INSTITUTO FEDERAL DO ESPIRITO SANTO
SISTEMAS DE INFORMAÇÃO

LUIZ HENRIQUE CABRAL ZANELLA

ESTIMATIVA DA VIDA ÚTIL REMANESCENTE DE MOTORES AERONÁUTICOS UTILIZANDO REDES NEURAIS LSTM: UM ESTUDO COM O CONJUNTO DE DADOS C-MAPSS

Cachoeiro de Itapemirim
2025

LUIZ HENRIQUE CABRAL ZANELLA

ESTIMATIVA DA VIDA ÚTIL REMANESCENTE DE MOTORES AERONÁUTICOS UTILIZANDO REDES NEURAIS LSTM: UM ESTUDO COM O CONJUNTO DE DADOS C-MAPSS

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Dr. Rafael Vargas

Cachoeiro de Itapemirim

2025

(Biblioteca do Campus Cachoeiro de Itapemirim)

Z28e Zanella, Luiz Henrique Cabral.

Estimativa da vida útil remanescente de motores aeronáuticos utilizando redes neurais LSTM: um estudo com o conjunto de dados C-MAPSS / Luiz Henrique Cabral Zanella. - 2025.
89 f. : il. ; 30 cm..

Orientador: Rafael Vargas Mesquita dos Santos

TCC (Graduação) Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, Sistemas de Informação, 2025.

1. Inteligência artificial . 2. Redes neurais . 3. Aprendizado do computador. 4. Aviões - Motores - Manutenção e reparo. I. Santos, Rafael Vargas Mesquita dos. II.Título III. Instituto Federal do Espírito Santo.

CDD: 006.3

Bibliotecário/a: Renata Lorencini Rizzi CRB6-ES nº 085

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DO ESPÍRITO SANTO
CAI - COORDENADORIA DO CURSO DE BACHARELADO EM SISTEMAS DE INFORMACAO

FOLHA DE APROVAÇÃO-TCC Nº 8 / 2025 - CAI-CCSI (11.02.18.01.08.02.13)

Nº do Protocolo: 23151.003500/2025-31

Cachoeiro De Itapemirim-ES, 02 de dezembro de 2025.

LUIZ HENRIQUE CABRAL ZANELLA

ESTIMATIVA DA VIDA ÚTIL REMANESCENTE DE MOTORES AERONÁUTICOS UTILIZANDO REDES NEURAIS LSTM: UM ESTUDO COMO CONJUNTO DE DADOSC-MAPSS

Trabalho de Conclusão de Curso apresentado à Coordenadoria de Sistemas de Informação do Instituto Federal do Espírito Santo, como requisito parcial para obtenção de título de Bacharel em Sistemas de Informação.

Aprovado em 01 de dezembro de 2025

COMISSÃO EXAMINADORA

D.sc. Rafael Vargas Mesquita dos Santos
Instituto Federal do Espírito Santo
Orientador

D.sc. Susana Brunoro Costa de Oliveira
Instituto Federal do Espírito Santo

D.sc. Ricardo Maroquio Bernardo
Instituto Federal do Espírito Santo

(Assinado digitalmente em 02/12/2025 09:46)
RAFAEL VARGAS MESQUITA DOS SANTOS
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 1544937

(Assinado digitalmente em 03/12/2025 02:25)
RICARDO MAROQUIO BERNARDO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 2152606

(Assinado digitalmente em 03/12/2025 14:26)
SUSANA BRUNORO COSTA DE OLIVEIRA
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCTI (11.02.18.01.08.02.07)
Matrícula: 1505999

DECLARAÇÃO DO AUTOR

Declaro, para fins de pesquisa acadêmica, didática e técnico-científica, que este Trabalho de Conclusão de Curso pode ser parcialmente utilizado, desde que se faça referência à fonte e ao autor.

Cachoeiro de Itapemirim, 01 de Dezembro de 2025.

LUIZ HENRIQUE CABRAL ZANELLA

RESUMO

Os avanços tecnológicos têm impulsionado a evolução das estratégias de manutenção, ressaltando a importância de abordagens preditivas para garantir a confiabilidade e a eficiência operacional de ativos. Este trabalho apresenta uma solução baseada em redes neurais recorrentes do tipo Long Short-Term Memory (LSTM) aplicada ao prognóstico da vida útil remanescente, ou Remaining Useful Life (RUL), de turbinas aeronáuticas, utilizando o conjunto de dados Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) disponibilizado pelo Centro de Excelência em Prognóstico da NASA. O modelo final, avaliado sobre a última janela de cada motor, alcançou desempenho expressivo, com erro médio absoluto (MAE) de 9,18 ciclos e raiz do erro quadrático médio (RMSE) de 12,60 ciclos, resultados competitivos quando comparados aos principais trabalhos da literatura para o subconjunto FD001. Além disso, foi implementado um protótipo Web que possibilita o carregamento de novos dados, a execução automática do pré-processamento e a estimativa da RUL em tempo real, evidenciando a aplicabilidade prática e o caráter operacional da solução desenvolvida.

Palavras-chave: LSTM; Inteligência Artificial; Manutenção Preditiva; Vida Útil Remanescente; Aprendizado Profundo.

ABSTRACT

Technological advancements have driven the evolution of maintenance strategies, highlighting the importance of predictive approaches to ensure the reliability and operational efficiency of assets. This work presents a solution based on recurrent neural networks of the Long Short-Term Memory (LSTM) type, applied to the prognostics of the Remaining Useful Life (RUL) of aircraft turbofan engines. The model was developed and evaluated using the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset provided by NASA's Prognostics Center of Excellence. When evaluated on the last time window of each engine, the final model achieved expressive performance, with a Mean Absolute Error (MAE) of 9.18 cycles and a Root Mean Squared Error (RMSE) of 12.60 cycles—results that are competitive when compared with leading studies in the literature for the FD001 subset. In addition, a Web prototype was implemented, enabling the upload of new data, the automatic execution of preprocessing steps, and real-time RUL estimation, demonstrating the practical applicability and operational relevance of the proposed solution.

Keywords: LSTM; Artificial Intelligence; Predictive Maintenance; Remaining Useful Life; Deep Learning.

LISTA DE ALGORITMOS

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

C-MAPSS	Commercial Modular Aero-Propulsion System Simulation
GRU	Gated Recurrent Unit
IA	Inteligência Artificial
L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NASA	National Aeronautics and Space Administration
PHM	Prognostics and Health Management
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RUL	Remaining Useful Life
RMSE	Root Mean Squared Error

SUMÁRIO

1 INTRODUÇÃO

A manutenção de sistemas complexos tem se tornado cada vez mais estratégica em diversos setores, pois falhas inesperadas acarretam prejuízos operacionais, riscos à segurança e interrupções críticas nos serviços. Nesse contexto, a manutenção preditiva surge como uma abordagem promissora, permitindo antecipar falhas com base em dados reais de operação. Estudos recentes destacam que essa estratégia pode aumentar a confiabilidade, reduzir custos e aprimorar a eficiência geral dos sistemas monitorados, consolidando-se como uma prática essencial em diferentes domínios de aplicação (BENHANIFIA et al., 2025).

Contudo, a evolução dessas estratégias encontra seu maior desafio na modelagem precisa da Vida Útil Remanescente (RUL) de componentes críticos. Na aviação, a falha não prevista de uma turbina acarreta riscos inaceitáveis à segurança e custos operacionais proibitivos associados à indisponibilidade não planejada da aeronave. Diante da alta dimensionalidade e do ruído inerente aos dados sensoriais de motores modernos, os métodos estatísticos tradicionais mostram-se limitados. Portanto, a investigação de abordagens baseadas em Aprendizado Profundo (*Deep Learning*) justifica-se como um imperativo técnico: é necessário utilizar arquiteturas capazes de capturar dependências temporais complexas e não lineares para garantir a confiabilidade exigida neste cenário de alto risco.

As abordagens tradicionais de manutenção, como a corretiva e a preventiva baseada em calendário, enfrentam limitações significativas. Enquanto a manutenção corretiva depende da ocorrência da falha, a preventiva pode levar à substituição desnecessária de componentes ainda funcionais. Nesse contexto, a manutenção preditiva surge como uma alternativa mais eficaz, permitindo intervenções baseadas em dados e condições reais dos ativos, o que contribui para maior precisão nas decisões e redução de custos operacionais (ELSHERIF et al., 2025).

Manutenções planejadas, assertivas e fundamentadas em dados operacionais contribuem diretamente para a competitividade organizacional, promovendo maior confiabili-

dade dos ativos, redução de custos e melhoria do desempenho frente à concorrência (EKANEM; USORO; BARIDAM, 2022).

O avanço das tecnologias de monitoramento e a crescente disponibilidade de dados operacionais têm viabilizado a aplicação de modelos de prognóstico baseados em inteligência artificial. Técnicas de aprendizado profundo, em especial, destacam-se pela capacidade de identificar padrões complexos de degradação ao longo do tempo, permitindo estimativas mais precisas da vida útil remanescente de sistemas físicos diversos (WANG et al., 2025).

Entre as abordagens de aprendizado profundo para modelagem de séries temporais, as redes Long Short-Term Memory (LSTM) se destacam por sua arquitetura especializada que permite preservar informações relevantes por longos períodos de tempo. Essa capacidade de capturar dependências de longo prazo é especialmente útil em contextos onde padrões temporais distantes influenciam as previsões futuras. Essa característica faz com que sejam amplamente utilizadas na previsão da vida útil remanescente de motores aeronáuticos e outros sistemas industriais (HOCHREITER; SCHMIDHUBER, 1997).

Devido à escassez de dados reais de falha (*run-to-failure*) na aviação, decorrente de rigorosas restrições de segurança, adotou-se para este trabalho o conjunto de dados disponibilizado no repositório de prognóstico da National Aeronautics and Space Administration (NASA). A escolha deste *dataset* justifica-se por seu status de *benchmark* global, oferecendo simulações de alta fidelidade que permitem validar a eficácia do modelo proposto em comparação direta com o estado da arte. Nele constam os resultados obtidos por meio do *software* de simulação Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) (FREDERICK; DECASTRO; LITT, 2007). O conjunto contém registros de utilização de diversas *turbofans*, configurações operacionais e medições de sensores que em grande parte estão contaminados com ruído. Os registros não incluem informações como desgaste inicial e nem mesmo variação de fabricação, simulando a incerteza de cenários reais (CHAO et al., 2021).

Diante desse cenário, este trabalho tem como objetivo desenvolver e avaliar um modelo

de rede neural do tipo LSTM para a previsão da vida útil remanescente em motores aeronáuticos simulados no conjunto de dados C-MAPSS. A proposta concentra-se em investigar o impacto de diferentes estratégias de pré-processamento, como normalização e engenharia de atributos, bem como da variação de hiperparâmetros de treinamento sobre o desempenho da rede. Estudos recentes, como os de Chaoub et al. (2021), Liu, Wen e Wang (2025), reforçam a relevância dessas práticas no desenvolvimento de modelos robustos de prognóstico de falhas baseados em aprendizado profundo.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

Desenvolver e avaliar um modelo preditivo baseado em redes neurais recorrentes do tipo LSTM para estimar a RUL de motores aeronáuticos simulados, utilizando o subconjunto FD001 do conjunto de dados C-MAPSS da NASA como estudo de caso.

1.1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste trabalho são:

a) Projetar, treinar e avaliar um modelo LSTM para previsão da RUL com base nas séries temporais multivariadas do subconjunto FD001;

b) Utilizar os dados de degradação simulada de motores aeronáuticos, disponibilizados pelo centro de prognóstico da NASA (PHM Society), como base experimental;

c) Investigar o impacto de diferentes estratégias de pré-processamento, como normalização e engenharia de atributos, sobre o desempenho do modelo;

d) Analisar como a variação de hiperparâmetros de treinamento influencia as métricas de erro (MAE e RMSE);

e) Comparar os resultados obtidos com os de modelos de prognóstico previamente publicados na literatura;

f) Disponibilizar uma interface web funcional para submissão de novos dados do FD001 e retorno das estimativas de RUL.

1.2 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em cinco capítulos, organizados de modo a conduzir o leitor desde os fundamentos teóricos até a aplicação prática e análise dos resultados. A organização segue a seguinte disposição:

Capítulo 2 – Referencial Teórico: Apresenta a fundamentação necessária para o entendimento da pesquisa, abordando os conceitos de manutenção preditiva, prognóstico de falhas e estimativa da Vida Útil Remanescente. Também detalha as características do conjunto de dados C-MAPSS e revisa os fundamentos de Inteligência Artificial, com ênfase em Aprendizado Profundo e nas arquiteturas de Redes Neurais Recorrentes.

