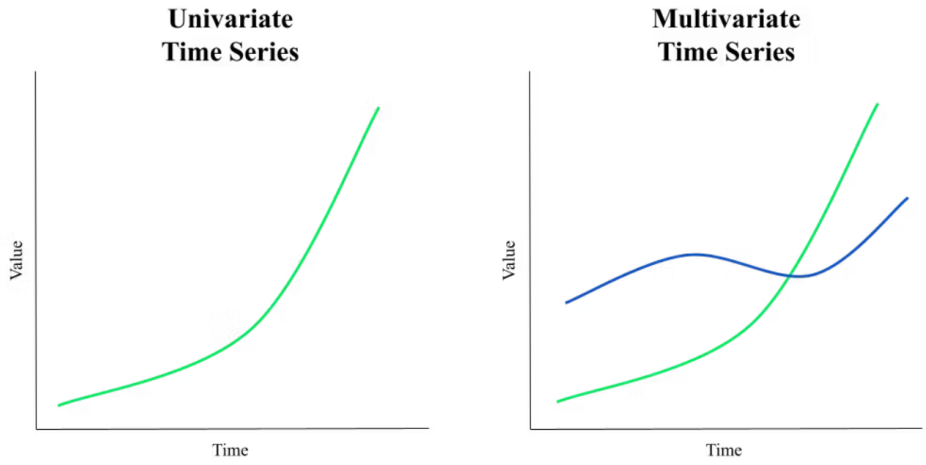
Por outro lado, quando $D > 1$, ou seja, cada observação é composta por múltiplos valores simultâneos, a série temporal é considerada multivariável (LIANG et al., 2024). Esse tipo de série permite representar cenários mais complexos, nos quais diversas variáveis evoluem juntas ao longo do tempo e podem apresentar relações de dependência entre si. Um exemplo seria o acompanhamento conjunto do preço da energia, da geração por usinas fotovoltaicas e da geração por usinas eólicas. Modelos multivariáveis permitem identificar relações entre variáveis que influenciam diretamente a qualidade das previsões. (MONTGOMERY; JENNINGS; KULAHCI, 2015).

Na Figura 1a é possível visualizar uma série temporal de apenas uma variável, já na Figura 1b a série temporal apresenta duas variáveis, sendo considerada multivariável por apresentar mais de um valor observado ao longo do tempo.

2.1.2 Estacionariedade

A estacionariedade é uma propriedade estatística fundamental em muitas abordagens de modelagem de séries temporais. Uma série é dita estacionária quando suas propriedades, como média, variância e autocorrelação, permanecem constantes ao longo do tempo. (MONTGOMERY; JENNINGS; KULAHCI, 2015)

Figura 1 – Diferença entre séries temporais univariáveis e multivariáveis



(a) Série temporal univariável (b) Série temporal multivariável

Fonte: MongoDB (2025)

2.1.3 Previsão de séries temporais

A previsão de séries temporais é muito utilizada em diversas aplicações reais, como antecipar a demanda por energia, prever preços de mercado ou estimar o número de acessos a um site. No entanto, prever o futuro com base em dados passados envolve diversos desafios, como a variabilidade dos padrões temporais, a ocorrência de mudanças repentinas e a limitação de informações disponíveis sobre eventos futuros. Para realizar essa previsão, metodologias baseadas em aprendizado de máquina (ML) têm ganhado cada vez mais relevância (DAS et al., 2024).

2.1.3.1 Métricas de avaliação

A avaliação de modelos de previsão de séries temporais requer métricas capazes de quantificar a diferença entre os valores reais y_t e os valores previstos $\hat{y}_t$. Essa diferença é geralmente expressa por um erro $e_t = y_t - \hat{y}_t$, que mostra o quanto o modelo errou em relação ao valor real.

De acordo com Montgomery, Jennings e Kulahci (2015), duas métricas clássicas muito utilizadas em trabalhos de avaliação de previsões em séries temporais são:

Erro Absoluto Médio (MAE): definido pela equação 2.1, é a média dos valores absolutos dos erros, mais robusto a outliers e interpretável em unidades da própria série.

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t| \tag{2.1}$$

Erro Quadrático Médio (MSE): representado pela equação 2.2, é a média dos

quadrados dos erros, que penaliza fortemente grandes desvios.

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2 \tag{2.2}$$

Apesar de úteis, ambas são métricas dependentes da escala da série, o que dificulta comparações entre domínios distintos. Como neste trabalho serão comparados diferentes modelos em diferentes conjuntos de dados, essas métricas não são suficientes pois a escala de medida entre cada conjunto é diferente. Por isso, acrescentam-se neste trabalho duas métricas que mensuram o erro independente da escala da série:

Erro Percentual Absoluto Médio (MAPE), representado pela equação 2.3, é o erro relativo em porcentagem (geralmente expresso como fração/decimal), calculando a média dos erros absolutos em relação ao valor real observado. Essa métrica é especialmente útil quando séries apresentam diferentes magnitudes, pois permite comparar desempenhos independentemente da escala (SKTIME, 2025a).

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \tag{2.3}$$

Erro Absoluto Escalado Médio (MASE), definido pela equação 2.4, avalia o desempenho em relação a um modelo de referência Naive. Assim, valores de MASE inferiores a 1 indicam que o modelo supera o Naive, enquanto valores maiores sinalizam pior desempenho (SKTIME, 2025b).

$$\text{MASE} = \frac{\frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|}{\frac{1}{n-1}\sum_{t=2}^{n}|y_t - y_{t-1}|} \tag{2.4}$$

Naive baseline: o modelo Naive, definido pela equação 2.5, assume que o próximo valor será igual ao último observado, servindo como referência básica em previsões de séries temporais. Ele pode ser usado isoladamente para comparação direta dos erros ou como base para métricas como o MASE, que avaliam o quanto um modelo supera essa previsão simples (HYNDMAN; ATHANASOPOULOS, 2018).

$$\text{Naive} = y_{t-1} \tag{2.5}$$

Dessa forma, a combinação de todas essas métricas fornece uma análise equilibrada, considerando tanto aspectos absolutos quanto relativos dos erros, além de permitir comparações entre diferentes séries temporais.

2.2 MACHINE LEARNING

Aprendizado de máquina, do inglês Machine Learning (ML), é um ramo da inteligência artificial que se refere a sistemas capazes de aprender com base em padrões

encontrados em dados, sem necessariamente serem programados para cada situação. Esses sistemas de ML são amplamente utilizados para realizar previsões ou tomar decisões com base em novas informações (MURPHY, 2012).

De forma geral, o ML é dividido em dois tipos principais:

- **Supervisionado:** Modelos supervisionados exigem que os dados de treinamento venham acompanhados de rótulos, ou seja, informações sobre a saída esperada para cada entrada. Durante o processo de treinamento, são mostrados ao modelo exemplos de dados junto com seus respectivos rótulos, permitindo que ele aprenda a relação entre entrada e saída (MURPHY, 2012).

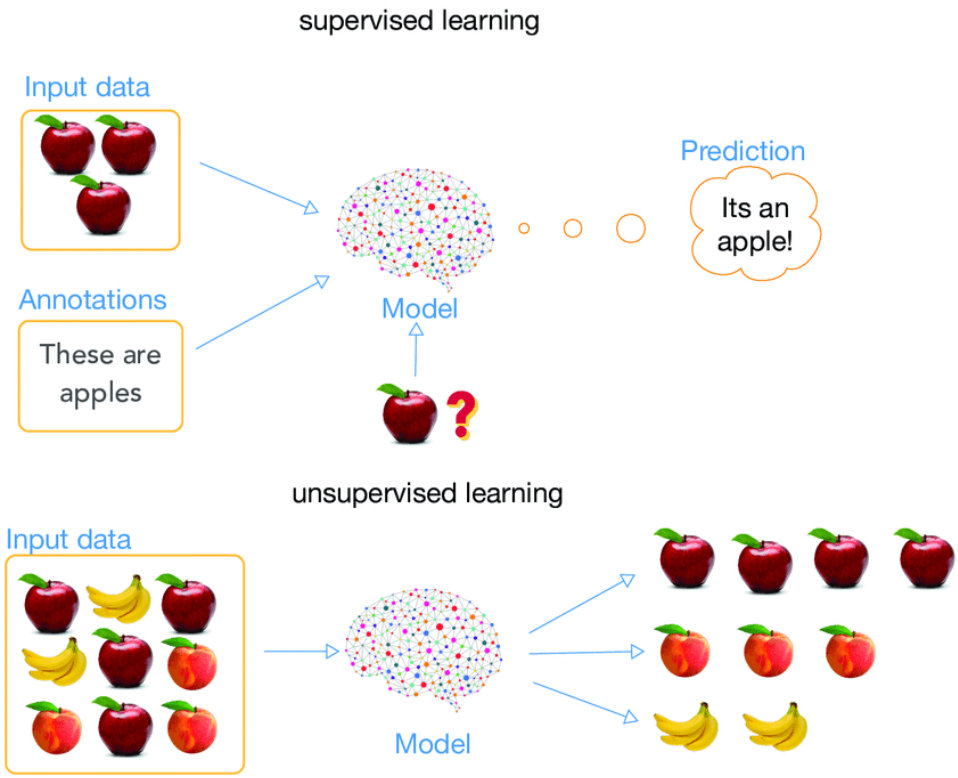
  Por exemplo, imagine que o objetivo é classificar séries temporais em três categorias: A, B ou C. Para treinar o modelo, são fornecidas diversas séries já rotuladas com suas respectivas categorias, permitindo que ele aprenda a distinguir entre os diferentes tipos.

- **Não supervisionado:** Modelos não supervisionados, por outro lado, não utilizam rótulos nos dados. Em vez disso, eles buscam identificar estruturas ocultas ou padrões nos dados, como agrupamentos ou distribuições (MURPHY, 2012). Um caso de uso comum é a detecção de anomalias. Suponha que um conjunto de dados é composto por diversas séries temporais com comportamento estável e o objetivo é identificar quando ocorre uma anomalia. Dessa forma, um modelo não supervisionado pode aprender o padrão de estabilidade dessas séries durante o treinamento e, quando encontrar uma nova série que fuja desse padrão, reconhecê-la como anômala, mesmo sem nunca ter sido treinado com exemplos explícitos de anomalias.

A Figura 2 ilustra a diferença entre aprendizado supervisionado e não supervisionado. Na parte superior da imagem, o modelo recebe dados de entrada acompanhados de rótulos, que no exemplo ilustrado é a indicação dos objetos serem maçãs. Esse processo caracteriza o aprendizado supervisionado, pois o modelo aprende a mapear entradas para saídas conhecidas. Por outro lado, a parte inferior da figura representa o aprendizado não supervisionado, no qual o modelo recebe apenas os dados de entrada, sem rótulos. Nesse caso, o objetivo do modelo é identificar padrões ou agrupamentos naturalmente presentes nos dados, como organizar diferentes frutas por similaridade visual, mesmo sem ter recebido rótulos de cada uma, ensinando exemplos de cada fruta previamente.