Capítulo 3 – Metodologia: Descreve o percurso experimental adotado, detalhando as etapas de pré-processamento, engenharia de atributos e normalização dos dados. Apresenta as ferramentas utilizadas, a definição das arquiteturas dos modelos (MLP e LSTM), as estratégias de busca de hiperparâmetros e os critérios para avaliação de desempenho.

Capítulo 4 – Resultados e Discussões: Exibe e analisa os resultados obtidos, demonstrando a evolução do desempenho desde o modelo baseline até a arquitetura final otimizada. Inclui a comparação (*benchmarking*) com trabalhos correlatos da literatura e apresenta o protótipo Web desenvolvido para disponibilização do modelo.

Capítulo 5 – Conclusão: Sintetiza os principais achados do estudo, retomando os objetivos propostos. Destaca as contribuições da pesquisa para a área de Prognostics and Health Management (PHM) e sugere perspectivas para trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 MANUTENÇÃO PREDITIVA

As máquinas e equipamentos possuem vida útil limitada e estão sujeitos a processos naturais de desgaste decorrentes das condições de operação. A degradação pode manifestar-se de diversas formas, incluindo alterações físicas observáveis, queda de desempenho, variações na qualidade do produto e aumento de emissões. Nesse contexto, a manutenção desempenha papel fundamental ao assegurar que esses desgastes sejam identificados, monitorados e tratados de forma a preservar o funcionamento adequado dos ativos.

De acordo com a ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (1994, p. 6), manutenção é a "combinação de todas as ações técnicas e administrativas, incluindo as de supervisão, destinadas a manter ou recolocar um item em um estado no qual possa desempenhar uma função requerida". Essa definição evidencia que a manutenção não se limita a ações corretivas, mas envolve um conjunto estruturado de práticas que visam garantir a confiabilidade e a disponibilidade dos ativos ao longo do seu ciclo de vida.

O bom funcionamento de máquinas, equipamentos e demais componentes industriais depende da aplicação de procedimentos técnicos específicos e de um conjunto estruturado de práticas preventivas e corretivas. Esses cuidados, que abrangem inspeção, ajuste, substituição e monitoramento, caracterizam o escopo das atividades de manutenção (ALMEIDA, 2018).

Desde a Primeira Revolução Industrial, as práticas de manutenção evoluíram significativamente em resposta aos avanços tecnológicos e ao aumento da complexidade dos sistemas produtivos. A mecanização inicial, seguida pela automação e pela digitalização, impulsionou o desenvolvimento de métodos mais precisos de inspeção, controle e intervenção. Nos últimos anos, esse progresso intensificou-se com o uso de sensores, sistemas informatizados e ferramentas de monitoramento contínuo, permitindo técnicas

de manutenção cada vez mais eficientes e adaptadas às necessidades específicas de cada indústria.

Nesse contexto, as metodologias de manutenção modernas buscam antecipar falhas e atuar de forma proativa, substituindo componentes antes que comprometam o processo produtivo. Apenas corrigir falhas não é suficiente; é necessário identificar suas causas fundamentais e implementar ações que reduzam a recorrência de intervenções não planejadas. Essa mudança de paradigma consolidou a transição de abordagens reativas para estratégias orientadas à previsão e prevenção (XENOS, 1998).

A manutenção preditiva consiste em monitorar continuamente o estado de funcionamento dos equipamentos por meio de medições periódicas ou em tempo real. Alterações em variáveis como vibração, temperatura ou ruído indicam o início de processos de desgaste e permitem identificar a necessidade de intervenção antes que ocorram falhas funcionais (ALMEIDA, 2018). Ao interpretar essas mudanças de comportamento, torna-se possível antecipar a degradação e planejar ações corretivas com maior precisão, reduzindo riscos e paradas não programadas.

Segundo Kardec e Nascif (2009, p. 45), “a manutenção preditiva estima as condições dos equipamentos, e quando a intervenção é decidida, o que se faz, na realidade, é uma manutenção corretiva planejada”. Esse entendimento reforça que o objetivo da abordagem preditiva não é eliminar totalmente as intervenções corretivas, mas transformá-las em ações programadas, realizadas no momento mais conveniente e com menor impacto na operação. Dessa forma, a organização reduz custos associados a paradas inesperadas e melhora o planejamento de recursos.

Na prática, a análise da evolução de um defeito envolve verificar se o nível de degradação permanece dentro de limites aceitáveis ou se já requer intervenção. A decisão deve considerar o momento mais econômico e operacionalmente viável para realizar a manutenção, de forma a maximizar a vida útil do equipamento sem comprometer sua segurança ou desempenho. Essa abordagem apoia-se no monitoramento contínuo de indicadores de saúde, permitindo que a intervenção ocorra antes que a falha se manifeste de forma crítica.

Nos últimos anos, a adoção de sensores inteligentes e de tecnologias associadas à Indústria 4.0 ampliou significativamente o alcance da manutenção preditiva. O monitoramento contínuo de variáveis como vibração, temperatura e pressão, aliado à capacidade de armazenamento em larga escala, possibilitou análises mais detalhadas do comportamento dos ativos. Essa evolução tecnológica marcou a transição das inspeções predominantemente visuais para abordagens orientadas a dados, criando as condições necessárias para a aplicação de técnicas de aprendizado de máquina (*Machine Learning*) e inteligência artificial no prognóstico de falhas (VASILACHE et al., 2024).

A manutenção preditiva está diretamente associada à estimativa da RUL dos ativos, métrica essencial para definir o momento ideal de intervenção e equilibrar confiabilidade, disponibilidade e custos operacionais. Dentro do escopo do Prognostics and Health Management, essa estimativa depende de técnicas de prognóstico capazes de antecipar o tempo restante até a falha. O prognóstico de falhas, discutido na próxima seção, constitui o mecanismo que viabiliza essa previsão.

Estudos clássicos já apontavam que práticas de manutenção bem estruturadas contribuem diretamente para a competitividade organizacional (PASCHOAL et al., 2009). Revisões recentes da literatura indicam que a manutenção preditiva deixou de ser uma prática complementar e passou a ocupar um papel estratégico nos ambientes industriais modernos (BENHANIFIA et al., 2025). Essa consolidação se torna ainda mais evidente quando a manutenção preditiva é associada a técnicas de aprendizado profundo aplicadas ao prognóstico de falhas, tendência destacada em trabalhos recentes (BENHANIFIA et al., 2025; ELSHERIF et al., 2025; WANG et al., 2025).

2.2 VIDA ÚTIL REMANESCENTE

A estimativa da vida útil remanescente é um dos elementos centrais da manutenção baseada na condição (LEE et al., 2006). A RUL representa o tempo restante para que um ativo continue operando dentro de suas especificações de projeto, considerando o seu estado atual de operação. Como esse tempo até a falha é, por natureza, desconhecido, sua estimativa depende diretamente das informações coletadas por sensores e dos registros de monitoramento disponíveis (SI et al., 2011).

A estimativa da RUL é tratada majoritariamente como um problema de regressão, no qual se busca prever um valor contínuo (o número de ciclos) a partir de dados históricos de degradação. Essa abordagem se diferencia dos problemas de classificação, que predizem uma categoria discreta (ex: 'Estado de Alerta' ou 'Falha Iminente'). A regressão oferece uma previsão mais detalhada e granular do tempo de falha, sendo por isso a metodologia central deste trabalho.

Ao iniciar sua operação, um equipamento encontra-se em sua condição de máxima vida útil. À medida que ciclos de uso se acumulam, processos naturais de desgaste reduzem gradualmente sua capacidade de funcionamento. A estimativa precisa da RUL torna-se, portanto, essencial para a tomada de decisão em manutenção, pois permite planejar intervenções de forma a equilibrar custo, segurança e disponibilidade operacional. Esse aspecto é especialmente crítico em setores de alto risco, como aviação, energia nuclear e operações offshore, nos quais falhas podem resultar em prejuízos significativos e impactos severos à segurança.

A norma ISO13381-1 (2015) descreve o prognóstico como o processo de estimar o tempo restante até a ocorrência de falhas, bem como o risco associado a modos de falha presentes ou potenciais. Essa concepção está diretamente alinhada ao conceito de vida útil remanescente, que expressa precisamente essa previsão temporal. Dessa forma, a RUL torna-se um elemento fundamental nas estratégias de manutenção baseadas na condição, integrando-se ao arcabouço do PHM.

Pesquisas recentes reforçam a relevância da estimativa da RUL no contexto do prognóstico. Liu, Wen e Wang (2025) apresentam uma revisão abrangente dos métodos atuais de previsão da vida útil remanescente, destacando o papel cada vez mais significativo das técnicas de aprendizado profundo na modelagem de séries temporais e na captura de padrões complexos de degradação. A síntese apresentada pelos autores confirma que a RUL permanece como um dos indicadores centrais nos sistemas modernos de prognóstico.

A estimativa da RUL pode ser formalmente tratada como um problema de regressão sobre séries temporais multivariadas, em que uma sequência histórica de medições

sensoriais e condições operacionais é utilizada para inferir o tempo restante até a falha (SI et al., 2011; LIU; WEN; WANG, 2025). A dependência temporal inerente a esses dados exige o uso de modelos capazes de capturar relações dinâmicas de curto e longo prazo, o que motiva a adoção de arquiteturas específicas para modelagem de sequências, como redes recorrentes e variantes baseadas em LSTM (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2.1 LIMITAÇÃO SUPERIOR DA RUL (RUL CAP)

Um procedimento recorrente na literatura é a aplicação de um limite superior (*capping*) à RUL, com o objetivo de reduzir a influência dos valores muito altos observados no início do ciclo de vida dos equipamentos. Esse tratamento torna o problema de regressão mais estável, direcionando o aprendizado do modelo para a fase em que a degradação efetivamente ocorre. No caso do conjunto FD001 do C-MAPSS, é comum a adoção de limites na faixa de 120 – 130 ciclos; neste trabalho, utilizou-se um valor de 125 ciclos, em linha com estudos recentes (PENG et al., 2022; {de Oliveira da Costa} et al., 2019).

2.2.2 SELEÇÃO E TRATAMENTO DE SENSORES

Além da formulação da RUL, a escolha das variáveis de entrada constitui uma etapa crítica no pré-processamento de dados. O subconjunto FD001 do C-MAPSS fornece medições de 21 sensores, mas estudos prévios destacam que nem todas essas variáveis contribuem significativamente para o prognóstico. Wang et al. (2023), Mitici et al. (2023) recomendam a exclusão de sensores com valores constantes ou baixa variabilidade, bem como de variáveis consideradas redundantes ou pouco informativas. Especificamente, a configuração de operação setting_3 e os sensores s1, s5, s10, s16, s18 e s19 foram removidos nesses trabalhos por não apresentarem correlação relevante com os estados de degradação das turbinas, contribuindo para modelos mais robustos e com menor risco de sobreajuste.

Neste trabalho, adotou-se a remoção automática dos sensores degenerados — isto é, aqueles que apresentavam valores constantes ao longo de todo o ciclo — como etapa de pré-processamento. Essa prática, amplamente empregada em estudos baseados no conjunto C-MAPSS, não configura uma técnica formal de seleção de atributos,

mas contribui para a redução de ruído, melhoria da qualidade das entradas e maior desempenho dos modelos de prognóstico. Dessa forma, nos modelos LSTM aqui desenvolvidos, foram desconsideradas as mesmas variáveis destacadas na literatura.

Por fim, é importante destacar que a estimativa da RUL está diretamente associada às técnicas de prognóstico de falhas, responsáveis por modelar a evolução da degradação ao longo do tempo. Essas técnicas serão detalhadas na seção seguinte.

2.3 PROGNÓSTICO DE FALHAS

O prognóstico de falhas é entendido como a capacidade de prever falhas antes de sua manifestação e estimar o tempo de vida útil restante de um equipamento (KIM; CHOI; AN, 2017). A ISO13381-1 (2015) define o prognóstico como o processo de estimar o tempo restante até a ocorrência de falhas, juntamente com a avaliação do risco associado a modos de falha atuais ou potenciais. Muitas outras definições foram propostas por diversos pesquisadores (SIKORSKA; HODKIEWICZ; MA, 2011).

Com suas raízes na indústria aeroespacial, o prognóstico de falhas permite avaliar um equipamento em tempo real levando em consideração as condições operacionais reais tornando possível estimar um estado futuro baseado em dados. Sendo a etapa primordial para o funcionamento de uma política de manutenção preditiva, o prognóstico prevê quanto tempo levará até a falha sob a atual condição operacional. Em resumo, prevê o momento em que um sistema ou componente deixará de funcionar como esperado, proporcionando aos usuários a oportunidade de reduzir riscos no nível do sistema e, ao mesmo tempo, prolongar sua vida útil (KIM; CHOI; AN, 2017).

A capacidade de prever quando um equipamento se aproxima do colapso é de suma importância para os mais diversos setores. Empresas de aviação, usinas nucleares, plataformas de petróleo são exemplos de atividades que possuem uma margem muito pequena de erro, quaisquer falhas em ativos podem ceifar a vida de muitos seres humanos. Normalmente, falhas em ativos são acompanhadas de grandes prejuízos, um mau funcionamento em apenas um único equipamento é capaz de afetar toda uma linha de montagem. O prognóstico de falhas fornece a possibilidade de se antecipar a uma falha, evitando o pior cenário que seria a parada de um equipamento que acarreta

nos pontos abordados anteriormente. No universo acadêmico, esse tipo de prognóstico é denominado vida útil remanescente, ou Remaining Useful Life (RUL) (JARDINE; LIN; BANJEVIC, 2006).

Diferentes abordagens têm sido propostas ao longo dos anos para viabilizar o prognóstico de falhas, variando de acordo com a disponibilidade de dados, a complexidade do sistema e o conhecimento técnico existente. De forma geral, a literatura classifica essas abordagens em três grandes categorias: baseada em conhecimento, baseada em modelos físicos e baseada em dados, cada uma com características próprias (SIKORSKA; HODKIEWICZ; MA, 2011).

2.3.1 PROGNÓSTICO DE FALHAS BASEADO EM CONHECIMENTO

Na literatura podemos observar a categorização das abordagens de prognóstico mais utilizadas. Correlacionando as experiências obtidas através da análise de situações passadas usando como base medição de sensores e as falhas ocorridas, a abordagem baseada em conhecimento consiste em regras predefinidas desenvolvidas utilizando o conhecimento obtido com as experiências. Podemos observar a implementação dessa técnica nos chamados “Sistemas especialistas”, que são sistemas compostos por regras SE e SENÃO, desenvolvidas a partir do conhecimento de um perito da área em questão (SIMEÓN, 2015).

2.3.2 PROGNÓSTICO DE FALHAS BASEADO EM MODELO

Adotando modelos matemáticos elaborados especificamente para o equipamento em questão, as abordagens baseadas em modelos físicos apresentam uma precisão superior pois incorporam descrições estruturais fundamentais do equipamento monitorado, sendo, muitas vezes, construídos considerando a base de conhecimento de cada elemento que compõem o todo. Segundo Kim, Choi e An (2017), a fundamentação para esse modelo é a suposição da existência de um modelo físico que descreve a evolução da degradação. Sendo essa suposição verdadeira, o comportamento futuro dos danos pode ser determinado pelo progresso do modelo de degradação no tempo futuro. Contudo, à medida em que o sistema se torna mais complexo, torna-se mais custoso e improvável prever todos os cenários e modos de operação para estimar a degradação.

2.3.3 PROGNÓSTICO DE FALHAS BASEADO EM DADOS

O prognóstico de falhas baseado em dados, também chamado de abordagem *data-driven*, utiliza informações coletadas por sensores e históricos de operação para identificar padrões de degradação e RUL. Diferente das abordagens baseadas em conhecimento ou em modelos físicos, esse método não depende de regras pré-definidas nem da formulação matemática dos mecanismos de falha, mas sim da análise direta dos dados de monitoramento (SIKORSKA; HODKIEWICZ; MA, 2011; SI et al., 2011).

Segundo Jardine, Lin e Banjevic (2006), essa abordagem tem se mostrado particularmente eficaz em sistemas complexos, onde a modelagem física seria inviável ou excessivamente custosa. Com o aumento da capacidade de armazenamento e processamento de dados, aliado ao avanço das técnicas de aprendizado de máquina, tornou-se possível aplicar modelos estatísticos, redes neurais artificiais, máquinas de vetor de suporte e, mais recentemente, arquiteturas de aprendizado profundo (ZHANG et al., 2019).

Pesquisas recentes demonstram que o uso de redes recorrentes, em especial as do tipo LSTM, tem proporcionado ganhos significativos na previsão da RUL, por sua capacidade de lidar com dependências temporais de longo prazo em séries temporais (ZHENG et al., 2017). Apesar dessas vantagens, os métodos baseados em dados apresentam como limitação a necessidade de grandes quantidades de dados históricos de qualidade para treinamento, além do desafio da interpretabilidade dos modelos de aprendizado profundo (NOR; PEDAPAIT; MUHAMMAD, 2021).

2.4 CONJUNTO DE DADOS C-MAPSS

O conjunto Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), desenvolvido pelo laboratório NASA Ames Research Center, tornou-se um padrão de referência para estudos de prognóstico de falhas e estimativa da RUL em motores aeronáuticos. Esse conjunto de dados simula trajetórias de degradação de turbinas a partir de diferentes perfis operacionais e modos de falha, fornecendo medições sensoriais e configurações de operação ao longo do tempo até o colapso do equipamento (SAXENA et al., 2008).

Cada instância da base representa um motor distinto operando até o momento da falha, com séries temporais multivariadas que descrevem as condições operacionais, configurações de uso e valores de sensores. O C-MAPSS é amplamente utilizado na literatura por refletir de forma realista os desafios da manutenção preditiva baseada em sensores, servindo como base para testes comparativos de diferentes algoritmos de aprendizado de máquina aplicados ao prognóstico de RUL.

2.5 INTELIGÊNCIA ARTIFICIAL

A Inteligência Artificial (IA) é um campo da ciência da computação voltado para a automação do comportamento inteligente. Segundo Luger (2013), a IA pode ser entendida como um campo da computação dedicado ao desenvolvimento de sistemas capazes de reproduzir comportamentos considerados inteligentes. Essa definição reflete a visão contemporânea da IA como disciplina aplicada à criação de sistemas capazes de executar tarefas que tradicionalmente exigiriam inteligência humana.

Russell et al. (2010), por sua vez, propõem que a IA pode ser compreendida a partir de duas dimensões: aquelas relacionadas a processos mentais e raciocínio, e aquelas relacionadas ao comportamento. Essa distinção permite organizar os diferentes enfoques adotados ao longo do desenvolvimento histórico da área.

Entre as definições clássicas, destacam-se quatro perspectivas principais:

a) Sistemas que pensam como seres humanos: entendem a IA como o esforço de modelar processos mentais humanos em máquinas, buscando reproduzir o raciocínio humano (HAUGELAND, 1985);

b) Sistemas que pensam racionalmente: concebem a inteligência como a aplicação de modelos formais de raciocínio baseados em lógica e inferências computacionais (CHARNIAK; MCDERMOTT, 1985);

c) Sistemas que atuam como seres humanos: associam a IA à construção de máquinas capazes de executar tarefas que exigiriam inteligência se fossem realizadas por pessoas (KURZWEIL, 1990);

d) Sistemas que atuam racionalmente: entendem a IA como o estudo de agentes que tomam decisões autônomas orientadas a metas, com base nas percepções do ambiente (POOLE; MACKWORTH; GOEBEL, 1998).

Conforme proposto por Russell et al. (2010), a inteligência artificial pode ser classificada com base em duas dimensões: pensamento vs. comportamento, e humano vs. racional. No contexto atual, a ênfase recai sobre a abordagem que busca criar agentes que atuem de forma racional em ambientes complexos.

Dentre os diversos subcampos da inteligência artificial — como raciocínio automatizado, visão computacional e processamento de linguagem natural — o aprendizado de máquina destacou-se, nos últimos anos, como o principal paradigma adotado no desenvolvimento de sistemas inteligentes (JORDAN; MITCHELL, 2015). Com os avanços recentes em aprendizado profundo, tornou-se possível modelar relações não lineares complexas e extrair automaticamente padrões latentes a partir de grandes volumes de dados sensoriais. Essas capacidades tornam o aprendizado profundo particularmente promissor para aplicações em prognóstico de falhas e na estimativa da RUL, especialmente em cenários com degradações complexas e alta dimensionalidade dos sinais (WU et al., 2024).

2.6 APRENDIZADO DE MÁQUINA

Ao desenvolver sistemas de apoio à decisão em ambientes complexos, os responsáveis buscam obter o melhor desempenho possível na execução. Em muitos casos isso se torna um grande desafio, visto a enorme quantidade de situações diferentes que o algoritmo deverá atuar.

O aprendizado de máquina, pode ser compreendido como a capacidade de sistemas computacionais aprimorarem seu desempenho por meio da análise de dados, ajustando seus comportamentos sem a necessidade de serem explicitamente programados para cada tarefa específica. Essa perspectiva, já discutida por Samuel (1959), caracteriza como um campo voltado ao desenvolvimento de algoritmos capazes de aprender a partir da experiência.

Luger (2013) define a capacidade de aprender como parte fundamental de qualquer sistema que reivindique possuir inteligência em um sentido geral. De forma semelhante, Simon (1983) entende o aprendizado como qualquer mudança em um sistema que melhore seu desempenho na segunda vez em que ele repita a mesma tarefa ou outra tarefa semelhante. Russell et al. (2010) complementam que um programa estará aprendendo se for capaz de melhorar seu desempenho nas tarefas após fazer observações em exemplos.

Mais recentemente, Jordan e Mitchell (2015) ampliaram essa definição, descrevendo o aprendizado de máquina como a disciplina que busca compreender e desenvolver algoritmos capazes de melhorar automaticamente seu desempenho com a experiência. Essa visão reflete o papel central dessa área como paradigma dominante da inteligência artificial contemporânea.

Russell et al. (2010) iniciam sua abordagem sobre aprendizado de máquina com um questionamento fundamental: por que desejaríamos que um sistema fosse capaz de aprender? Essa reflexão é especialmente pertinente para aqueles que estão tendo o primeiro contato com o tema. Uma vez que é percebido melhorias a serem implementadas, por que simplesmente os desenvolvedores não programam essas melhorias? Os autores listam três principais motivos:

a) Não é possível prever e simular todos os cenários em que a aplicação será exposta. Por exemplo, um robô desenvolvido para ganhar jogos de xadrez poderá se deparar com diferentes cenários a cada partida, seja em razão de enfrentar diferentes jogadores além das próprias circunstâncias da partida (SILVER et al., 2017);

b) Não se consegue saber quais mudanças ocorrerão ao longo do tempo. Um programa desenvolvido para prever os preços do mercado de ações do dia seguinte deve aprender a se adaptar quando o cenário de alta muda para baixa (LU et al., 2020);

c) Limitação dos programadores humanos. Por exemplo, grande parte das pessoas consegue reconhecer o rosto de conhecidos, mas até mesmo os melhores pro-

> gramadores são incapazes de desenvolver programas que realizam tal tarefa sem utilizar algoritmos de aprendizagem (AUTOR, 2014).

Em vez de inserir o conhecimento em computadores através de códigos, o aprendizado de máquina busca, através da observação de exemplos, aprender automaticamente padrões e relacionamentos relevantes (BISHOP; NASRABADI, 2006). Essa capacidade de extrair padrões complexos a partir dos dados tem impulsionado sua aplicação em diversos setores industriais.

O objetivo deste trabalho é investigar soluções que contribuam para o aumento da disponibilidade dos equipamentos, por meio da aplicação de técnicas de aprendizado de máquina no prognóstico de falhas. A utilização de modelos capazes de aprender padrões de degradação a partir de dados históricos torna-se particularmente atrativa, uma vez que tais sistemas podem evoluir continuamente suas previsões conforme novos dados são coletados (ZHANG et al., 2019).

É importante ressaltar, entretanto, que o bom desempenho dos algoritmos de aprendizado de máquina depende diretamente da qualidade e da representatividade dos dados utilizados para o treinamento. Conjuntos de dados incompletos ou desatualizados podem comprometer a acurácia das previsões, exigindo constante atualização e validação dos modelos. Estudos recentes destacam que a escassez de dados rotulados e a necessidade de manutenção da qualidade dos conjuntos de treinamento estão entre os principais desafios para a adoção efetiva de técnicas *data-driven* em prognóstico industrial (LI et al., 2024).

2.6.1 PRINCIPAIS FORMAS DE APRENDIZAGEM

2.6.1.1 Aprendizagem não supervisionada

Baseado nos exemplos de entrada, o algoritmo aprende padrões mesmo não sendo fornecido nenhum *feedback* explícito. A ideia é que o algoritmo faça o agrupamento dos exemplos recebidos de acordo com a similaridade de seus atributos. Esses exemplos são fornecidos sem rótulos e cabe ao algoritmo compor agrupamentos baseado nas semelhanças (RUSSELL et al., 2010).

No aprendizado não supervisionado, não há um “professor” que indique as respostas corretas; cabe ao próprio algoritmo identificar padrões nos dados e formar seus conceitos de maneira autônoma (LUGER, 2013).