Além dos tipos clássicos supervisionado e não supervisionado, existem variações que combinam ou estendem essas abordagens. Uma delas é o aprendizado auto-supervisionado, muito utilizado em modelos grandes como os modelos pré-treinados que serão abordados neste trabalho. Nessa estratégia, os próprios dados geram sinais de aprendizagem por meio de rótulos artificiais criados automaticamente ao mascarar partes de uma

Figura 2 – Exemplo de diferença entre aprendizado supervisionado e não supervisionado



Fonte: Devopedia (2022)

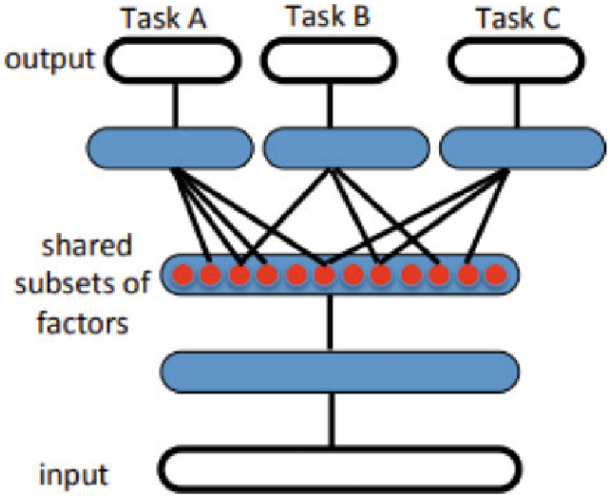
sequência, prever o próximo trecho ou comparar visões diferentes da mesma amostra, permitindo que o modelo aprenda representações úteis sem nenhuma anotação humana (BERGMANN, 2023).

2.2.1 Representation Learning

Representation Learning é um conceito essencial em ML e refere-se à capacidade dos modelos de aprender automaticamente representações ou características úteis diretamente a partir dos dados brutos (LECUN; BENGIO; HINTON, 2015). Isso é particularmente importante em problemas complexos, como séries temporais onde padrões temporais e variações podem ser difíceis de capturar de maneira tradicional.

A Figura 3 ilustra o processo de transformação de dados brutos em um conjunto de fatores internos, também chamados de representações. Esses fatores correspondem a características abstratas aprendidas automaticamente pelo modelo durante o processo de treinamento e constituem uma camada intermediária que explica aspectos relevantes da entrada (BENGIO; COURVILLE; VINCENT, 2014). Embora a figura tenha sido originalmente apresentada no contexto de múltiplas tarefas, o princípio fundamental mostrado permanece válido mesmo em modelos de tarefa única: a representação interna é composta por fatores que organizam a informação do dado de forma mais valiosa e útil para a predição.

Figura 3 – Exemplo de transformação de dados brutos em representações



Fonte: Bengio, Courville e Vincent (2014)

O aprendizado dessas representações pode ocorrer em diferentes paradigmas de treinamento de ML. No aprendizado supervisionado, as representações são moldadas diretamente pelos rótulos, de modo que o modelo aprende características internas que facilitam a previsão da variável-alvo. No aprendizado não supervisionado, o modelo aprende representações a partir da estrutura intrínseca dos dados, buscando capturar padrões, correlações e representações sem o uso de rótulos externos. Já no aprendizado autosupervisionado, o próprio conjunto de dados é utilizado para gerar tarefas auxiliares, permitindo que o modelo aprenda representações úteis mesmo em cenários com pouca disponibilidade de rótulos.

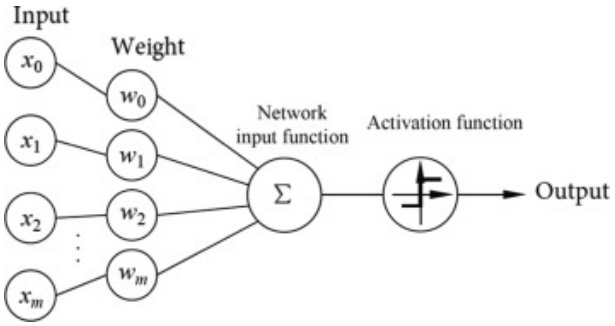
2.2.2 Redes Neurais

As redes neurais artificiais (RNAs) são modelos computacionais inspirados no funcionamento paralelo e distribuído do cérebro humano. Elas têm como característica principal a capacidade de aprender padrões e relações complexas a partir de dados, generalizando seu aprendizado para novas situações. Isso as torna especialmente eficazes em tarefas como classificação, regressão e reconhecimento de padrões, que exigem modelagem não linear e adaptativa (HAYKIN, 1998).

De acordo com Goodfellow, Bengio e Courville (2016), uma RNA é composta por alguns elementos fundamentais que colaboram para o seu funcionamento:

- **Neurônios Artificiais:** São as unidades básicas de processamento da rede. Cada neurônio recebe sinais de entrada, calcula uma soma ponderada (que envolve pesos e viéses) e aplica uma função de ativação para produzir uma saída.
- **Sinapses (Pesos Sinápticos):** Representam as conexões entre os neurônios. Cada sinapse possui um peso que ajusta a importância do sinal recebido, e esses pesos são

Figura 4 – Perceptron



Fonte: Liang (2020)

atualizados durante o treinamento para que a rede aprenda as relações presentes nos dados.

- **Camadas:** As Rede Neural Artificials (RNAs) são estruturadas em camadas. A camada de entrada recebe os dados brutos, as camadas intermediárias (ou *camadas ocultas*) processam as informações progressivamente, e a camada de saída gera o resultado final da rede.

2.2.2.1 Perceptron

O Perceptron é uma rede neural de camada única, funcionando como um classificador linear binário. Seu funcionamento matemático é fundamentado na combinação linear dos atributos de entrada e na aplicação de uma função de ativação para produzir a saída (HAYKIN, 1998). A Figura 4 ilustra a topologia de um perceptron.

De acordo com Haykin (1998), dado um vetor de entrada $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top$ e um vetor de pesos $\mathbf{w} = [w_1, w_2, \ldots, w_n]^\top$, o perceptron calcula a ativação a como:

$$a = \mathbf{w}^\top \mathbf{x} + b \tag{2.6}$$

Na equação 2.6, o termo b representa o viés do modelo. A saída y do perceptron é então obtida aplicando uma função de ativação $f(a)$, que pode ser descrita da seguinte forma:

$$y = f(a) = \begin{cases} 1, & \text{se } a \geq 0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.7}$$

A equação 2.7 define a decisão binária do perceptron, classificando a entrada em uma das duas classes. Durante o treinamento, os pesos do perceptron são ajustados iterativamente com base no erro de classificação. Para cada exemplo de treinamento $(\mathbf{x}^{(i)}, y^{(i)})$, os pesos são atualizados segundo a seguinte regra:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta(y^{(i)} - \hat{y}^{(i)})\mathbf{x}^{(i)} \tag{2.8}$$

Na equação 2.8, η é a taxa de aprendizado e $\hat{y}^{(i)}$ é a previsão atual do perceptron. Essa atualização busca minimizar o erro de classificação, ajustando os pesos para que o hiperplano de decisão se aproxime das fronteiras corretas.

Apesar de sua simplicidade, o perceptron é limitado à classificação de dados linearmente separáveis. Essa limitação motivou a criação das redes neurais multicamadas, que são capazes de lidar com não linearidades mais complexas e capturar padrões que modelos lineares não conseguem identificar (GOODFELLOW; BENGIO; COURVILLE, 2016).

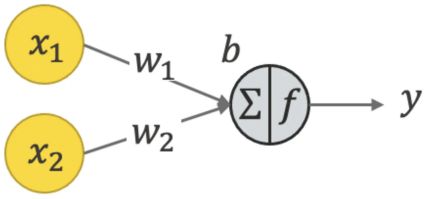
2.2.3 Deep Learning

A partir das redes neurais artificiais (RNA), surgiu a demanda por modelos mais profundos e expressivos, capazes de lidar com problemas mais complexos. Essa necessidade é impulsionada por conjuntos de dados cada vez maiores e por relações temporais ou espaciais de difícil captura com técnicas tradicionais. O Deep Learning (DL) representa justamente essa evolução: um ramo do aprendizado de máquina que explora arquiteturas neurais com múltiplas camadas, capazes de extrair representações hierárquicas e não lineares dos dados. Enquanto redes simples, como o perceptron, lidam com relações básicas, o Deep Learning (DL) constrói sucessivas abstrações, permitindo que modelos aprendam características mais ricas e relevantes diretamente dos dados brutos (GOODFELLOW; BENGIO; COURVILLE, 2016).

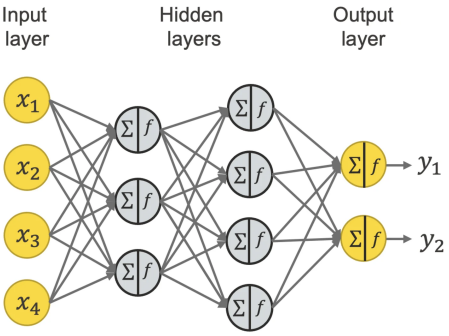
Essas características tornam o DL capaz de resolver problemas que antes eram considerados intratáveis por métodos tradicionais. Por meio de seu funcionamento em camadas profundas, as redes de DL automatizam o processo de *Representation Learning*, transformando dados brutos em representações abstratas e informativas. Modelos de DL podem ser vistos como modelos de *Representation Learning* compostos por múltiplos níveis de representação que aprendem progressivamente funções cada vez mais complexas, de forma inteiramente automática (LECUN; BENGIO; HINTON, 2015).

É comum organizar a evolução dos modelos mais relevantes de *DL* no contexto de previsão de séries temporais de forma cronológica, partindo das primeiras arquiteturas sequenciais até alcançar as abordagens mais sofisticadas utilizadas atualmente. A previsão de séries temporais com DL passou por uma evolução significativa, na qual cada nova arquitetura procurou superar limitações da anterior, especialmente no que diz respeito à capacidade de capturar dependências temporais de longo alcance, lidar com múltiplas variáveis e melhorar a eficiência computacional. A seguir são apresentados os principais modelos utilizados nessa tarefa, organizados de forma cronológica.

Figura 5 – Comparação entre a arquitetura de um perceptron e de uma rede neural profunda



(a) Arquitetura de um perceptron



(b) Arquitetura de uma rede neural profunda

Fonte: Melcher (2023)

2.2.3.1 Redes Neurais Profundas

As redes neurais profundas constituem a evolução natural do perceptron. Enquanto o perceptron realiza apenas uma transformação entre as entradas e a saída, exigindo que os dados possam ser separados de forma simples, as arquiteturas profundas empilham diversas camadas de processamento. Cada nova camada refina o que a anterior já aprendeu, permitindo que o modelo descubra padrões mais sutis e relações complexas diretamente a partir dos dados. Isso amplia consideravelmente a capacidade de resolver problemas que vão além do alcance de um único perceptron (GOODFELLOW; BENGIO; COURVILLE, 2016).

A Figura 5 mostra essas diferenças. Na subfigura 5a, observa-se o perceptron com sua única camada de conexões. Já a subfigura 5b apresenta uma rede profunda feed-forward, ou seja, a informação flui em apenas uma direção: ela inicia pela camada de entrada, responsável apenas por definir a quantidade de atributos, sem realizar cálculos. Em seguida, há duas camadas ocultas em sequência: a primeira recebe os valores de entrada e gera a saída de seus três neurônios, enquanto a segunda utiliza essas saídas para calcular os quatro neurônios seguintes. Por fim, a camada de saída toma esses quatro valores como entrada e produz a predição final. Esse encadeamento entre camadas internas demonstra como a profundidade permite construir representações cada vez mais abstratas dos dados. Esse mecanismo é um dos elementos centrais dos modelos de DL (MELCHER, 2023).

2.2.3.2 Redes Neurais Recorrentes (RNN)

Redes Neurais Recorrentes (RNN) são redes neurais artificiais projetadas para processar dados sequenciais, tornando-se um padrão inicial para tarefas de previsão de

séries temporais, devido à sua capacidade de capturar dependências de curto prazo entre pontos consecutivos (GOODFELLOW; BENGIO; COURVILLE, 2016). Ao contrário das redes profundas feed-forward, as Rede Neural Recorrentes (RNNs) introduzem um estado oculto que se propaga de um instante para o próximo, permitindo que o modelo retenha informações recentes. Em cada passo de tempo, o estado é combinado com a entrada corrente, gerando uma nova representação interna que reflete o histórico da sequência. No entanto, as RNNs enfrentam dificuldades para manter informações por janelas temporais longas (AMAZON WEB SERVICES, 2025).