No contexto do prognóstico de falhas, a aprendizagem não supervisionada tem sido utilizada para detectar anomalias e agrupar padrões de degradação em sensores de equipamentos industriais. Essa abordagem é útil quando não há rótulos explícitos de falhas, permitindo identificar estados anormais de funcionamento que podem anteceder uma falha (ZHANG et al., 2019).

2.6.1.2 Aprendizagem por reforço

O agente não sabe diretamente o que fazer ou quais ações tomar, mas descobre quais ações proporcionam as maiores recompensas por meio da exploração. O comportamento do agente afeta não apenas a recompensa imediata, mas também as ações e as eventuais recompensas subsequentes (LUGER, 2013).

O algoritmo não recebe a resposta correta, mas sim sinais de reforço que podem ser punições ou recompensas. Cabe ao mesmo decidir, a partir de uma hipótese feita sobre o exemplo recebido qual das ações realizadas foram as responsáveis pelo resultado do reforço.

Na área de manutenção preditiva, a aprendizagem por reforço vem sendo explorada para otimizar estratégias de manutenção. Ao simular diferentes cenários, o agente pode aprender a recomendar o momento mais adequado para intervenções, equilibrando custo de manutenção e risco de falhas (LI et al., 2024).

2.6.1.3 Aprendizagem supervisionada

É o método de aprendizado mais utilizado. Tendo como objetivo construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados, nessa modalidade, ao observar os exemplos de entrada e saída, o algoritmo aprende uma função que faz o mapeamento da entrada para a saída (RUSSELL et al., 2010).

No prognóstico de falhas, a aprendizagem supervisionada é a abordagem predominante para estimar a RUL, pois modelos podem ser treinados a partir de dados rotulados de falhas conhecidas. Esse método permite prever com maior precisão o tempo até a falha, sendo amplamente adotado em estudos com o *dataset* C-MAPSS e em aplicações reais de manutenção baseada em condição (ZHANG et al., 2019).

2.6.2 NORMALIZAÇÃO DE DADOS

Em tarefas de aprendizado de máquina, é comum lidar com variáveis de entrada em escalas distintas, o que pode comprometer a estabilidade numérica e o desempenho de modelos baseados em gradiente, como redes neurais profundas. Para mitigar esse problema, aplica-se a normalização dos dados, ou seja, o reescalonamento das variáveis para uma faixa comum. Neste trabalho, foi utilizada a técnica de normalização Min-Max, que transforma os dados para o intervalo [0, 1] com base nos valores mínimo (x_{min}) e máximo (x_{max}) observados, conforme a Equação 1:

$$x_{\text{norm}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \tag{1}$$

Essa abordagem é particularmente eficaz em redes neurais, pois preserva a distribuição dos dados, evita distorções causadas por variáveis com magnitudes desbalanceadas e acelera a convergência durante o treinamento (PATRO; SAHU, 2015). No contexto de séries temporais multivariadas utilizadas em prognóstico de falhas, a normalização Min-Max permite que sensores com escalas diferentes contribuam de forma equilibrada para o processo de aprendizagem.

Neste trabalho, a normalização Min-Max foi aplicada de forma global por atributo, utilizando exclusivamente os valores mínimos e máximos calculados a partir do conjunto de treinamento. Em seguida, os mesmos parâmetros de escala foram reutilizados para transformar o conjunto de teste, garantindo consistência entre os conjuntos e evitando vazamento de informação dos dados de teste para a fase de treinamento.

2.6.3 PRÉ-PROCESSAMENTO EM SÉRIES TEMPORAIS: JANELA DESLIZANTE

O uso de janelas deslizantes ou *sliding windows* é uma técnica recorrente no pré-processamento de séries temporais multivariadas voltadas ao prognóstico de falhas. Essa abordagem transforma sequências históricas de sensores em amostras supervisionadas, permitindo que modelos de aprendizado de máquina aprendam a estimar a RUL com base em padrões recentes de degradação. Ao deslocar progressivamente uma janela fixa ao longo da trajetória de um ativo, obtém-se uma representação mais rica da evolução temporal do sistema. Essa técnica tem sido aplicada com sucesso em diferentes domínios, como na previsão de preços de ações com redes neurais (HOTA; HANDA; SHRIVAS, 2017), e é amplamente adotada em estudos de manutenção preditiva com o *dataset* C-MAPSS.

Para além da transformação das séries temporais em janelas supervisionadas, a literatura enfatiza a importância de que a divisão entre os conjuntos de treinamento e validação respeite a ordem temporal dos dados, evitando qualquer forma de vazamento de informação. Em problemas de prognóstico, é inadequado utilizar amostras futuras para ajustar o modelo, pois isso induz um cenário artificialmente favorável e incompatível com aplicações reais, comprometendo a confiabilidade da avaliação. Conforme destacam Yang, Li e Jiang (2024), a utilização indevida de informações futuras durante o treinamento gera vazamento de informação, resultando em avaliações excessivamente otimistas e pouco representativas do desempenho em cenários reais.

No caso específico do C-MAPSS, cada unidade apresenta uma trajetória completa e independente de degradação. Por essa razão, estudos destacam que a separação entre treino e validação deve ocorrer por unidade, e não por janela individual, garantindo que todo o histórico de um motor pertença exclusivamente a um único conjunto (LI; DING; SUN, 2018). Essa prática assegura coerência temporal e impede que padrões futuros de uma mesma série apareçam inadvertidamente no processo de treinamento, o que resultaria em avaliação irrealisticamente otimista.

2.6.4 ENGENHARIA DE ATRIBUTOS PARA SÉRIES TEMPORAIS

Este trabalho complementa os sensores brutos do FD001 com atributos derivados capazes de representar padrões locais de degradação que não são explicitamente

observáveis nos sinais originais. A literatura em PHM mostra que, para estimativa de RUL, é essencial incorporar informação temporal estruturada em vez de tratar cada ciclo como uma observação independente.

No estudo clássico de Heimes (2008), é enfatizado que modelos que ignoram a dinâmica temporal - como arquiteturas estáticas baseadas em linhas de atraso - tendem a superajustar e não capturam adequadamente o processo de desgaste. Esse resultado reforça a necessidade de representar a evolução temporal dos sensores, seja por modelagem recorrente, seja pela construção de atributos derivados.

No campo de engenharia de atributos, Christ et al. (2018) mostram que estatísticas locais computadas sobre janelas temporais — como médias, desvios-padrão e tendências lineares — estão entre os atributos mais relevantes extraídos automaticamente pelo pacote *tsfresh*, tornando-se elementos fundamentais na caracterização de séries temporais.

De forma específica para o problema de RUL, Alomari, Andó e Baptista (2023) demonstram, aplicando um pipeline interpretável ao conjunto C-MAPSS, que atributos estatísticos simples — como média, variabilidade e medidas de tendência local — capturam a dinâmica de degradação e contribuem para resultados competitivos quando combinados com seleção de atributos. Esses autores utilizam janelas deslizantes para extrair características que representam suavização, variação e tendência local dos sensores.

Além disso, Wang et al. (2022) destacam que atributos derivados de janelas móveis são componentes essenciais em pipelines de previsão com séries temporais, reforçando que a extração de estatísticas locais (incluindo as três utilizadas neste trabalho) melhora a expressividade do modelo com baixo custo adicional.

Com base nesses fundamentos, este estudo adota uma abordagem enxuta e interpretável: para cada sensor não constante, são extraídos três atributos em uma janela móvel de cinco ciclos:

a) **Média móvel (mean5)** — representa o nível médio recente do sinal;

b) **Desvio-padrão móvel (std5)** — captura a variabilidade local;

c) **Inclinação temporal (slope)** — calculada como a diferença entre o valor atual e o ciclo anterior.

Esses atributos são incorporados às janelas temporais de 30 ciclos utilizadas como entrada no modelo LSTM, permitindo que a rede observe tanto os valores brutos quanto suas estatísticas locais. Assim, a engenharia de atributos adotada amplia a capacidade do modelo de identificar tendências de degradação, mantendo o controle da dimensionalidade e preservando a interpretabilidade.

2.7 APRENDIZADO PROFUNDO

O Aprendizado Profundo é uma subárea do aprendizado de máquina que se baseia no uso de redes neurais artificiais com múltiplas camadas para extrair representações hierárquicas dos dados. Segundo LeCun, Bengio e Hinton (2015), essa abordagem se tornou um paradigma dominante da inteligência artificial moderna, permitindo avanços expressivos em visão computacional, processamento de linguagem natural e prognóstico de falhas em sistemas complexos.

Uma das principais características é sua capacidade de aprender automaticamente representações relevantes a partir de grandes volumes de dados, superando a limitação dos métodos clássicos de aprendizado de máquina que dependiam de extração manual de características. Esse potencial explica o sucesso do aprendizado profundo em PHM, especialmente na previsão da RUL (ZHANG et al., 2019).

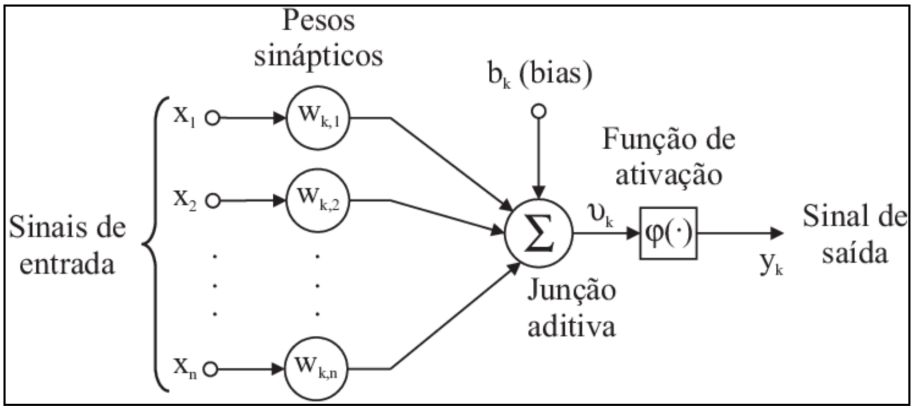
Na sequência, serão discutidas algumas arquiteturas de redes neurais artificiais que compõem a base do aprendizado profundo, com destaque para perceptrons multicamadas, Recurrent Neural Network (RNN) e sua variação LSTM.

2.8 REDES NEURAIS ARTIFICIAIS

Na definição de Chassagnon et al. (2020), redes neurais artificiais são projetadas para imitar o modo como o cérebro humano processa informações, utilizando sucessões de operações simples que simulam o comportamento dos neurônios biológicos. Janiesch, Zschech e Heinrich (2021) complementam essa definição dizendo que cada conexão entre neurônios transmitem sinais que são ajustados continuamente ao longo do processo de aprendizagem, podendo ser amplificados ou atenuados. Baseado nos sistemas biológicos, os neurônios artificiais são representações matemáticas de unidades de processamento conectadas. Existem vários tipos de Redes Neurais diferentes que são implementadas nas mais variáveis arquiteturas. Nesse trabalho, falaremos sobre as Redes Perceptron de Multiplas Camadas e as Redes de Memória de Longo-Curto Prazo.

As redes neurais são capazes de aprender com os dados analisados, isso significa que elas são capazes de encontrar uma forma de atingir um objetivo estabelecido como por exemplo classificar imagens, prever valores, sem a necessidade de um desenvolvedor criar uma série de códigos contendo testes condicionais. Utilizando o método de aprendizagem supervisionada, por exemplo, a rede recebe apenas os dados de entrada e saída na etapa de treinamento e é ela quem encontra as relações para transformar as entradas da etapa de teste na saída esperada.

Figura 1 – Neurônio Computacional



Fonte: Bianchini (2001)

A Figura 1 mostra o modelo de um neurônio computacional, base da construção de redes neurais. Ele é composto por três elementos principais:

a) Conjunto de sinapses: O conjunto de sinapses é composto pelos sinais de entrada e pesos sinápticos, de modo que, cada sinal de entrada será multiplicado pelo peso sináptico associado a ele. Assim, o neurônio irá tratar de maneira diferente cada impulso recebido, atribuindo-lhe um valor;

b) Um somador: Também conhecido como bias é responsável por somar os sinais de entrada, ponderados pelo respectivo valor de peso;

c) Função de ativação: é utilizada para restringir a amplitude do sinal de saída do neurônio.

Esse modelo é o alicerce para arquiteturas mais complexas, em que diversas unidades são interconectadas para resolver problemas como classificação, regressão e previsão de séries temporais (SCHMIDHUBER, 2015; BIANCHINI, 2001).

2.8.1 REDES NEURAIS PERCEPTRON MULTICAMADAS

O perceptron multicamadas ou multi-layer perceptron (MLP) é uma das arquiteturas mais utilizadas e serve como um modelo de referência (*baseline*) neste trabalho. Ele é caracterizado por possuir ao menos uma camada intermediária (oculta) entre a entrada e a saída. Seu funcionamento é baseado no aprendizado supervisionado: durante o treinamento, a rede ajusta iterativamente os pesos sinápticos de acordo com os erros cometidos em relação aos valores esperados (FANUCCHI; OLESKOVICZ; BARBOSA, 2013; JOST, 2015).

O desempenho das redes MLP pode ser sensivelmente impactado pela escolha das variáveis utilizadas nos vetores de entrada. Para contornar esse desafio, estudos têm utilizado técnicas de seleção de atributos, como o algoritmo Random Forest, para reduzir a dimensionalidade e selecionar sensores mais relevantes (WANG et al., 2023).

Apesar de sua ampla aplicação em tarefas de classificação e regressão, o MLP apresenta limitações quando utilizado em séries temporais longas. Isso ocorre porque a arquitetura *feedforward* não possui mecanismos internos de memória, exigindo es-

tratégias adicionais para capturar dependências temporais. Essa restrição motivou o surgimento de arquiteturas mais avançadas, como as redes neurais recorrentes.

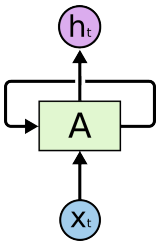
Neste trabalho, a arquitetura MLP foi adotada como modelo *baseline* por sua simplicidade e ampla aplicação em tarefas de regressão. Essa escolha está alinhada à literatura recente, que emprega redes do tipo Perceptron Multicamadas como ponto de partida para comparação com modelos mais sofisticados de prognóstico de falhas (WANG et al., 2023).

2.8.2 REDES NEURAIS RECORRENTES

Redes neurais recorrentes ou recurrent neural Network (RNN) utilizam-se de *loops* (estruturas de repetição), conforme podemos observar na Figura 2, para persistir dados processados anteriormente usando-os como base de conhecimento para novos aprendizados. Basicamente elas fazem com que uma RNN seja capaz de armazenar dados já processados não precisando recomeçar do zero durante o processo de aprendizagem, utilizando as informações já processadas como uma forma de "entender"o contexto do dado que esta sendo processado no momento (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para que possam armazenar os dados processados anteriormente, as redes desse tipo possuem modificações quando comparadas a outras como Perceptron e MLP, umas dessas modificações são nas unidades da rede que conhecemos como neurônio, elas recebem a capacidade de armazenar uma memória interna onde ficam contidas as informações sobre os dados já processados. Dessa forma, elas acabam diferindo do funcionamento dos neurônios humanos e com isso autores como Hochreiter e Schmidhuber (1997) nomeiam as unidades da rede como "célula".

Figura 2 – Rede Neural Recorrente



Fonte: Olah (2015)

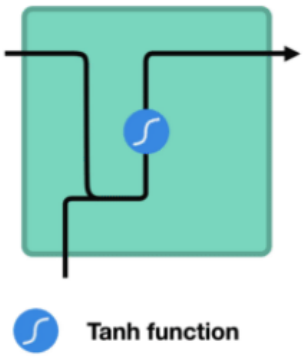
As informações sobre entradas passadas são armazenadas por um período de tempo não determinado inicialmente mas que é definido baseado nos pesos dos dados de entrada. Ao aumentar o intervalo de informações fornecidas, a rede pode enfrentar dificuldade ao tentar acessar informações processadas a muito tempo. Isso é apontado, através de teorias e experimentos feitos por Bengio, Simard e Frasconi (1994) explorando a limitação relacionada as dependências de longo prazo. Esse problema limita a capacidade das RNNs clássicas em processar séries temporais mais extensas.

2.8.2.1 Redes Neurais de Memória de Longo-Curto Prazo (LSTM)

As redes Long Short-Term Memory, introduzidas por Hochreiter e Schmidhuber (1997) foram desenvolvidas para solucionar o problema das dependências de longo prazo, elas possuem estruturas capazes de aprender e selecionar quais dados são relevantes para se manter armazenada na memória de longo tempo, descartando os dados desnecessários.

Na Figura 3 podemos observar a composição de uma célula da rede recorrente tradicional. Ela apresenta uma função de ativação Tanh que recebe a combinação de uma nova entrada e as entradas anteriores que atuam como a memória da rede. A função Tanh é responsável por manter os valores no intervalo de -1 e 1. O resultado da função será todas as entradas anteriores que serão passadas para uma nova célula.

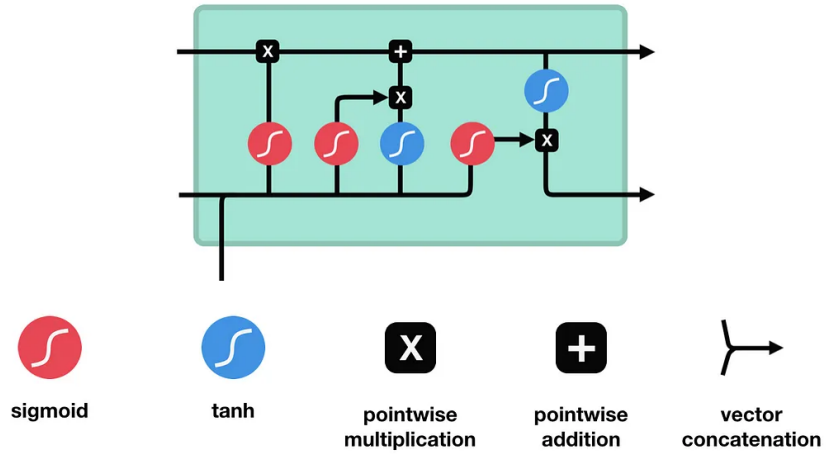
Figura 3 – Arquitetura básica de uma rede neural recorrente



Fonte: Phi (2018)

Uma rede LSTM implementa modificações na estrutura da célula para que seja possível armazenar apenas os dados relevantes. No modelo proposto por Hochreiter e Schmidhuber (1997), uma célula dispõe de camadas internas de controle chamadas portões. A Figura 4 demonstra a arquitetura da célula LSTM.

Figura 4 – Arquitetura básica de uma rede LSTM



Fonte: Phi (2018)

Diferente da função Tanh que comprime os valores entre -1 e 1, a Sigmóide comprime entre 0 e 1 e são utilizadas nesse caso para fazer com que dados sejam descartados.

a) Primeiramente os dados passam pelo portão de esquecimento que utiliza uma função sigmóide para definir quais dados da combinação entre estado oculto anterior, que seria a memória da rede, e as informações da entrada atual irão permanecer, valores mais próximos de zero serão esquecidos;

b) O próximo passo dar-se pela utilização do portão de entrada, nele a combinação entre estado oculto anterior e entrada atual são enviados para uma função sigmóide que seleciona os dados importantes e tanh que mantem os valores regulados, logo após os resultados de ambas funções são multiplicados;

c) Com esses dados é possível calcular o estado atual da célula. O estado anterior é multiplicado pela saída do portão de esquecimento removendo valores julgados como não importantes e logo após é somado a saída do portão de entrada onde novas informações definidas nas etapas anteriores são passadas para o estado atual da célula;

d) A última etapa é feita no portão de saída, nesse portão é definido qual será o

próximo estado oculto. O estado oculto anterior é combinado com a entrada atual e são submetidos a uma função sigmóide, posteriormente são multiplicados pelos valores de saída de uma função tanh alimentada com o estado atual da célula definido na etapa anterior.

Em resumo, o portão de esquecimento fica encarregado de descartar valores não importantes. O portão de entrada define o que possui relevância para ser adicionado no estado atual e o portão de saída sacramenta o próximo estado oculto. Esses portões permitem a rede controlar o fluxo de informações, o que resolve o problema de acessar informações processadas há muito tempo, tornando a arquitetura LSTM especialmente eficaz para modelagem de séries temporais complexas e amplamente utilizada em prognóstico de falhas (GOODFELLOW; BENGIO; COURVILLE, 2016; GREFF et al., 2017).

Diversos estudos demonstram que arquiteturas LSTM podem alcançar melhor capacidade de modelagem quando organizadas de forma empilhada, isto é, com múltiplas camadas recorrentes sucessivas. Essa configuração aprofunda o nível de representação temporal e permite que camadas superiores capturem padrões de degradação mais abstratos e de maior complexidade. Revisões sistemáticas, como a de Greff et al. (2017), destacam que o empilhamento de LSTMs é uma prática consolidada em aplicações de séries temporais, especialmente em cenários que exigem modelagem de dinâmicas não lineares de longo prazo, como é o caso do prognóstico da vida útil remanescente.

2.8.3 CONEXÕES RESIDUAIS EM REDES PROFUNDAS

As conexões residuais foram introduzidas por He et al. (2016) como uma solução para o problema de degradação em redes profundas, no qual o aumento da profundidade dificulta o processo de otimização e pode levar à piora do desempenho. A proposta consiste em permitir que cada bloco aprenda uma função residual F(x), combinada diretamente com a entrada original por meio de um atalho de identidade, resultando na expressão F(x) + x. Esse mecanismo melhora o fluxo de gradientes, estabiliza o treinamento e viabiliza arquiteturas significativamente mais profundas.

Embora originalmente aplicadas em redes convolucionais, Wu et al. (2016) demonstraram que o mesmo princípio pode ser estendido a redes recorrentes profundas. No sistema Google Neural Machine Translation, os autores empregaram conexões residuais entre camadas LSTM empilhadas, permitindo treinar arquiteturas substancialmente mais profundas com estabilidade e melhor propagação do gradiente.

Com base nesses fundamentos teóricos, este trabalho adota uma conexão residual entre as duas primeiras camadas LSTM do modelo proposto, com o objetivo de facilitar a otimização e aprimorar a capacidade da rede em capturar padrões temporais complexos presentes no conjunto FD001.

2.8.4 FUNÇÃO DE ATIVAÇÃO

A função de ativação, também chamada de função de transferência, é um componente matemático fundamental das redes neurais. Ela aplica uma transformação não linear sobre a combinação linear das entradas ponderadas pelos pesos sinápticos, determinando o valor de saída do neurônio. Sem essa não linearidade, a rede seria incapaz de representar funções complexas, limitando-se a modelos lineares (GOODFELLOW; BENGIO; COURVILLE, 2016).

Historicamente, as funções mais utilizadas foram a logística (sigmóide) e a tangente hiperbólica (tanh), ambas adequadas para compressão de valores em intervalos restritos (SANTOS et al., 2005). No caso específico das redes LSTM, essas funções desempenham um papel central: a sigmóide é utilizada nos portões para controlar o fluxo de informações (quais valores devem ser esquecidos ou armazenados), enquanto a tanh regula a escala dos estados internos, mantendo-os entre -1 e 1.

2.9 AVALIAÇÃO DE DESEMPENHO DO MODELO

Como o objetivo da rede neural é prever os valores de RUL dos equipamentos, é necessário utilizar métricas para avaliar o desempenho, medindo a precisão das previsões. Para modelos de regressão, algumas métricas são comumente utilizadas, entre as quais destacam-se:

a) Erro Médio Absoluto: Mean Absolute Error (MAE);

b) Raiz do Erro Quadrático Médio: Root Mean Squared Error (RMSE).

Willmott e Matsuura (2005) argumentam que o RMSE não é a métrica mais adequada, pois sua função expressa três aspectos diferentes da distribuição dos erros, e não apenas o erro médio. Para os autores, o MAE representa de forma mais natural a magnitude média dos erros, enquanto o RMSE, à medida que a variabilidade aumenta, tende a assumir valores cada vez maiores em relação ao MAE. Dessa forma, eles defendem o uso preferencial do MAE.

Em contrapartida, Chai e Draxler (2014) apresentam uma visão divergente, destacando que não é recomendável descartar o RMSE. Segundo os autores, cada métrica possui particularidades que se ajustam melhor a determinados tipos de modelos. No caso de erros com distribuição aproximadamente normal, o RMSE tende a ser mais apropriado. Por isso, defendem que a combinação das métricas permite uma avaliação mais robusta e eficiente do desempenho dos modelos.

De modo geral, ambas as métricas medem a diferença entre os valores previstos e os valores reais. O MAE calcula a média das diferenças absolutas, tratando todos os erros da mesma forma e sendo menos sensível a valores discrepantes (outliers). O RMSE, por sua vez, eleva os erros ao quadrado antes da média e, por isso, penaliza mais fortemente os erros grandes, tornando-se mais sensível a outliers.

Considerando essas diferentes perspectivas, este trabalho adotará tanto o MAE quanto o RMSE como métricas complementares, buscando uma avaliação mais abrangente do desempenho preditivo da rede neural no prognóstico da vida útil remanescente.