2.2.3.3 Long Short-Term Memory (LSTM)

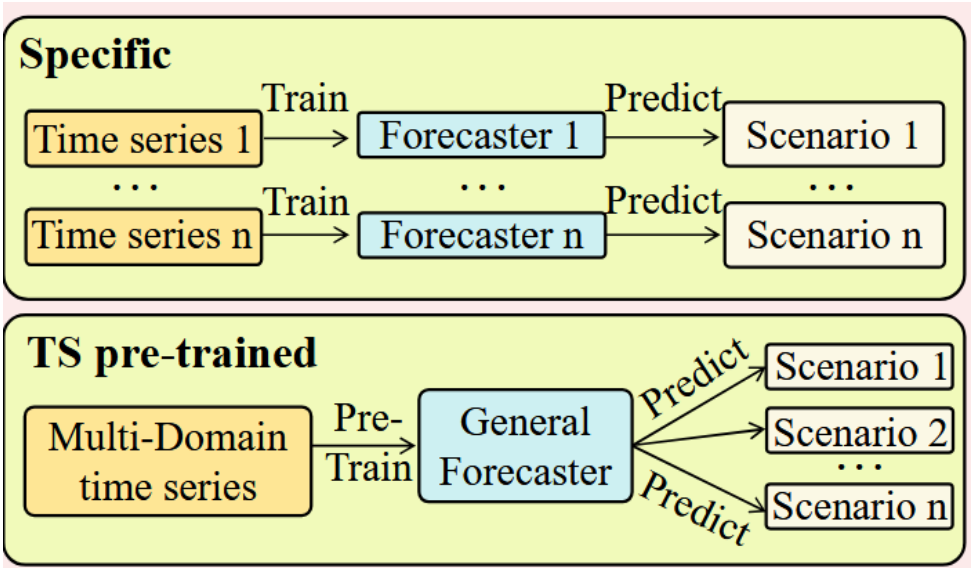
A LSTM aperfeiçoa as RNNs ao incorporar um mecanismo de memória capaz de armazenar informações relevantes por períodos prolongados. Esse componente acumula sinais ao longo do tempo e decide quando apagar ou manter seu conteúdo, o que reduz drasticamente a perda de informação temporal. Como resultado, a LSTM é altamente eficaz na previsão de séries temporais, especialmente em janelas de previsão mais longas (LECUN; BENGIO; HINTON, 2015).

2.2.3.4 Transformer

O *Transformer* é atualmente uma das arquiteturas mais disruptivas da inteligência artificial. Ele abriu caminho para avanços significativos em diversos campos, como processamento de linguagem natural, visão computacional e análise e previsão de séries temporais. Como foi originalmente desenvolvido para linguagem natural, a versão proposta em 2017 precisou passar por diversos ajustes para funcionar adequadamente com séries temporais, já que dados textuais e dados numéricos possuem características bastante distintas. A partir dessas adaptações, diversos modelos começaram a ser desenvolvidos com base no *Transformer*, incorporando melhorias para superar suas limitações e torná-lo mais eficaz na previsão de séries temporais (ZHOU et al., 2021). De acordo com Zhou et al. (2021), a aplicação direta do *Transformer* em tarefas de séries temporais de longo prazo apresenta três limitações principais:

- **Alto custo computacional**: o mecanismo de autoatenção tradicional possui complexidade quadrática, o que dificulta seu uso com sequências muito longas.
- **Uso excessivo de memória**: ao empilhar várias camadas com entradas longas, o modelo consome muita memória, limitando sua escalabilidade.
- **Previsão lenta**: o processo de decodificação passo a passo torna a previsão de sequências longas lenta e propensa a acúmulo de erro.

Figura 6 – Diferença entre modelos específicos e TSFM.



Fonte: Adaptado de Li et al. (2025)

Diversos modelos posteriores, baseados na arquitetura original do *Transformer*, conseguiram superar essas limitações. Desde então, os *Transformers* têm sido bastante explorados em pesquisas voltadas à previsão de séries temporais. Novas ideias continuam surgindo e ampliando o uso dessa arquitetura nesse campo, trazendo previsões cada vez mais precisas.

2.3 MODELOS FUNDACIONAIS DE SÉRIES TEMPORAIS (TSFM)

Modelos Fundacionais de Séries Temporais, ou *Time Series Foundation Models* (TSFM), são redes neurais artificiais geralmente baseadas na arquitetura *Transformer*, pré-treinadas em séries temporais de múltiplos domínios (energia, clima, saúde, finanças, etc). A meta é aprender representações gerais e reutilizáveis, permitindo boa performance sem treinar um modelo do zero para cada aplicação (LIANG et al., 2024).

A Figura 6 ilustra a diferença entre os modelos tradicionais (denominados Specific na figura) e os modelos pré-treinados, mostrando que os TSFM funcionam como preditores gerais aplicáveis a diferentes cenários.

Os principais TSFM têm a arquitetura baseada na estrutura do *Transformer*, devido à sua capacidade de capturar dependências de longo prazo em sequências temporais com alta eficiência. Esses modelos adaptam o mecanismo de atenção para lidar com a natureza contínua e multiescala dos dados temporais, sendo capazes de realizar previsões robustas mesmo em horizontes longos.

De acordo com Liang et al. (2024), uma vez que um modelo é pré-treinado, o mesmo TSFM pode ser reutilizado de três maneiras:

- ***Zero-shot*** – Aplicação direta a um domínio nunca visto, sem dados rotulados adicionais. Útil quando rotular é caro ou inviável.
- ***Few-shot*** – Ajuste leve usando um subconjunto *mínimo* de exemplos rotulados (tipicamente ≤ 5 % do que seria exigido num treinamento completo), suficiente para adaptar as camadas de saída ou normalizar escalas (LI et al., 2025).
- ***Fine-tuning*** – Re-treino parcial ou total das camadas do modelo no novo domínio.

2.4 MONASH TIME SERIES FORECASTING REPOSITORY

O *Monash Time Series Forecasting Repository* foi proposto por pesquisadores da Monash University para preencher a lacuna de um acervo abrangente que permita avaliar modelos globais ou multivariados de previsão em conjuntos heterogêneos de séries temporais. O objetivo é disponibilizar diversos conjuntos de dados acessíveis publicamente, provenientes de domínios variados e com diferenças de frequência, tamanho das séries e presença de valores ausentes, criando assim uma base padronizada para comparação de novos algoritmos de previsão (GODAHEWA et al., 2021).

Ao todo, o repositório soma trinta conjuntos de dados principais. Eles abrangem áreas como finanças, turismo, web, energia, transporte, saúde e macroeconomia, e apresentam amostragens que variam de anual até registros de alta frequência a cada quatro segundos. A partir desses trinta conjuntos, derivam-se cinquenta e oito variações, obtidas por diferenças de frequência ou pela criação de versões com e sem valores ausentes, o que amplia o leque de cenários de avaliação disponíveis (GODAHEWA et al., 2021).

Esses conjuntos de dados são referência na área de previsão de séries temporais e são amplamente utilizados na avaliação do desempenho dos principais benchmarks e modelos no estado da arte (LI et al., 2025). A padronização, diversidade de domínios e facilidade de acesso contribuem para que sejam adotados como base comum em estudos comparativos para modelos de previsão.

3 METODOLOGIA

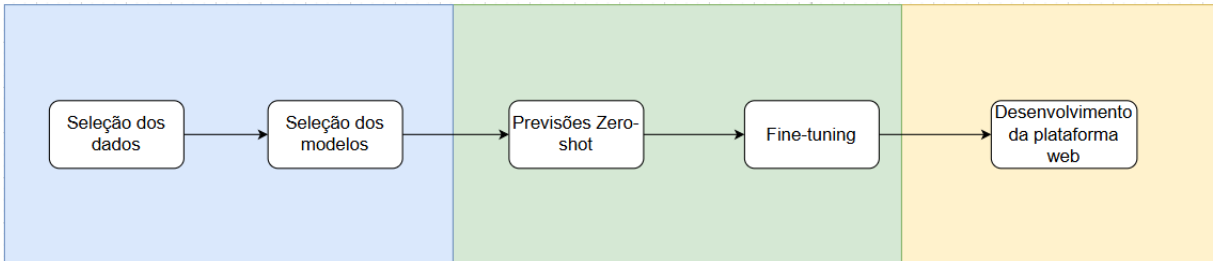
Neste capítulo é descrita a metodologia que foi seguida para atingir os objetivos deste projeto. A Figura 7 ilustra o fluxo geral das etapas que seguidas, mostrando em azul, as etapas de seleção dos dados e dos modelos utilizados, e em verde, as etapas de previsão zero-shot e fine-tuning, por fim, em amarelo, a etapa de construção da plataforma web.

3.1 SELEÇÃO DOS CONJUNTOS DE DADOS

Toda a avaliação foi conduzida utilizando subconjuntos do *Monash Time Series Forecasting Repository*, uma coleção amplamente reconhecida na área de previsão de séries temporais. Os conjuntos foram escolhidos por cobrirem diferentes domínios de aplicação e frequências temporais, de forma a garantir diversidade nos cenários de teste. Foram selecionados os seguintes datasets:

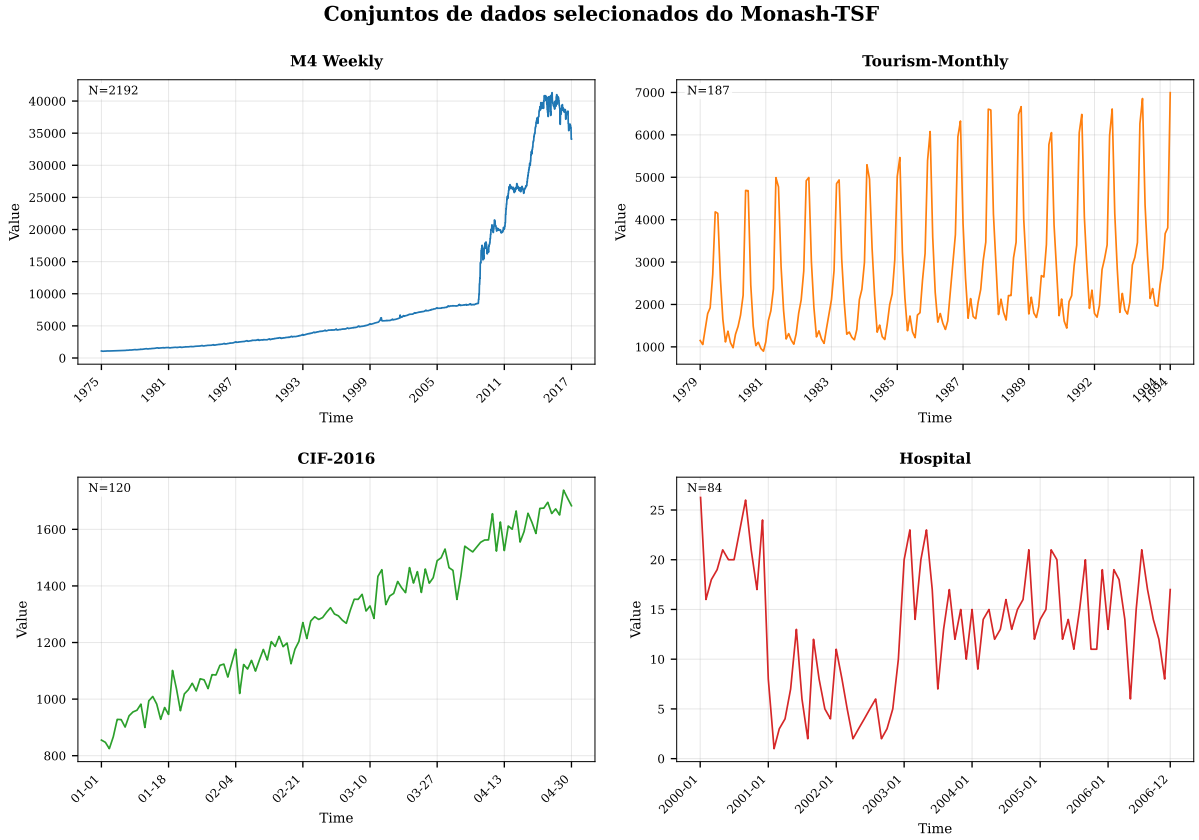
- **Tourism-Monthly**: Conjunto de dados apresentado originalmente em 2011 que contém 366 séries temporais mensais de dados de turismo, e são popularess por serem utilizados em competições de previsão de séries temporais (ATHANASOPOULOS et al., 2011);
- **CIF-2016**: Conjunto de 72 séries temporais mensais do setor bancário, das quais 24 são séries reais e 48 geradas artificialmente, conforme definido para a competição de previsão CIF 2016. Nesta competição, 57 séries tinham horizonte de previsão de 12 meses e as outras 15 séries tinham horizonte de 6 meses (STEPNICKA; BURDA, 2017);
- **Hospital**: Contém 767 séries temporais mensais representando contagens de pacientes, abrangendo de janeiro de 2000 a dezembro de 2006 (HYNDMAN, 2015);

Figura 7 – Fluxo geral das etapas do projeto



Fonte: Próprio autor

Figura 8 – Gráficos de séries temporais dos conjuntos de dados selecionados do Monash Time Series Forecasting Repository.



Fonte: Próprio autor

- **M4 Competition Weekly**: O M4 Competition Weekly é um subconjunto do desafio M4 de previsão de séries temporais, composto por 359 séries temporais de frequência semanal (MAKRIDAKIS; SPILIOTIS; ASSIMAKOPOULOS, 2020).

A Figura 8 apresenta o gráfico de cada um dos conjuntos de dados selecionados do Monash Time Series Forecasting Repository.

Além dos conjuntos de dados selecionados do repositório Monash, também foi incluído o conjunto Bike Sharing Dataset, do UCI Machine Learning Repository. Esse conjunto possui frequência horária e diária e complementa os demais dados selecionados. Ele contém registros de aluguel de bicicletas do sistema Capital Bikeshare em Washington, D.C., abrangendo medições horárias e diárias de 2011 a 2012, incluindo variáveis como número de bicicletas alugadas, condições climáticas e fatores sazonais para análise de demanda (FANAEE-T, 2013).

A inclusão desse conjunto foi necessária para garantir que nenhum dos dados utilizados na avaliação tivesse sido empregado no pré-treinamento dos TSFM escolhidos, uma vez que muitos desses modelos utilizam séries de frequência horária do Monash em seu treinamento. Assim, o *Bike Sharing Dataset* foi selecionado para complementar o

conjunto de avaliação, permitindo incorporar uma granularidade horária que não aparece no pré-treinamento e facilitando a escolha dos modelos apresentados na próxima seção. Ressalta-se que, caso fossem considerados apenas TSFM que não utilizam os conjuntos horários do Monash, haveria maior dificuldade para encontrar modelos que atendessem a esse critério.

3.2 SELEÇÃO DOS MODELOS

O estudo buscou comparar arquiteturas modernas de TSFM que ocupam as melhores colocações em rankings e *benchmarks* atuais. Com o tema em alta e novos modelos surgindo constantemente, foram selecionados três modelos que persistem no estado da arte com alto aproveitamento nos *benchmarks* de maior destaque (LI et al., 2025).

Os modelos escolhidos foram:

- **Moirai (Salesforce)**: Lançado em 2024 pela Salesforce, o Moirai introduziu um modelo fundacional universal para séries temporais, capaz de realizar previsões zero-shot em múltiplos domínios e frequências. Foi pré-treinado no conjunto LOTSA (27 bilhões de observações). Ele aborda desafios como aprendizado multi-frequência e séries multivariadas arbitrárias por meio de melhorias na arquitetura Transformer, e alcançou desempenho competitivo ou superior a modelos especializados sem necessidade de re-treinar em cada novo conjunto de dados, demonstrando o potencial de modelos pré-treinados para previsão de uso geral (WOO et al., 2024). Neste trabalho, foi utilizada a versão **moirai-1.0-R-small**, uma variante compacta do modelo original;

- **Chronos (Amazon)**: Apresentado em 2024 pela Amazon, o Chronos aplica abordagens de modelo de linguagem às séries temporais, "aprendendo a linguagem" desses dados. Foi pré-treinado em uma grande coleção de conjuntos públicos (complementados por séries sintéticas via processos gaussianos), explicitamente sem utilizar o Monash, que é reservado apenas para avaliação. Ele converte valores em tokens discretos e utiliza Transformers para treinar um modelo probabilístico que prevê próximas sequências de valores, obtendo resultados superiores em benchmarks tradicionais e desempenho zero-shot comparável ou melhor que modelos treinados especificamente nos dados-alvo (ANSARI et al., 2024). Neste estudo, foi empregada a versão **chronos-t5-tiny**, baseada na arquitetura T5 e otimizada para eficiência computacional;

- **TinyTimeMixer (IBM)**: Desenvolvido pelo IBM Research e lançado em 2024, o TTM teve a proposta de ser um modelo fundacional menor e mais rápido para previsão de séries temporais com menos de 1 milhão de parâmetros. Foi pré-treinado em

séries de frequência diária ou menor dos repositórios LibCity e Monash. Entretanto, nenhum desses conjuntos do Monash utilizados no pré-treino do TinyTimeMixer foi empregado nos dados deste trabalho. Focado em rapidez e eficiência, ele foi projetado para previsões zero-shot e few-shot de séries multivariadas com baixo custo computacional, superando diversos modelos maiores em cenários com poucos ou nenhum dado de treino (EKAMBARAM et al., 2024). A versão selecionada foi o **TTM-1024-96**.

3.3 PREVISÕES ZERO-SHOT

3.3.1 Parâmetros de previsão

Conhecendo-se como janela de contexto (C) o número de observações passadas utilizadas como entrada do modelo, e como horizonte de previsão (H) o número de passos futuros que o modelo deve estimar (WOO et al., 2024), as escolhas desses parâmetros foram orientadas pelas sazonalidades dos conjuntos de dados utilizados, de modo a cobrir ciclos relevantes e permitir avaliação em curto, médio e longo prazos. As configurações específicas por frequência estão consolidadas na Tabela 1.

Para cada série, os últimos H pontos foram reservados como conjunto de teste, enquanto o restante serviu como histórico disponível. Para viabilidade computacional, quando aplicável, limitou-se a 100 o número de séries avaliadas por conjunto. Nas séries mensais e semanais, foram avaliadas múltiplas combinações de janelas de contexto e horizontes; no conjunto horário de *bike sharing*, adotou-se um par fixo de valores, compatível com a granularidade desse conjunto de dados específico.

Tabela 1 – Janelas de contexto fixas (C) e horizontes de previsão (H) por frequência

Frequência	Janelas de contexto (C)	Horizontes (H)
Mensal	$\{24, 36, 48\}$	$\{12, 24, 40\}$
Semanal	$\{52, 104\}$	$\{12, 24, 40\}$
Horária (Bike-Sharing)	1024	96

Fonte: Próprio autor

Essa configuração garante, por exemplo, que em dados mensais o contexto abranja de dois a quatro anos de histórico, e em dados semanais, de um a dois anos, capturando ciclos e variações anuais sem tornar computacionalmente custoso. Em dados horários, a combinação fixa escolhida provê histórico suficiente para padrões diários e semanais e um horizonte compatível com janelas de poucos dias. As escolhas seguem práticas recorrentes em *benchmarks* recentes, que avaliam múltiplos horizontes e utilizam diferentes comprimentos de contexto para contemplar dados de diferentes frequências (LI et al., 2025).

3.3.2 Métricas de avaliação

A performance dos modelos foi avaliada por meio de quatro métricas, calculadas separadamente para cada série e depois agrupadas por meio da média para obter o valor final.

- **MAE**: erro absoluto médio, diretamente na escala dos dados;
- **MSE**: erro quadrático médio, penalizando mais fortemente grandes desvios;
- **MAPE**: erro absoluto médio percentual, menos sensível a valores próximos de zero;
- **MASE**: erro absoluto médio escalado em relação a uma *baseline* sazonal. Valores < 1 indicam desempenho superior ao *baseline*.

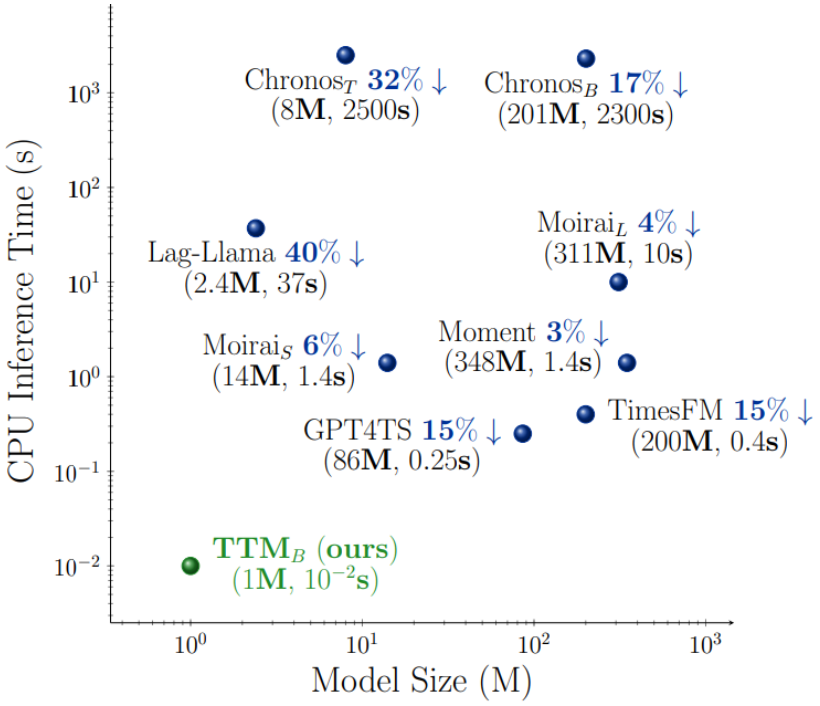
A escolha das métricas de avaliação foi baseada em práticas consolidadas em *benchmarks* recentes de previsão de séries temporais, como o TSFM-Bench, que utiliza MAE e MSE para a construção de um *benchmark* de diferentes modelos de previsão de séries temporais (LI et al., 2025). Além disso, também foram consideradas métricas consideradas padrão no Monash Time Series Forecasting Repository, como Symmetric Mean Absolute Percentage Error (SMAPE) e MASE (GODAHEWA et al., 2021). No entanto, optou-se por substituir o SMAPE pelo MAPE, uma vez que os conjuntos de dados escolhidos não apresentam valores próximos de zero, condição na qual o SMAPE se torna mais adequado. Dessa forma, o conjunto final de métricas (MAE, MSE, MAPE e MASE) buscou equilibrar as métricas utilizadas nos principais *benchmarks* da literatura, permitindo uma avaliação robusta.