Além dessas métricas tradicionais, estudos da área de PHM frequentemente empregam a pontuação S (S-score) como forma complementar de avaliação, especialmente em cenários onde erros assimétricos possuem impacto operacional distinto. Introduzida na competição PHM08 Saxena et al. (2008), essa métrica foi projetada para penalizar subestimativas de RUL de maneira mais severa do que superestimativas, refletindo

a criticidade de prever que um equipamento falhará antes do tempo real. A função de penalidade é exponencial e assimétrica, atribuindo custos maiores a previsões excessivamente pessimistas, uma vez que essas podem conduzir a substituições prematuras, interrupções operacionais e desperdício de recursos. Dessa forma, a pontuação S complementa o MAE e o RMSE ao incorporar explicitamente a noção de risco associada ao erro de previsão, sendo amplamente utilizada em trabalhos que analisam o desempenho de modelos de prognóstico no conjunto C-MAPSS e em aplicações industriais críticas.

Em suma, este capítulo estabeleceu os fundamentos teóricos essenciais para a pesquisa, abordando desde os conceitos de manutenção preditiva e a importância da estimativa da RUL até as especificidades das redes neurais recorrentes (LSTM) e as métricas de avaliação. Com a base conceitual e o estado da arte sobre o conjunto de dados C-MAPSS consolidados, o capítulo a seguir detalhará o percurso metodológico adotado, descrevendo as etapas práticas de pré-processamento, a arquitetura dos modelos e as estratégias experimentais definidas para alcançar os objetivos do trabalho.

3 METODOLOGIA

Este capítulo descreve o percurso metodológico adotado para alcançar o objetivo central deste trabalho, que consiste em desenvolver, treinar e avaliar modelos de redes neurais para estimativa da vida útil remanescente de turbofans. São apresentados: (i) a caracterização da pesquisa; (ii) o conjunto de dados utilizado; (iii) as tecnologias empregadas; (iv) os procedimentos de pré-processamento e preparação dos dados; (v) a modelagem dos modelos MLP e LSTM; (vi) a estratégia de treinamento e teste; (vii) as métricas de avaliação; (viii) a forma de comparação com resultados publicados; e (ix) a estratégia de disponibilização do modelo desenvolvido. O detalhamento busca garantir reprodutibilidade e transparência experimental.

3.1 CARACTERIZAÇÃO DA PESQUISA

A pesquisa pode ser caracterizada como aplicada, quantitativa e de natureza experimental. É aplicada porque visa resolver um problema prático de manutenção: estimar antecipadamente o momento de falha de um equipamento de modo a permitir ações de manutenção preditiva e redução de paradas não planejadas. É quantitativa pois opera sobre medições numéricas de sensores e produz estimativas numéricas de vida útil remanescente. É experimental porque envolve a implementação, o treinamento e a avaliação empírica de modelos computacionais sob diferentes condições de configuração.

A motivação para este estudo surgiu da observação de processos reais de manutenção em ambiente industrial. A partir dessa necessidade prática, investigou-se o uso de métodos de aprendizado profundo capazes de aprender padrões de degradação ao longo do tempo e, assim, prever a RUL do componente monitorado antes da ocorrência da falha.

3.2 CONJUNTO DE DADOS

3.2.1 ORIGEM E DESCRIÇÃO

Os dados utilizados foram obtidos do simulador C-MAPSS, desenvolvido pelo Prognostics Center of Excellence da NASA. Esse ambiente de simulação produz séries temporais multivariadas com medições de sensores de motores turbofan sob diferentes condições operacionais e perfis de degradação (FREDERICK; DECASTRO; LITT, 2007).

3.2.2 SUBCONJUNTO SELECIONADO: FD001

Neste trabalho, foi utilizado o subconjunto FD001, caracterizado por um único modo de falha e uma única condição operacional. Os detalhes sobre os sensores disponíveis e as configurações podem ser visualizados na Tabela A.1. O conjunto dispõe de 100 motores no conjunto de treino e 100 no conjunto de teste. Essa escolha simplifica o problema inicial e facilita a comparação com resultados da literatura. Além disso, o FD001 é amplamente adotado como *benchmark* em estudos sobre prognóstico de falhas, por oferecer um cenário mais controlado para avaliação isolada da arquitetura do modelo.

Os dados de teste do subconjunto FD001 são fornecidos separadamente em relação ao conjunto de treinamento, tanto em termos de registros quanto de ciclos operacionais. Ambos os conjuntos contêm 100 motores simulados, mas com séries temporais distintas: no conjunto de treinamento, os 100 motores totalizam 20.631 registros, acompanhando cada unidade até a ocorrência da falha. No conjunto de teste, os mesmos motores somam 13.096 registros, interrompidos antes da falha e acompanhados de um arquivo auxiliar contendo a RUL verdadeira de cada unidade. Apesar de os identificadores de motor se repetirem, nenhuma linha de medição do conjunto de teste está presente no conjunto de treino, garantindo a integridade da separação. Essa configuração visa simular um cenário realista de predição, onde o modelo deve estimar a vida útil remanescente com base em dados parciais, permitindo avaliar sua capacidade de generalização.

3.3 TECNOLOGIAS E FERRAMENTAS

A implementação foi realizada em Python, utilizando as bibliotecas TensorFlow e Keras para construção e treinamento dos modelos, Pandas e NumPy para manipulação dos dados, Scikit-Learn para pré-processamento e avaliação, e Matplotlib para visualização dos resultados. O ambiente Google Colab foi empregado como plataforma de execução, devido à sua acessibilidade e suporte nativo a bibliotecas de aprendizado de máquina. No entanto, buscando um maior rigor na reprodutibilidade e na comparação dos resultados, o treinamento final e a avaliação do modelo foram realizados em um ambiente local com especificações fixas de hardware: um processador Intel Core i5-12400F, 16GB de memória RAM e uma GPU NVIDIA GeForce RTX 2060. O uso deste material de processamento paralelo foi crucial para a viabilidade e aceleração do treinamento de redes neurais profundas.

3.4 PRÉ-PROCESSAMENTO DOS DADOS

3.4.1 ANÁLISE EXPLORATÓRIA

Foi conduzida uma análise preliminar dos sensores para seleção das variáveis mais informativas para o modelo, com base em sua variabilidade e relação com o tempo de falha.

Conforme discutido no capítulo anterior e com base em estudos da literatura que destacam a baixa relevância de alguns sensores do conjunto FD001, como Mitici et al. (2023), foi adotada uma estratégia de seleção de atributos. Foram mantidos apenas os sensores mais representativos, somando 15 sensores físicos e 2 variáveis operacionais (setting_1 e setting_2).

3.4.2 CÁLCULO DA RUL

A RUL foi calculada para cada motor como o número de ciclos restantes até a falha, conforme amplamente adotado na literatura. Para cada unidade i, com falha ocorrendo no ciclo C_i, a RUL no ciclo t é dada por:

$$\text{RUL}_{i,t} = C_i - t \tag{2}$$

A estimativa da RUL, sendo um problema de regressão, foi modelada para que a rede neural fornecesse como saída o valor contínuo do tempo restante até a falha. Essa escolha justifica a utilização do Erro Quadrático Médio como função de perda durante o treinamento e de métricas baseadas no erro contínuo, como o MAE e o RMSE, na avaliação de desempenho, conforme detalhado na Seção 3.7.

3.4.3 NORMALIZAÇÃO

Para uniformizar a escala das variáveis de entrada e otimizar o desempenho dos modelos de redes neurais, foi aplicada a normalização do tipo Min-Max de forma global por atributo, utilizando como referência os valores mínimo e máximo observados no conjunto de treino. Esses mesmos parâmetros foram posteriormente aplicados aos dados de validação e teste, garantindo a consistência entre os conjuntos e evitando vazamento de informações durante o processo de modelagem.

Nos experimentos iniciais com a rede LSTM, a normalização foi realizada individualmente por unidade operacional (motor), com a aplicação de um scaler distinto para cada série temporal. Essa estratégia visava preservar características intrínsecas ao funcionamento de cada equipamento, porém, introduzia discrepâncias entre os conjuntos de treino e teste, comprometendo a comparabilidade e a generalização do modelo. Diante disso, a abordagem foi substituída pela normalização global por atributo, que se mostrou mais adequada do ponto de vista metodológico e experimental.

3.4.4 JANELA TEMPORAL

Os dados foram organizados utilizando janelas deslizantes de tamanho fixo, com $T = 30$ ciclos. Cada amostra foi composta pelas leituras de múltiplos sensores ao longo dos 30 ciclos anteriores ao ciclo alvo, e a RUL associada ao último ciclo da janela foi usada como saída esperada.

A escolha do valor de 30 ciclos como tamanho da janela baseou-se na literatura especializada. Li, Ding e Sun (2018) aplicaram esse mesmo valor em sua proposta de estimativa de RUL utilizando redes neurais profundas, demonstrando que janelas com essa dimensão permitem capturar padrões locais de degradação sem comprometer a eficiência computacional.

Após o cálculo da RUL, foi aplicado um limite superior de 125 ciclos (RUL cap), conforme discutido na Seção 2.2.1. Essa técnica visa evitar a dominância de valores elevados de RUL na função de perda e melhorar a estabilidade do treinamento do modelo.

3.5 MODELAGEM E IMPLEMENTAÇÃO

3.5.1 MODELO BASELINE: MLP

Como ponto de partida, foi desenvolvido um modelo baseline utilizando uma rede neural do tipo MLP. O modelo recebe como entrada os dados vetorizados de uma janela temporal fixa, sem considerar explicitamente a ordem dos ciclos. O objetivo desse modelo é servir como linha de base para avaliação dos ganhos obtidos com arquiteturas mais avançadas.

Para reduzir a dimensionalidade dos dados e selecionar variáveis mais informativas, foi aplicada uma etapa de seleção de atributos utilizando o algoritmo Random Forest. A partir dessa análise, foram selecionados os dez sensores com maior importância relativa: S_4, S_7, S_8, S_9, S_11, S_12, S_13, S_14, S_15 e S_21.

Os valores desses sensores foram então normalizados utilizando a técnica Min-Max.

A arquitetura do MLP foi composta por seis camadas ocultas com 25 neurônios cada, utilizando função de ativação ReLU, função de perda MSE e o otimizador L-BFGS com até 2.000 iterações. Essa configuração foi reproduzida com base na proposta de Wang et al. (2023), aplicada ao mesmo subconjunto FD001 do C-MAPSS.

O modelo MLP baseline foi treinado com o conjunto completo `train_FD001.txt` e testado com `test_FD001.txt`, conforme proposto no artigo de Wang et al. (2023). Não foi aplicada divisão 80/20 como nas redes LSTM, dado seu caráter introdutório comparativo. Essa escolha visa manter fidelidade à proposta original do artigo e estabelecer uma referência simples, porém funcional.

3.5.2 MODELO PRINCIPAL: LSTM

O modelo principal desenvolvido é baseado em uma arquitetura residual composta por três camadas LSTM unidirecionais. As duas primeiras camadas possuem 96

unidades cada, com ativação tanh e regularização por *dropout* de 10%. Suas saídas são combinadas por meio de uma soma elementar (residual), a qual é então processada por uma terceira camada LSTM com 48 unidades. Em seguida, aplica-se mais uma camada de *dropout*, conectada a uma camada densa com uma unidade de saída e ativação ReLU, utilizada para garantir predições não negativas de RUL.

A arquitetura foi implementada utilizando a API funcional do Keras, o que permitiu flexibilidade na construção do modelo, inclusão de conexões residuais e aplicação de estratégias de regularização.

3.5.3 BUSCA DE HIPERPARÂMETROS

Para a rede LSTM, empregou-se uma estratégia de busca aleatória (*random search*) para otimização dos hiperparâmetros. Foram definidos intervalos plausíveis com base em trabalhos da literatura e testes preliminares, sendo avaliadas diversas combinações aleatórias. A arquitetura da rede foi mantida fixa, variando-se os parâmetros de configuração.

A adoção da estratégia de busca aleatória nesta etapa teve como objetivo permitir a exploração eficiente de múltiplas configurações de hiperparâmetros dentro de um espaço multidimensional. Essa abordagem foi escolhida por sua capacidade de amostrar diversas combinações com baixo custo computacional, favorecendo a descoberta empírica de configurações promissoras sem a necessidade de varrer sistematicamente todo o espaço de busca. Essa decisão mostrou-se eficaz para guiar a construção do modelo LSTM com base em evidências observadas durante a validação.

A cada combinação, o modelo foi treinado e validado com base em uma divisão por unidade de motor, respeitando a ordem temporal de cada turbina e evitando sobreposição entre conjuntos. Foram utilizados mecanismos de *early stopping* e redução adaptativa da taxa de aprendizado. A configuração com menor erro de validação foi selecionada como finalista e posteriormente avaliada no conjunto de teste composto por motores não utilizados durante o treinamento, conforme descrito na próxima seção.

3.6 TREINAMENTO E TESTE DOS MODELOS

3.6.1 DIVISÃO DOS DADOS

O conjunto FD001 já fornece os dados separados em treino e teste por unidade de motor, o que foi mantido nesta pesquisa.

Além da separação original em treino e teste, o conjunto de treino foi subdividido em 80% das unidades (motores) para treinamento e 20% para validação. Essa divisão foi realizada por unidade, de modo que cada motor permanece completo em um único conjunto. Com isso, evita-se o vazamento de informações futuras, respeita-se a sequência temporal dos ciclos e simula-se um cenário realista de predição da vida útil remanescente. Dos 100 motores presentes no conjunto de treino, 80 foram utilizados para o treinamento do modelo, e 20 foram reservados exclusivamente para validação.

3.6.2 TREINAMENTO

Ambos os modelos foram treinados utilizando o otimizador Adam, com função de perda MSE. Mecanismos como *early stopping* e redução de taxa de aprendizado foram empregados para evitar sobreajuste.

3.6.3 TESTE

Após o treinamento, os modelos foram aplicados aos dados de teste para estimar a RUL. Para cada unidade, foi utilizada apenas a última janela de observações disponível antes da falha, simulando o cenário real de predição próxima ao fim de vida do equipamento. Essa abordagem é frequentemente adotada em *benchmarks* com o conjunto FD001 e permite uma comparação direta com resultados publicados na literatura.

3.7 MÉTRICAS DE AVALIAÇÃO DE DESEMPENHO

Foram utilizadas duas métricas principais:

a) **MAE** – Erro Médio Absoluto;

b) **RMSE** – Raiz do Erro Quadrático Médio.

Essas métricas foram escolhidas por sua ampla utilização na literatura e pela capacidade de capturar erros médios e erros extremos, respectivamente.

Além das métricas principais, este trabalho também utiliza a pontuação S (S-score) de forma complementar, restrita à etapa de comparação com resultados publicados na literatura. Essa métrica, tradicionalmente empregada em estudos que seguem o protocolo da competição PHM08, apresenta uma função de penalidade assimétrica que atribui custos maiores a subestimativas de RUL. Sua utilização neste trabalho tem como objetivo permitir um *benchmarking* consistente com pesquisas que adotam esse mesmo critério de avaliação.

3.8 ESTRATÉGIA DE COMPARAÇÃO

Os resultados do modelo LSTM foram comparados com abordagens existentes na literatura que utilizaram o mesmo subconjunto FD001 do C-MAPSS. A comparação foi conduzida de forma quantitativa, utilizando as métricas pontuação S e RMSE, adotadas amplamente nesse tipo de tarefa. Além disso, buscou-se garantir condições metodológicas equivalentes, considerando exclusivamente a última janela de cada unidade no conjunto de teste como entrada do modelo. Essa abordagem simula um cenário realista de predição próxima ao fim de vida útil dos equipamentos e favorece uma comparação justa e padronizada com outros estudos.

Essa estratégia é amplamente empregada em competições e estudos de referência, como Society (2016), Hsu e Jiang (2018), Elsheikh, Yacout e Ouali (2019), que também reportam um único valor de RUL por unidade.

3.9 DISPONIBILIZAÇÃO E REPRODUTIBILIDADE

Com o objetivo de garantir a transparência e a reprodutibilidade dos experimentos realizados, todo o processo de treinamento do modelo foi desenvolvido em Python e executado no Google Colab, incluindo as etapas de pré-processamento, engenharia de atributos, geração das janelas temporais e ajuste do modelo LSTM residual. Os artefatos gerados durante o treinamento — o modelo final (model.h5) e o normalizador (scaler.pkl) — foram exportados e integrados ao protótipo Web desenvolvido em

Streamlit. Todo o código-fonte utilizado no treinamento, na preparação dos dados e na aplicação do modelo foi disponibilizado em um repositório público no GitHub, organizado em diretórios separados para o protótipo, os artefatos do modelo e os arquivos de teste. Essa estrutura garante que pesquisadores e profissionais possam reproduzir integralmente o fluxo experimental apresentado neste trabalho.

Neste capítulo, foram definidos os procedimentos experimentais e as ferramentas necessárias para a execução do estudo, incluindo o tratamento do subconjunto FD001, as estratégias de engenharia de atributos e a configuração das arquiteturas de redes neurais (MLP e LSTM). Uma vez estabelecido o rigor metodológico e o ambiente de validação, o próximo capítulo apresentará e discutirá os resultados obtidos, analisando a evolução do desempenho desde o modelo baseline até a arquitetura final otimizada, bem como a sua aplicação no protótipo desenvolvido.

4 RESULTADOS E DISCUSSÕES

Esta seção apresenta e analisa os principais resultados obtidos ao longo do desenvolvimento experimental deste trabalho. Os resultados são organizados de forma progressiva, iniciando pelo desempenho do modelo *baseline*, evoluindo pelas etapas de experimentação e busca de hiperparâmetros, até chegar ao desempenho final do modelo LSTM proposto. Também são discutidas comparações com trabalhos da literatura e análises críticas dos resultados obtidos.

4.1 BASELINE MLP

O primeiro experimento conduzido neste trabalho consistiu na construção de um modelo *baseline* baseado em uma rede do tipo Multi-Layer Perceptron, inspirado na estrutura proposta por Wang et al. (2023). O objetivo desse *baseline* é estabelecer um ponto de referência inicial para quantificar o ganho proporcionado pelo modelo LSTM desenvolvido posteriormente.

Como pode ser observado no Algoritmo 1, o modelo MLP adotado possui seis camadas ocultas com 25 neurônios cada, utilizando a função de ativação Rectified Linear Unit (ReLU) e o otimizador Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS). As entradas do modelo foram normalizadas com *Min-Max scaling*, organizadas em janelas deslizantes de 30 ciclos e filtradas mediante seleção de atributos por *Random Forest*, mantendo apenas os 10 sensores mais relevantes. Também foi aplicada suavização exponencial com α = 0.3 aos sinais, buscando reduzir flutuações de alta frequência.

Algoritmo 1 Pré-processamento e configuração do modelo MLP *baseline*

Entrada: Sinais sensoriais brutos do subconjunto FD001
Saída: Entradas estruturadas e configuração do MLP

1. Suavização dos sinais:
Aplicar suavização exponencial com α = 0.3 aos sinais sensoriais, ciclo a ciclo por motor, reduzindo oscilações de alta frequência.

2. Seleção de atributos:
Ajustar um modelo *Random Forest* para ranquear a importância dos sensores.
Calcular as importâncias.
Selecionar os 10 sensores mais relevantes.

3. Normalização:
Aplicar *Min–Max scaling* apenas aos 10 sensores selecionados.

4. Estruturação das entradas:
Organizar os dados em janelas deslizantes de 30 ciclos por unidade.

5. Configuração do MLP:
Definir um MLP com 6 camadas ocultas e 25 neurônios por camada.
Usar função ReLU e o otimizador L-BFGS.

A avaliação foi realizada no conjunto de teste FD001, utilizando todas as janelas deslizantes geradas e os dados do sensor_4, sensor_9, sensor_21, sensor_11, sensor_14, sensor_12, sensor_7, sensor_13, sensor_8 e sensor_15.

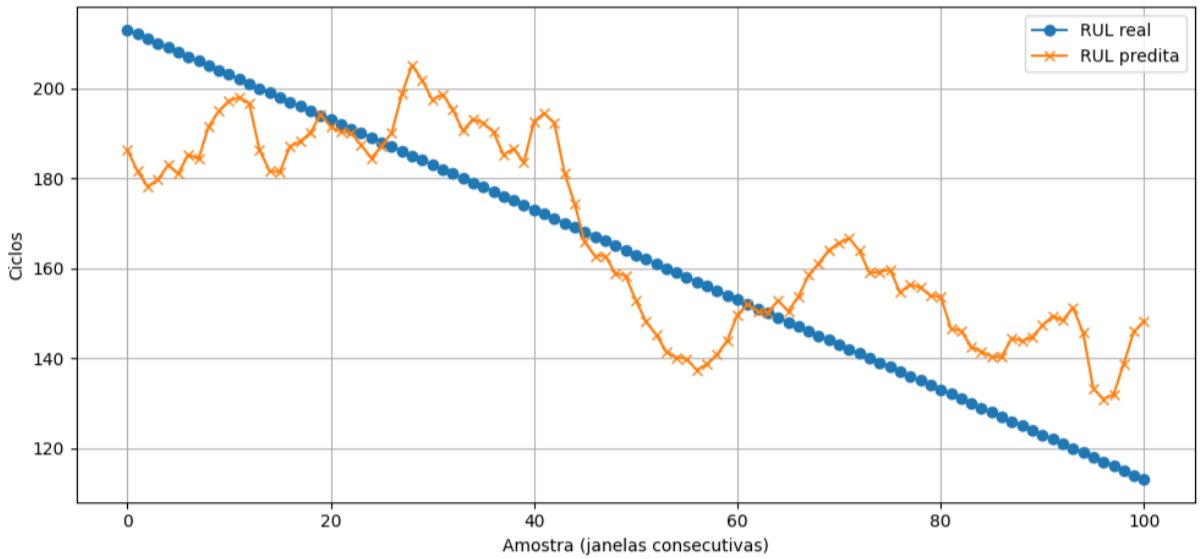
Os resultados obtidos foram:

a) MAE = 26,70 ciclos;

b) RMSE = 36,35 ciclos.

A Figura 5 ilustra a comparação entre os valores reais e preditos da RUL para todas as janelas geradas a partir do motor 23. Observa-se que o modelo apresenta alta variabilidade nas estimativas e não acompanha de forma consistente a tendência de degradação do motor ao longo do tempo, especialmente em regiões onde a RUL deveria decrescer suavemente com o avanço dos ciclos.

Figura 5 – RUL real vs. predita pelo MLP – todas as janelas do motor 23 (FD001).



Fonte: Elaborado pelo autor.

De modo geral, o desempenho do modelo *baseline* revelou-se limitado, apresentando erros relativamente elevados — MAE de 26,70 ciclos e RMSE de 36,35 ciclos. Embora o MLP tenha sido capaz de capturar parcialmente padrões locais após a aplicação da suavização, sua arquitetura do tipo *feedforward* se mostra menos eficaz para modelar dependências temporais, que frequentemente desempenham papel central em tarefas de prognóstico de falhas. Diante disso, tornam-se relevantes arquiteturas que exploram de forma mais explícita a dimensão temporal dos dados, como redes recorrentes LSTM ou Gated Recurrent Unit (GRU), que vêm sendo empregadas com bons resultados na literatura.

Considerando essas limitações do modelo *baseline*, optou-se neste trabalho por investigar o uso de redes LSTM, dada sua capacidade de lidar com sequências temporais e de preservar informações ao longo do tempo. Embora não se trate da única abordagem possível, as LSTM oferecem uma estrutura coerente com a natureza sequencial dos dados de degradação presentes no conjunto FD001. A seção a seguir apresenta o processo de construção, treinamento e avaliação do modelo baseado nessa arquitetura.

4.2 MODELO LSTM

Esta seção apresenta em detalhes o processo de construção, ajustes e resultados obtidos com os modelos LSTM aplicados à previsão da vida útil remanescente. O

desenvolvimento foi conduzido em etapas progressivas, iniciando por uma busca aleatória de hiperparâmetros, seguida da definição de um modelo inicial, melhorias incrementais e, por fim, a configuração do modelo final.

4.2.1 BUSCA ALEATÓRIA DE HIPERPARÂMETROS

Antes de definir a arquitetura inicial da rede LSTM, foi conduzida uma busca aleatória (*Random Search*) para explorar diferentes combinações de hiperparâmetros e identificar configurações promissoras com bom desempenho preditivo. Essa etapa teve como objetivo orientar a escolha de parâmetros iniciais de forma empírica, reduzindo a dependência de ajustes manuais.

A busca foi realizada sobre as seguintes variáveis: número de unidades nas camadas LSTM ({32, 64, 128} na primeira camada e {16, 32, 64} na segunda), taxa de *dropout* ({0.1, 0.2, 0.3}), taxa de aprendizado ({0.01, 0.001, 0.0005, 0.0001}) e tamanho do *batch* ({64, 128, 256}). Ao todo, foram executadas 20 combinações aleatórias, cada uma treinada por até 80 épocas com as estratégias de *early stopping* e *reduce on plateau* para evitar sobreajuste e otimizar o tempo de execução. A lógica geral dessa etapa é sintetizada no Algoritmo 2.