3.4 FINE-TUNING

O modelo selecionado para o *fine-tuning* foi o TinyTimeMixer (TTM), escolhido por ser significativamente menor que os demais modelos, assim como mostra a Figura 9, que viabiliza o ajuste fino em ambientes com recursos computacionais limitados. O processo de fine-tuning foi realizado em uma GPU Tesla T4 com 16GB de memória, disponibilizada pelo plano gratuito do Google Colab. O conjunto de dados utilizado foi o *bike_sharing*, o mesmo empregado nos experimentos de zero-shot. O objetivo do fine-tuning é adaptar o modelo para capturar de maneira mais precisa as dinâmicas específicas desse conjunto, verificando a capacidade do modelo melhorar seu desempenho se especializando nesse conjunto de dados.

A arquitetura do TTM estrutura-se em três componentes principais: o backbone, que extrai *features* temporais universais através de *mixer layers* que processam patches e canais da série histórica; o *decoder*, que gera predições futuras e, no modo 'mix_channel'

Figura 9 – Diferença de tamanho entre o TTM e outros TSFMs



Fonte: Ekambaram et al. (2024)

utilizado aqui, permite interação explícita entre variáveis através de mecanismos de atenção; e o *head*, que realiza o mapeamento final para valores numéricos. Esta arquitetura hierárquica separa responsabilidades: o *backbone* atua como extrator genérico e transferível, enquanto o *decoder* assume o papel de adaptador específico ao domínio.

Na abordagem utilizada para o processo de *fine-tuning*, que segue as recomendações feitas por Ekambaram et al. (2024), todo o backbone encoder foi completamente congelado, representando 59% dos 948.920 parâmetros totais do modelo. Esta decisão preserva o conhecimento pré-treinado em larga escala, que incluiu milhares de datasets e aprendeu padrões temporais universais como detecção de tendências, identificação de sazonalidade e captura de autocorrelação. Em contrapartida, todo o decoder foi mantido como treinável, com especial atenção aos módulos de *channel feature mixer* que possuem blocos de atenção e redes MLP para mistura de informações. Estes componentes, somando 41% dos parâmetros do modelo, foram submetidos a treinamento supervisionado para aprender dependências e correlações específicas do dataset alvo.

A justificativa para esta estratégia fundamenta-se na observação de que diferentes componentes da arquitetura possuem naturezas complementares: o backbone extrai *features* temporais genéricas e transferíveis, enquanto o decoder precisa adaptar como estas *features* são combinadas e projetadas em predições futuras. Congelar o backbone protege o conhecimento universalmente útil adquirido no pré-treinamento, enquanto treinar o decoder permite especialização no domínio específico. Esta abordagem resulta em eficiência

computacional.

3.4.1 Divisão dos dados

O conjunto de dados foi particionado temporalmente, respeitando a ordem cronológica das observações, em três partes:

- **Treino**: 0% a 80% da série temporal;
- **Validação**: 80% a 90%;
- **Teste**: 90% a 100%.

Onde o conjunto de treino será utilizado no re-treinamento do TTM, o conjunto de validação será utilizado para monitorar o desempenho durante o treinamento, e o conjunto de teste será utilizado para avaliar o desempenho final do modelo.

3.4.2 Parâmetros de treinamento

O modelo foi configurado para utilizar um comprimento de contexto de 1024 passos, prevendo um horizonte de 96 passos à frente. O que é um dos padrões recomendados desse modelo.

O treinamento foi conduzido com um máximo de 50 épocas, tamanho de lote de 64 amostras e taxa de aprendizado inicial de aproximadamente 5.2×10^{-4}, obtida por meio de busca automática de hiperparâmetros.

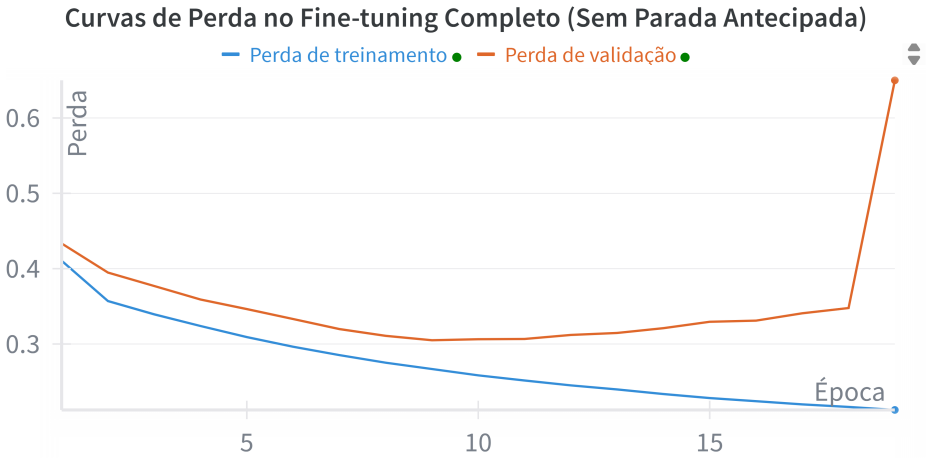
3.4.3 Mecanismos de controle

Para evitar sobreajuste, foram empregados dois mecanismos principais:

- **Early stopping**: interrompe o treinamento após 10 épocas consecutivas sem melhora da métrica de validação;
- **Monitoramento da perda**: o modelo selecionado ao final do processo corresponde à menor *eval_loss* obtida, assegurando o melhor desempenho no conjunto de validação.

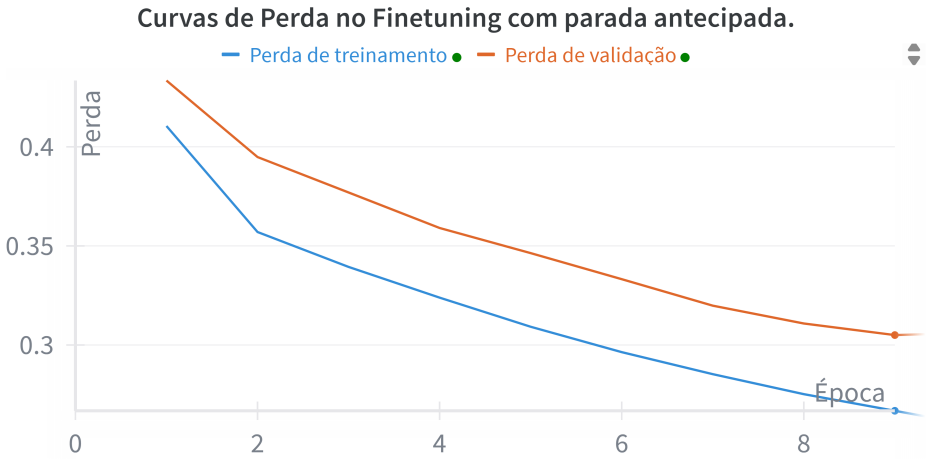
Na Figura 10, é mostrada a curva de perda ao longo de todo o processo de *fine-tuning*, incluindo as épocas que seriam executadas caso o early stopping não fosse aplicado. Essa visualização evidencia que o modelo tenderia ao sobreajuste (overfitting), já que a perda de validação começaria a piorar após determinado ponto.

Figura 10 – Curva de perda durante todo o processo de *fine-tuning* (antes do early stopping)



Fonte: Próprio autor

Figura 11 – Curva de perda do fine-tuning



Fonte: Próprio autor

Na Figura 11, é exibido apenas o trecho correspondente ao checkpoint selecionado pelo *early stopping*, ou seja, as 9 épocas efetivamente utilizadas no treinamento. Observa-se que, nesse intervalo, a perda de validação permaneceu menor que a perda de treino, indicando que o modelo foi treinado de forma adequada, graças à interrupção antecipada antes que o sobreajuste ocorresse.

O *fine-tuning* permitiu especializar o modelo em um novo conjunto de dados não visto no seu treinamento. A curva de validação mostra que, durante o treinamento do modelo, o desempenho foi melhorando até que o *early stopping* interrompesse o treinamento na época 9. Então, o modelo selecionado foi o modelo da época 9 que apresentou a menor perda de validação durante todo o processo.

3.5 CONSTRUÇÃO DA PLATAFORMA WEB

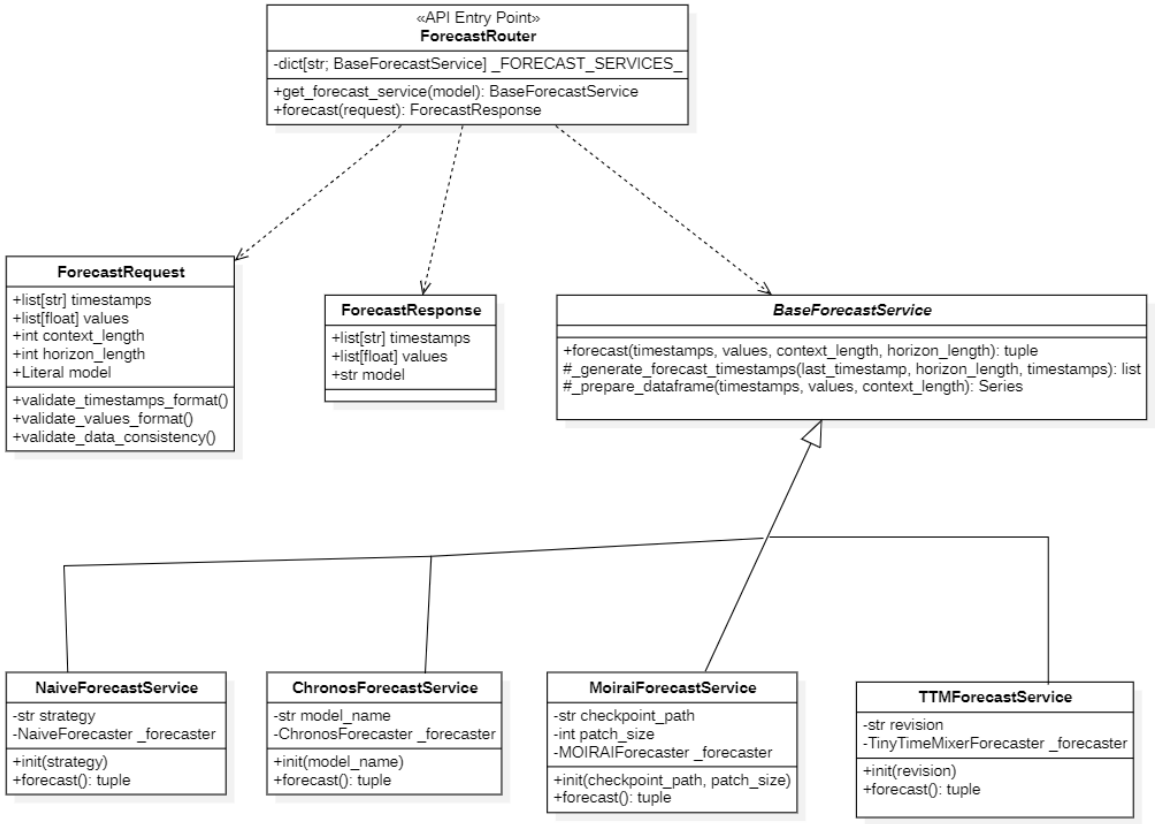
Com o objetivo de disponibilizar uma forma prática de utilizar os TSFM apresentados neste trabalho, foi desenvolvido o *TSFM Hub*, uma plataforma web que funciona como um *hub* de modelos de previsão de séries temporais. Essa plataforma permite que qualquer usuário realize previsões com diferentes TSFM utilizando seus próprios dados, sem a necessidade de escrever código ou treinar modelos. Ela é composta por um frontend implementado em Next.js, responsável pela interação com o usuário, e por um backend em Python desenvolvido com o framework FastAPI, que concentra toda a lógica de processamento e execução das previsões.

O backend é responsável pela lógica de previsão e utiliza a biblioteca sktime pois todos os TSFM selecionados estão presentes na biblioteca, facilitando a implementação e a utilização dos modelos. Na arquitetura do backend, cada previsor é representado por uma classe que herda de `BaseForecastService`, que define o comportamento comum entre todos os modelos TSFM. Isso torna a arquitetura flexível e fácil de expandir, bastando implementar uma nova classe que herde de `BaseForecastService` para cada novo modelo TSFM que for adicionado. A Figura 12 apresenta o diagrama de classes que organiza os principais componentes do backend. Essa estrutura centraliza toda a lógica de previsão em uma API simples, extensível e padronizada.

Os componentes visualizados na Figura 12 são:

- **ForecastRouter**: ponto de entrada da API, responsável por receber requisições de previsão e encaminhá-las ao serviço apropriado com base no modelo selecionado pelo usuário.
- **ForecastRequest**: estrutura que representa os dados enviados pelo usuário, contendo `timestamps`, `values`, o tamanho da janela de contexto (`context_length`), o horizonte de previsão (`horizon_length`) e o modelo escolhido. Essa classe também executa validações de formato e consistência.
- **ForecastResponse**: estrutura padronizada utilizada para devolver ao usuário os *timestamps* previstos, seus respectivos valores e o nome do modelo utilizado.
- **BaseForecastService**: classe abstrata que define o comportamento comum entre todos os modelos TSFM. Ela implementa métodos de preparação da série temporal, geração de novos *timestamps* e a interface principal de previsão.
- **Serviços específicos de modelo**: classes concretas responsáveis por executar a previsão utilizando cada modelo selecionado. Incluem:
  - `NaiveForecastService`

Figura 12 – Diagrama de classes UML da arquitetura do backend



Fonte: Próprio autor

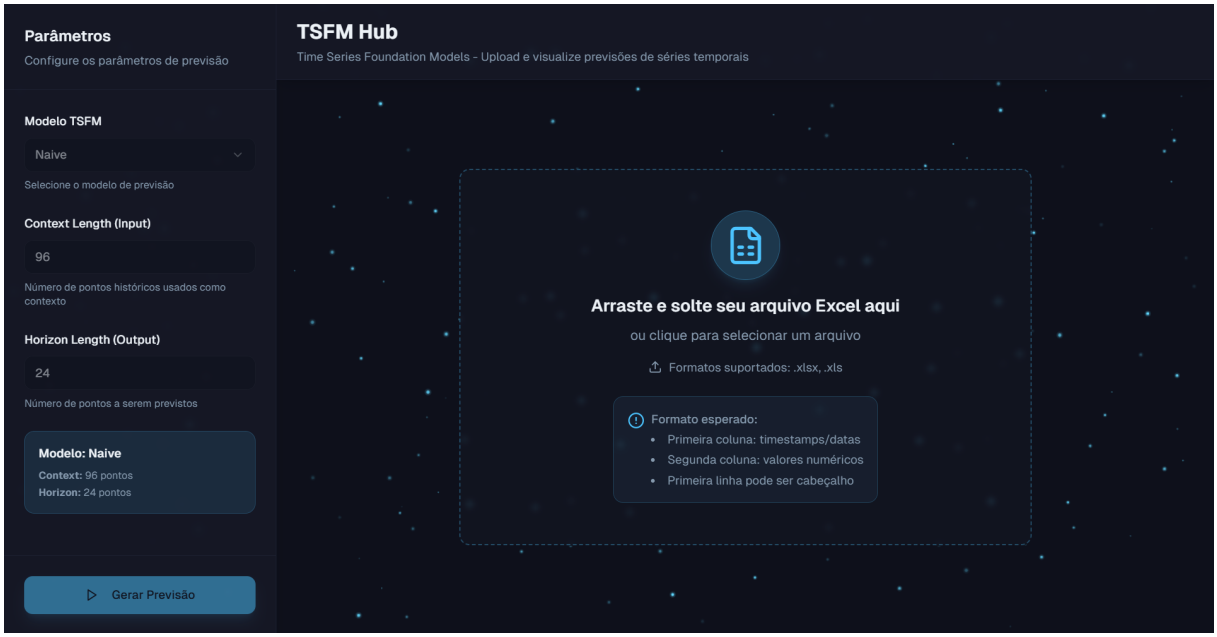
- `ChronosForecastService`
- `MoiraiForecastService`
- `TTMForecastService`

Dessa forma, a partir do modelo selecionado pelo usuário no frontend, o *ForecastRouter* interpreta a requisição e instancia o serviço correspondente, ele também garante que todos os modelos compartilhem a mesma interface e o mesmo fluxo de operação. Essa arquitetura permite adicionar novos TSFM ao *hub* sem modificar o código existente, bastando implementar uma nova classe que herde de `BaseForecastService`.

O frontend atua como camada de interação, permitindo que o usuário carregue os dados, selecione o modelo e ajuste parâmetros como a janela de contexto e o horizonte de previsão e visualize o resultado das previsões. A Figura 13 apresenta a interface visual da plataforma, onde são exibidos os parâmetros selecionados no painel lateral e o gráfico contendo a série temporal e a previsão gerada pelo modelo escolhido.

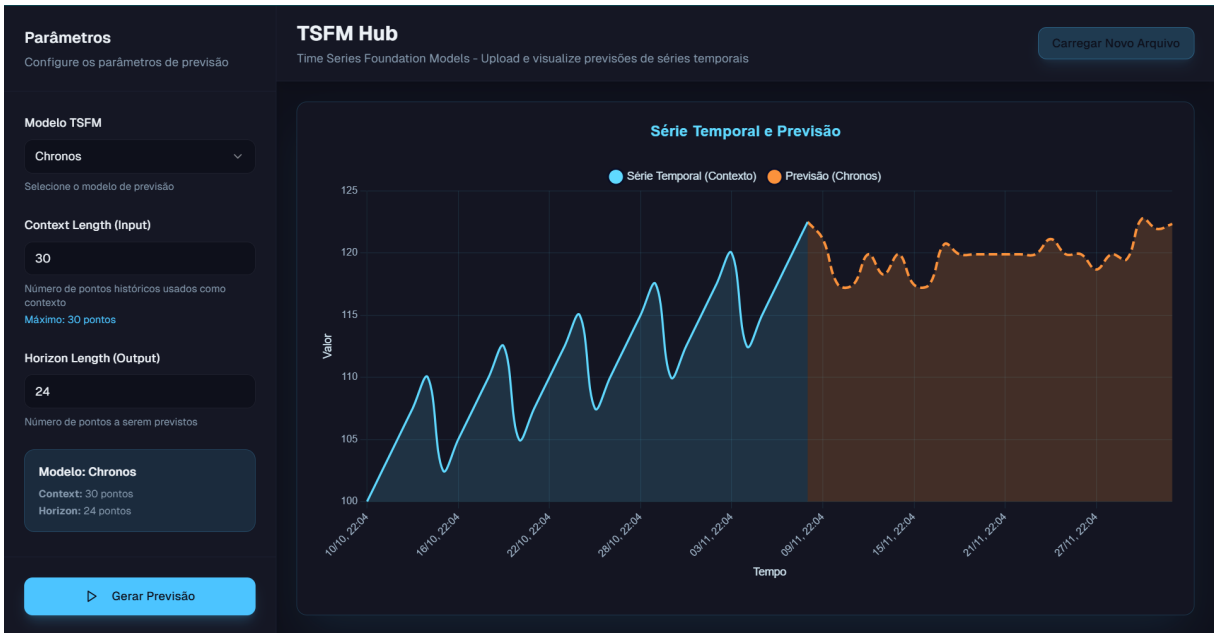
Após carregar o arquivo, escolher os parâmetros de previsão e clicar no botão de previsão, a plataforma irá gerar a previsão e exibir o resultado na tela, assim como

Figura 13 – Interface visual do TSFM Hub



Fonte: Próprio autor

Figura 14 – Exemplo de previsão gerada pelo TSFM Hub



Fonte: Próprio autor

mostra a Figura 14. Dessa forma, a plataforma web enfatiza a aplicabilidade dos TSFM, permitindo realizar previsões de forma rápida e acessível.

4 RESULTADOS OBTIDOS

Neste capítulo, são apresentados os resultados obtidos na avaliação dos modelos de TSFM, considerando duas abordagens: (i) previsões *zero-shot* e (ii) *fine-tuning*. As métricas e configurações utilizadas seguem aquelas descritas no Capítulo 3.

4.1 PREVISÕES ZERO-SHOT

4.1.1 Comparação geral entre modelos

A Tabela 2 apresenta o MAE médio por conjunto de dados e horizonte, comparando os modelos Chronos, Moirai, TTM e a *baseline* Naïve.

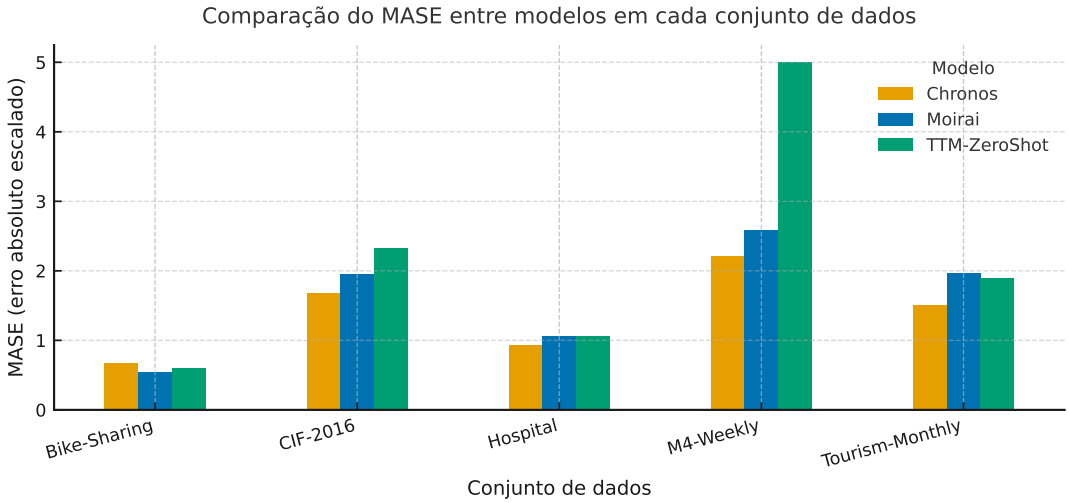
Observa-se que o modelo Chronos obteve o menor MAE médio na maioria dos cenários e horizontes, destacando-se especialmente em *Hospital*, *M4-Weekly* e *Tourism-Monthly*. O Moirai apresentou melhor desempenho apenas no *Bike-Sharing*, enquanto o Naïve manteve desempenho competitivo em algumas configurações (por exemplo, CIF-2016 em horizontes mais curtos). Esses resultados sugerem que o Chronos possui maior robustez entre domínios distintos.

Tabela 2 – MAE médio por conjunto de dados e horizonte (previsões zero-shot).

Conjunto de dados	H	**Chronos**	**Moirai**	**TTM**	**Naïve**
Bike-Sharing	96	13,01	9,30	10,19	14,05
CIF-2016	12	300 375,42	1 172 630,03	1 624 296,15	180 324,42
	24	238 757,83	2 291 739,03	1 722 877,29	214 886,57
	40	1 139 394,33	2 456 326,01	3 425 212,32	2 052 598,37
Hospital	12	13,16	14,48	19,07	14,17
	24	14,04	25,42	18,41	14,59
	40	20,82	22,21	26,40	18,82
M4-Weekly	12	380,06	459,94	744,35	459,49
	24	443,61	529,13	694,41	487,84
	40	474,92	558,95	749,49	504,27
Tourism-Monthly	12	4 409,42	6 299,59	5 735,30	7 185,13
	24	5 192,04	7 033,12	6 477,55	7 259,44
	40	7 318,01	7 234,23	9 740,24	7 658,26

Fonte: Próprio autor

Figura 15 – Comparação do MASE entre modelos em cada conjunto de dados.



Fonte: Próprio autor

4.1.2 Erro escalado entre datasets

A Figura 15 apresenta o MASE médio para cada modelo em todos os conjuntos de dados. O uso do MASE permite comparar desempenhos de forma justa entre séries com escalas distintas.

Nota-se que o **Chronos** obteve o menor MASE na maior parte dos conjuntos analisados, reforçando sua consistência geral. O **Moirai** apresentou vantagem no *Bike-Sharing*, um conjunto com alta granularidade temporal (dados horários), onde o modelo conseguiu capturar melhor variações de curto prazo. O **TTM** teve desempenho inferior em modo *zero-shot*, o que motiva a avaliação posterior do *fine-tuning*.

Embora nenhum dos objetivos deste trabalho tenha sido reproduzir os resultados divulgados nos principais benchmarks, os valores obtidos mostraram-se muito próximos aos relatados nas publicações originais dos modelos. Essa semelhança pôde ser observada especialmente nos casos do Chronos (ANSARI et al., 2024) e do Moirai (WOO et al., 2024), que utilizaram, em suas avaliações, os mesmos conjuntos de dados usados neste trabalho para as previsões *zero-shot*, com exceção do Bike Sharing. Ao comparar os resultados médios obtidos com as faixas reportadas nesses trabalhos, observou-se que os erros médios absolutos e escalados (MAE e MASE) permaneceram na mesma ordem de magnitude daqueles apresentados nas publicações originais. Algumas variações podem ser observadas dependendo das configurações de previsões utilizadas, visto diferenças de janela de contexto e horizonte de previsão podem variar o erro dependendo da natureza do conjunto de dados. Mas de forma geral, os resultados indicam que os modelos mantêm o mesmo padrão de desempenho descrito na literatura.

4.1.3 Detalhamento das métricas por conjunto de dados e horizonte

A Tabela 3 apresenta os resultados detalhados das previsões *zero-shot* para todas as métricas calculadas, em cada conjunto de dados e horizonte. A tabela detalhada confirma as conclusões anteriores sobre a métrica MAE, mostrando que o Chronos obteve o melhor desempenho em grande parte dos conjuntos e horizontes. Entretanto, é possível ver que todos os TSFM apresentaram resultados piores em horizontes de previsão mais longos, aumentando o erro e ficando mais próximos de serem superados pela baseline Naïve. É possível observar que a métrica MAPE não se comportou bem para o dataset *Bike-Sharing*, onde o erro foi muito alto, o que pode ser devido aos valores da série temporal serem muito baixos, o que faz com que o erro relativo seja muito alto.

Tabela 3 – Resultados detalhados das previsões zero-shot por conjunto de dados e horizonte.

Conjunto de dados	H	**Modelo**	**MAE**	**MSE**	**MAPE (%)**	**MASE**
Bike-Sharing	96	Chronos	13,01	345,97	77,55	0,673
		Moirai	9,30	157,36	79,67	0,544
		TTM	10,19	189,55	83,46	0,595
		Naïve	14,05	402,26	78,65	0,724
CIF-2016	12	Chronos	300 375,42	$4,36 \times 10^{12}$	18,49	1,284
		Moirai	1 172 630,03	$8,08 \times 10^{13}$	25,09	1,621
		TTM	1 624 296,15	$2,19 \times 10^{14}$	28,03	2,044
		Naïve	180 324,42	$1,89 \times 10^{12}$	17,84	1,407
CIF-2016	24	Chronos	238 757,83	$6,82 \times 10^{12}$	21,14	1,615
		Moirai	2 291 739,03	$4,99 \times 10^{14}$	25,52	1,878
		TTM	1 722 877,29	$3,34 \times 10^{14}$	23,91	2,144
		Naïve	214 886,57	$3,29 \times 10^{12}$	20,90	1,714
CIF-2016	40	Chronos	1 139 394,33	$2,48 \times 10^{14}$	17,50	2,136
		Moirai	2 456 326,01	$9,94 \times 10^{14}$	19,96	2,330
		TTM	3 425 212,32	$2,06 \times 10^{15}$	20,30	2,791
		Naïve	2 052 598,37	$7,13 \times 10^{14}$	17,70	2,200
Hospital	12	Chronos	13,16	845,41	22,68	0,865
		Moirai	14,48	1 046,35	26,76	0,955
		TTM	19,07	2 510,54	26,90	1,051
		Naïve	14,17	938,80	24,87	0,966
Hospital	24	Chronos	14,04	900,95	24,09	0,874
		Moirai	25,42	8 001,12	31,35	1,108

Continua na próxima página

Tabela 3 – Continuação da página anterior

Conjunto de dados	H	Modelo	MAE	MSE	MAPE (%)	MASE
		TTM	18,41	2 244,45	25,06	0,961
		Naïve	14,59	914,55	26,70	0,974
Hospital	40	Chronos	20,82	3 001,91	25,24	1,028
		Moirai	22,21	8 810,78	29,21	1,092
		TTM	26,40	5 044,87	25,94	1,152
		Naïve	18,82	1 800,82	29,65	1,126
M4-Weekly	12	Chronos	380,06	680 143,77	8,82	1,486
		Moirai	459,94	$1,08 \times 10^6$	11,40	1,846
		TTM	744,35	$1,51 \times 10^6$	15,69	5,636
		Naïve	459,49	988 386,69	10,60	1,615
M4-Weekly	24	Chronos	443,61	773 831,18	11,32	2,277
		Moirai	529,13	949 915,03	14,79	2,715
		TTM	694,41	$1,30 \times 10^6$	16,98	4,510
		Naïve	487,84	915 267,52	12,37	2,118
M4-Weekly	40	Chronos	474,92	$1,00 \times 10^6$	11,58	2,873
		Moirai	558,95	$1,09 \times 10^6$	17,48	3,163
		TTM	749,49	$1,67 \times 10^6$	18,12	4,845
		Naïve	504,27	$1,01 \times 10^6$	13,21	3,002
Tourism-Monthly	12	Chronos	4 409,42	$2,58 \times 10^8$	17,96	1,412
		Moirai	6 299,59	$4,22 \times 10^8$	26,00	2,026
		TTM	5 735,30	$3,36 \times 10^8$	23,31	1,874
		Naïve	7 185,13	$4,75 \times 10^8$	26,02	2,410
Tourism-Monthly	24	Chronos	5 192,04	$3,44 \times 10^8$	20,82	1,432
		Moirai	7 033,12	$7,85 \times 10^8$	29,12	1,862
		TTM	6 477,55	$5,01 \times 10^8$	25,74	1,782
		Naïve	7 259,44	$4,44 \times 10^8$	30,14	2,274
Tourism-Monthly	40	Chronos	7 318,01	$9,52 \times 10^8$	24,15	1,664
		Moirai	7 234,23	$5,80 \times 10^8$	36,47	2,018
		TTM	9 740,24	$1,94 \times 10^9$	32,65	2,029
		Naïve	7 658,26	$4,97 \times 10^8$	39,80	2,212

Fonte: Próprio autor

4.2 FINE-TUNING

No capítulo anterior, detalhou-se o *fine-tuning* que foi conduzido para o TTM no *Bike-Sharing dataset* (H=96). A Tabela 4 compara o TTM ajustado com as abordagens *zero-shot* e com o Naïve, de forma a avaliar a diferença de desempenho entre a abordagem *zero-shot* e a abordagem de fine-tuning. Nota-se que a métrica MAPE foi excluída pois em experimentos passados foi observado que não se comportou bem para o dataset *Bike-Sharing*, por conta da natureza dos dados.

Tabela 4 – Fine-tuning do TTM no *Bike-Sharing* (H=96). Melhores valores em negrito.

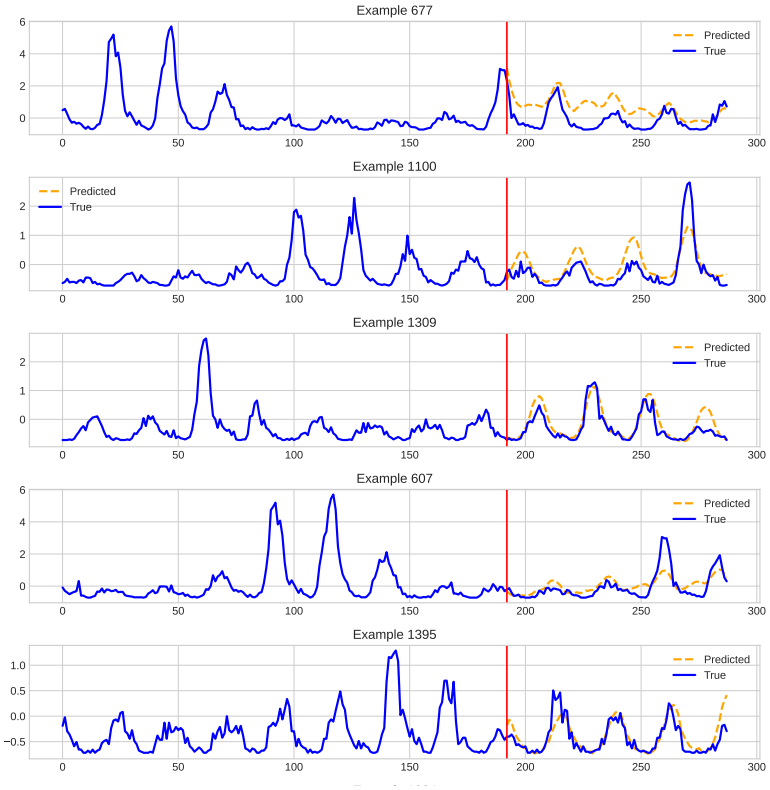
Modelo	**MAE**	**MSE**	**MASE**
Chronos	13.008	345.971	0.673
Moirai	9.299	157.359	0.544
Naive	14.049	402.257	0.724
TTM-ZeroShot	10.188	189.546	0.595
TTM-Finetuned	**5.783**	**59.746**	**0.364**

Fonte: Próprio autor

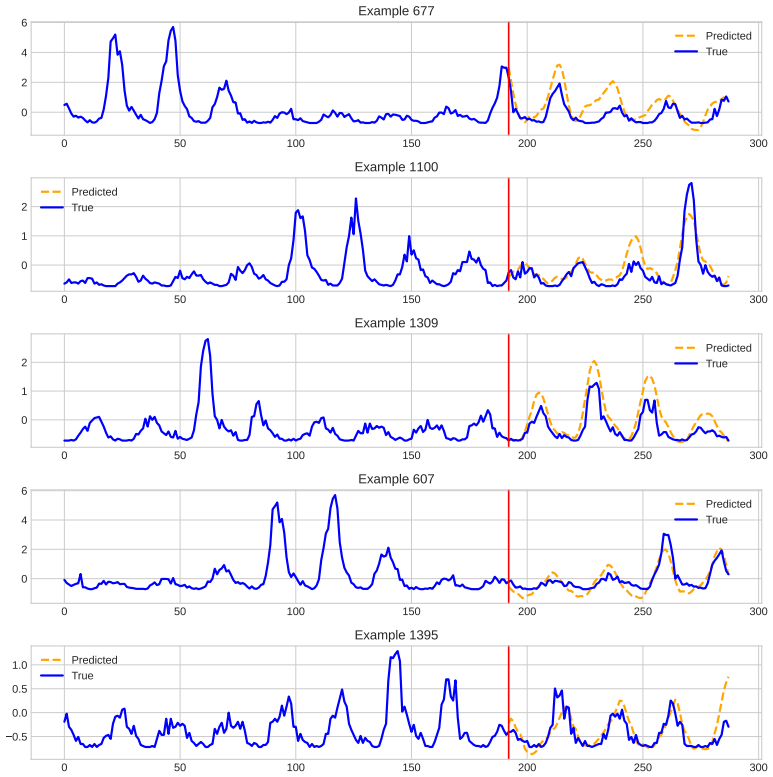
Depois do processo de *fine-tuning*, o modelo TTM apresentou resultados significativamente melhores em todas as métricas. Ele superou todos os outros modelos, reduzindo o MAE de 10.19 para 5.78 e o MSE de 189.55 para 59.75. Também teve uma redução substancial no MASE (de 0.595 para 0.364) confirmando que o modelo ajustado produziu previsões mais precisas em relação ao *baseline* sazonal.

A Figura 16 mostra a comparação entre as previsões em modo *zero-shot* e após o *fine-tuning* do modelo TTM. Observa-se que, após o *fine-tuning*, o modelo passa a capturar melhor a sazonalidade e a tendência das séries, reduzindo o erro absoluto na maior parte dos exemplos apresentados. Essas visualizações da Figura 16 foram geradas no mesmo ambiente em que ocorreu o processo de ajuste fino e, portanto, não correspondem exatamente às previsões utilizadas no cálculo das métricas apresentadas anteriormente. Para gerar diferentes amostras de séries temporais nesse ambiente, foi empregada uma janela deslizante sobre o conjunto de testes, o que difere da metodologia de avaliação usada nas etapas anteriores.

Figura 16 – Comparação entre previsões em modo *zero-shot* e após *fine-tuning* do modelo TTM.



(a) Previsões em modo *zero-shot*



(b) Previsões após *fine-tuning*

Fonte: Próprio autor.

5 CONCLUSÕES

O trabalho atingiu o objetivo principal de avaliar o desempenho de diferentes modelos fundacionais de séries temporais, permitindo comparar previsões em modo *zero-shot* e por meio de *fine-tuning*. Os resultados mostraram que o Chronos apresentou os melhores indicadores na maior parte dos testes, enquanto o Moirai se destacou em conjuntos de maior granularidade, capturando melhor variações rápidas. O TinyTimeMixer teve desempenho mais modesto no *zero-shot*, porém mostrou grande capacidade de adaptação após o *fine-tuning*, superando todos os demais modelos no conjunto utilizado. Esses resultados reforçam que, embora modelos pré-treinados tenham bom desempenho imediato, a etapa de ajuste fino pode elevar significativamente a precisão, especialmente em domínios específicos.

Além disso, foi possível verificar que os testes realizados ficaram próximos dos *benchmarks* atuais apresentados pelos autores dos modelos, o que confirma a consistência da metodologia adotada. Os objetivos específicos também foram atingidos, incluindo a análise experimental dos modelos, a comparação entre previsões *zero-shot* e *fine-tuning* e o desenvolvimento de uma plataforma web capaz de executar previsões de forma acessível. A plataforma demonstrou a aplicabilidade prática dos TSFM ao permitir que qualquer usuário carregue dados, selecione modelos e visualize previsões de maneira direta, evidenciando como essas ferramentas podem ser utilizadas fora do ambiente acadêmico.

Também foi observado que a abordagem *zero-shot* é notável pela capacidade de generalizar entre diferentes tipos de séries temporais ignorando a necessidade de aquisição de um conjunto de dados de treinamento, mas essa generalização apresenta limites conforme o domínio e o horizonte de previsão aumentam. Em cenários onde esses limites ficam evidentes, o *fine-tuning* continua sendo uma estratégia essencial para melhorar o desempenho e especializar o modelo. Dessa forma, o trabalho cumpriu seus objetivos, e além disso, também destacou o potencial e as limitações atuais dos TSFM, indicando caminhos claros para evoluções futuras.

5.1 TRABALHOS FUTUROS

Por ser tratar de um tema tão atual e com tantas possibilidades de aplicação, o trabalho pode ser expandido de várias formas. Uma primeira possibilidade é a adição de mais modelos para a previsão *zero-shot* e também no *TSFM Hub*, permitindo a comparação entre mais modelos, visto que esse trabalho selecionou modelos de menor custo computacional. Também é possível implementar métricas de avaliação no TSFM Hub, permitindo a comparação de modelos de forma eficiente. Além disso, é possível adicionar

outras funcionalidades ao *hub*, como a possibilidade de comparação da previsão feita com diferentes baselines no ato da previsão, de forma a visualizar a diferença de desempenho entre o modelo e o baseline no próprio gráfico gerado. Para encerrar, uma última possibilidade seria realizar o *fine-tuning* de diferentes modelos, para verificar a capacidade de adaptação de diferentes modelos a diferentes conjuntos de dados.

REFERÊNCIAS

AMAZON WEB SERVICES. **O que é RNN? — Explicação sobre redes neurais recorrentes — AWS**. pt-BR. Acesso em: 12 jun. 2025. 2025. Disponível em: <https://aws.amazon.com/pt/what-is/recurrent-neural-network/>. Acesso em: 12 jun. 2025.

ANSARI, Abdul Fatir et al. **Chronos: Learning the Language of Time Series**. 2024. arXiv: 2403.07815 [cs.LG]. Disponível em: <https://arxiv.org/abs/2403.07815>.

ATHANASOPOULOS, George et al. The tourism forecasting competition. **International Journal of Forecasting**, v. 27, n. 3, p. 822–844, 2011.

BENGIO, Yoshua; COURVILLE, Aaron; VINCENT, Pascal. **Representation Learning: A Review and New Perspectives**. 2014. arXiv: 1206.5538 [cs.LG]. Disponível em: <https://arxiv.org/abs/1206.5538>.

BERGMANN, Dave. **What is self-supervised learning?** 5 dez. 2023. Disponível em: <https://www.ibm.com/think/topics/self-supervised-learning>. Acesso em: 23 jun. 2025.

CHEN, Si-An et al. **TSMixer: An All-MLP Architecture for Time Series Forecasting**. 2023. arXiv: 2303.06053 [cs.LG]. Disponível em: <https://arxiv.org/abs/2303.06053>.

DAS, Abhimanyu et al. **A decoder-only foundation model for time-series forecasting**. 2024. arXiv: 2310.10688 [cs.CL]. Disponível em: <https://arxiv.org/abs/2310.10688>.

DEVOPEDIA. **Supervised vs Unsupervised Learning**. Version 33, January 12. Accessed 2025-06-08. 2022. Disponível em: <https://devopedia.org/supervised-vs-unsupervised-learning>.

EKAMBARAM, Vijay et al. **Tiny Time Mixers (TTMs): Fast Pre-trained Models for Enhanced Zero/Few-Shot Forecasting of Multivariate Time Series**. 2024. arXiv: 2401.03955 [cs.LG]. Disponível em: <https://arxiv.org/abs/2401.03955>.

FANAEE-T, Hadi. **Bike Sharing**. 2013. UCI Machine Learning Repository. Disponível em: <https://doi.org/10.24432/C5W894>.

GODAHEWA, Ruvinda et al. Monash Time Series Forecasting Archive. In: NEURIPS Datasets and Benchmarks Track. 2021. Disponível em: <https://forecastingdata.org/>.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. MIT Press, 2016. http://www.deeplearningbook.org.

HAYKIN, Simon. **Neural Networks: A Comprehensive Foundation**. 2nd. USA: Prentice Hall PTR, 1998. ISBN 0132733501.

HYNDMAN, Rob J. **expsmooth: Data Sets from Forecasting with Exponential Smoothing**. 2015. https://CRAN.R-project.org/package=expsmooth. R package version 2.3.

HYNDMAN, Rob J.; ATHANASOPOULOS, George. **Forecasting: Principles and Practice**. Melbourne, Australia: OTexts, 2018. Disponível em: https://otexts.com/fpp3/.

LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep learning. eng. **Nature**, v. 521, n. 7553, p. 436–444, mai. 2015. ISSN 1476-4687. DOI: 10.1038/nature14539.

LI, Zhe et al. **TSFM-Bench: A Comprehensive and Unified Benchmark of Foundation Models for Time Series Forecasting**. arXiv, jun. 2025. arXiv:2410.11802 [cs]. DOI: 10.48550/arXiv.2410.11802. Disponível em: <http://arxiv.org/abs/2410.11802>. Acesso em: 19 jun. 2025.

LIANG, Xiaoyao. Chapter 1 - Theoretical basis. In _____. **Ascend AI Processor Architecture and Programming**. Edição: Xiaoyao Liang. Elsevier, jan. 2020. P. 1–40. ISBN 978-0-12-823488-4. DOI: 10.1016/B978-0-12-823488-4.00001-1. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780128234884000011>. Acesso em: 9 jun. 2025.

LIANG, Yuxuan et al. Foundation Models for Time Series Analysis: A Tutorial and Survey. In: PROCEEDINGS of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, ago. 2024. (KDD '24), p. 6555–6565. DOI: 10.1145/3637528.3671451. Disponível em: <http://dx.doi.org/10.1145/3637528.3671451>.

MAKRIDAKIS, Spyros; SPILIOTIS, Evangelos; ASSIMAKOPOULOS, Vassilios. The M4 Competition: 100,000 time series and 61 forecasting methods. **International Journal of Forecasting**, v. 36, n. 1, p. 54–74, 2020. M4 Competition. ISSN 0169-2070. DOI: https://doi.org/10.1016/j.ijforecast.2019.04.014. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169207019301128>.

MELCHER, Kathrin. **A Friendly Introduction to Deep Neural Networks**. Blog post. KNIME. 2023. Disponível em: <https://www.knime.com/blog/a-friendly-introduction-to-deep-neural-networks>. Acesso em: 9 jun. 2025.

MONGODB. **Time Series Data Introduction**. en-US. Accessed 2025-06-08. MongoDB. 2025. Disponível em: <https://www.mongodb.com/resources/basics/time-series-data-analysis>.

MONTGOMERY, Douglas C.; JENNINGS, Cheryl L.; KULAHCI, Murat. **Introduction to Time Series Analysis and Forecasting**. 2. ed. Hoboken, NJ: John Wiley & Sons, 2015. P. 672. (Wiley Series in Probability and Statistics). ISBN 978-1-118-74511-3.

MURPHY, Kevin P. **Machine Learning: A Probabilistic Perspective**. The MIT Press, 2012. ISBN 0262018020.

REIS, Marcelo Menezes. **Análise de séries temporais**. 2023. https://www.inf.ufsc.br/~marcelo.menezes.reis/Cap4.pdf. Florianópolis, p. 55. Acesso em: 11 maio 2025.

SKTIME. **MeanAbsolutePercentageError**. 2025. https://www.sktime.net/en/latest/api_reference/auto_generated/sktime.performance_metrics.forecasting.MeanAbsolutePercentageError.html. Acesso em 21 de setembro de 2025.

_____. **MeanAbsoluteScaledError**. 2025. https://www.sktime.net/stable/api_reference/auto_generated/sktime.performance_metrics.forecasting.MeanAbsoluteScaledError.html. Acesso em 21 de setembro de 2025.

STEPNICKA, Miroslav; BURDA, Martin. On the results and observations of the time series forecasting competition CIF 2016. In: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). 2017. P. 1–6.

WOO, Gerald et al. **Unified Training of Universal Time Series Forecasting Transformers**. 2024. arXiv: 2402.02592 [cs.LG]. Disponível em: <https://arxiv.org/abs/2402.02592>.

ZHOU, Haoyi et al. **Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting**. arXiv, mar. 2021. arXiv:2012.07436 [cs]. DOI: 10.48550/arXiv.2012.07436. Disponível em: <http://arxiv.org/abs/2012.07436>. Acesso em: 18 jun. 2025.

APÊNDICE A – REPOSITÓRIO DA PLATAFORMA WEB DESENVOLVIDA

O link abaixo leva ao repositório da plataforma web desenvolvida:

https://github.com/gugasth/tsfm_hub