Algoritmo 2 Busca aleatória de hiperparâmetros para LSTM

Entrada: Conjunto de treino e validação (FD001) já pré-processado
Saída: Melhor combinação de hiperparâmetros com base no RMSE

1. Definição dos espaços de busca:
*lstm*1 $\in \{32, 64, 128\}$
*lstm*2 $\in \{16, 32, 64\}$
dropout $\in \{0.1, 0.2, 0.3\}$
learning_rate $\in \{0.01, 0.001, 0.0005, 0.0001\}$
batch_size $\in \{64, 128, 256\}$

Inicializar lista de resultados vazia.

for $i = 1$ **até** 20 **do**
    Amostrar aleatoriamente uma combinação de hiperparâmetros.
    Construir o modelo LSTM com os hiperparâmetros amostrados.
    Treinar o modelo usando:
    - Early Stopping (tolerância = 10)
    - ReduceLROnPlateau (fator = 0.5, tolerância = 5)

    Avaliar na validação (última janela por unidade).
    Calcular MAE e RMSE.
    Armazenar o resultado desta configuração.
end for

Ordenar resultados pelo RMSE crescente.
Retornar a melhor combinação.

A validação foi conduzida com divisão por unidade de motor, conforme estratégia definida na metodologia (Seção 3.6.1), preservando a ordem temporal e garantindo integridade na avaliação. O desempenho das combinações foi avaliado por meio das métricas MAE e RMSE, calculadas sobre a última janela de cada unidade na validação.

Os resultados foram ordenados pelo menor RMSE, e os melhores parâmetros identificados serviram como base para a construção do modelo LSTM inicial. Essa abordagem forneceu uma base empírica sólida para selecionar uma arquitetura inicial coerente com o comportamento sequencial do conjunto FD001. Os detalhes de todas as 20 combinações testadas podem ser visualizados na Tabela 1.

Tabela 1 – Desempenho das 20 combinações testadas na busca aleatória de hiperparâmetros para LSTM - avaliação na validação com última janela por unidade (FD001)

LSTM1	LSTM2	Dropout	LR	Batch	MAE	RMSE
64	64	0,1	0,0100	64	9,49	13,93
128	32	0,1	0,0100	64	9,98	14,08
64	16	0,2	0,0100	256	10,11	13,63
64	32	0,2	0,0010	64	10,99	15,24
32	64	0,1	0,0100	64	11,29	15,09
64	16	0,1	0,0100	128	11,41	15,24
128	64	0,1	0,0010	64	11,53	14,94
128	64	0,2	0,0010	128	11,67	15,52
64	32	0,2	0,0005	64	14,28	17,57
128	16	0,1	0,0005	64	14,69	18,14
64	64	0,3	0,0001	64	18,44	22,53
128	32	0,2	0,0100	64	38,03	42,20
32	64	0,3	0,0100	256	38,14	42,28
128	32	0,3	0,0010	64	38,21	42,34
128	64	0,2	0,0001	128	43,44	48,60
32	64	0,2	0,0001	256	55,09	63,71
32	64	0,1	0,0001	256	55,22	63,86
128	64	0,2	0,0001	256	55,96	64,76
128	16	0,1	0,0001	64	57,32	66,39
128	32	0,1	0,0001	256	65,29	75,40

Fonte: Elaborado pelo autor.

Observa-se que a melhor configuração encontrada foi composta por duas camadas LSTM com 64 unidades cada, *dropout* de 0,1, taxa de aprendizado de 0,01 e *batch size* de 64, resultando em um RMSE de 13,93 ciclos. Esta configuração serviu de base para o modelo LSTM inicial descrito na próxima subseção.

4.2.2 MODELO LSTM INICIAL

Com base na melhor configuração identificada pela busca aleatória de hiperparâmetros, foi construído o primeiro modelo LSTM completo deste trabalho, doravante denominado modelo LSTM inicial. Esta etapa marca a transição entre a experimentação exploratória e o início da construção incremental da arquitetura final.

A arquitetura escolhida foi composta por duas camadas LSTM com 64 unidades cada, seguidas por uma camada densa com um único neurônio para regressão. Foi aplicada uma taxa de *dropout* de 0,1 entre as camadas LSTM, com o objetivo de mitigar o risco

de sobreajuste. A ativação final foi linear, e a função de perda adotada foi o Mean Squared Error (MSE). O otimizador selecionado foi o Adam com taxa de aprendizado inicial de 0,01.

Os dados foram submetidos a um rigoroso processo de pré-processamento. Primeiramente, sensores e configurações com valores constantes foram removidos, uma vez que não forneciam informações discriminativas como pode ser observado nas figuras 6a, 6b, 7a, 7b, 8a, 8b, 8c.

Figura 6 – 'setting_3' e 's1' removidos por apresentarem valores constantes no conjunto FD001.

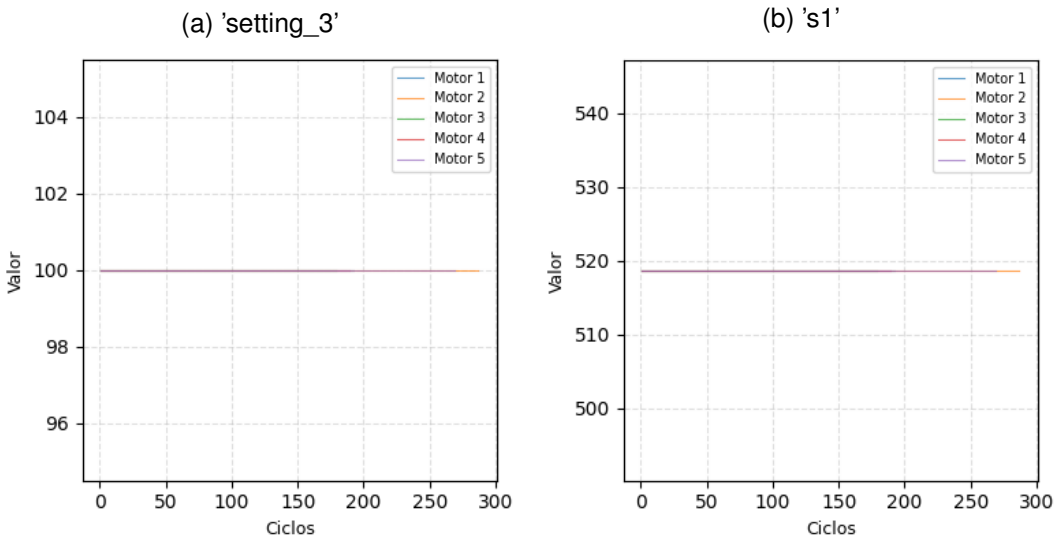


Figura 7 – 's5' e 's10' removidos por apresentarem valores constantes no conjunto FD001.

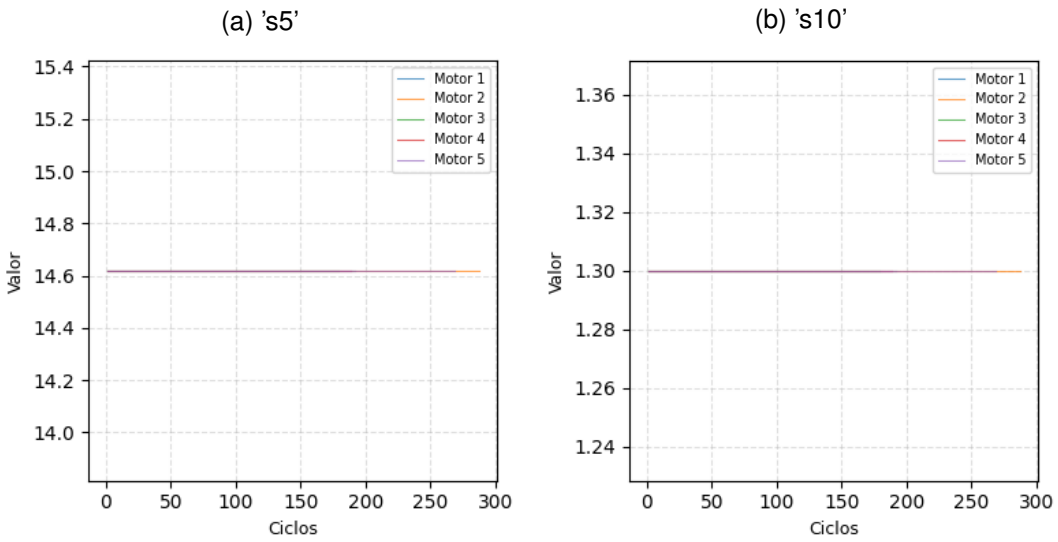
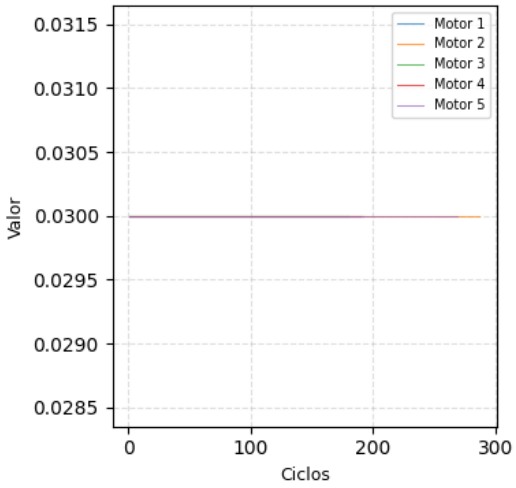
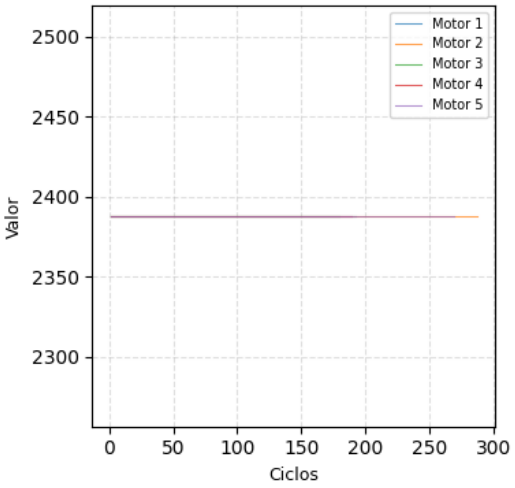


Figura 8 – ’s16’, ’s18’ e ’s19’ removidos por apresentarem valores constantes no conjunto FD001.

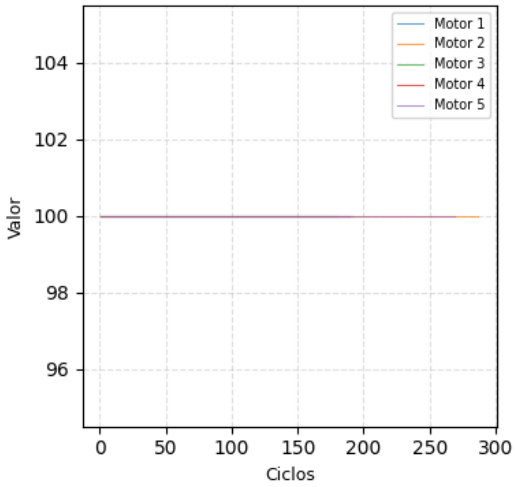
(a) ’s16’



(b) ’s18’



(c) ’s19’



Fonte: Elaborado pelo autor.

Em seguida, os dados foram normalizados individualmente por unidade, utilizando *Min-Max scaling* aplicado com base exclusivamente no conjunto de treino de cada motor, evitando vazamento de informação. A modelagem temporal foi conduzida por meio de janelas deslizantes de 30 ciclos, e o valor alvo de RUL foi limitado a um teto de 125 ciclos para evitar a saturação dos valores iniciais. O Algoritmo 3 exibe a lógica por trás desses procedimentos.

Algoritmo 3 Normalização, limitação de RUL e criação de janelas temporais

Entrada: Dados brutos com RUL calculado
Saída: Sequências temporais normalizadas com RUL limitado

for all unidade u no conjunto de treino **do**
    Ajustar *Min-Max scaler* utilizando apenas os dados de u
    Aplicar o scaler correspondente em u e armazenar para uso futuro
end for
for all unidade u no conjunto de teste **do**
    Aplicar o scaler previamente ajustado para a unidade u
end for
for all registro i nos dados **do**
    Aplicar limite superior de RUL: $RUL_i \leftarrow \min(RUL_i,\ 125)$
end for
for all unidade u **do**
    Deslizar uma janela temporal de tamanho 30 ciclos sobre os dados normalizados
    Armazenar a sequência como entrada e o RUL do último ciclo da janela como alvo
end for

A divisão entre treinamento e validação foi realizada conforme descrito na Seção 3.6.1, preservando a separação por unidade de motor e a ordem cronológica dos dados, o que assegura uma avaliação realista e sem vazamentos de informação.

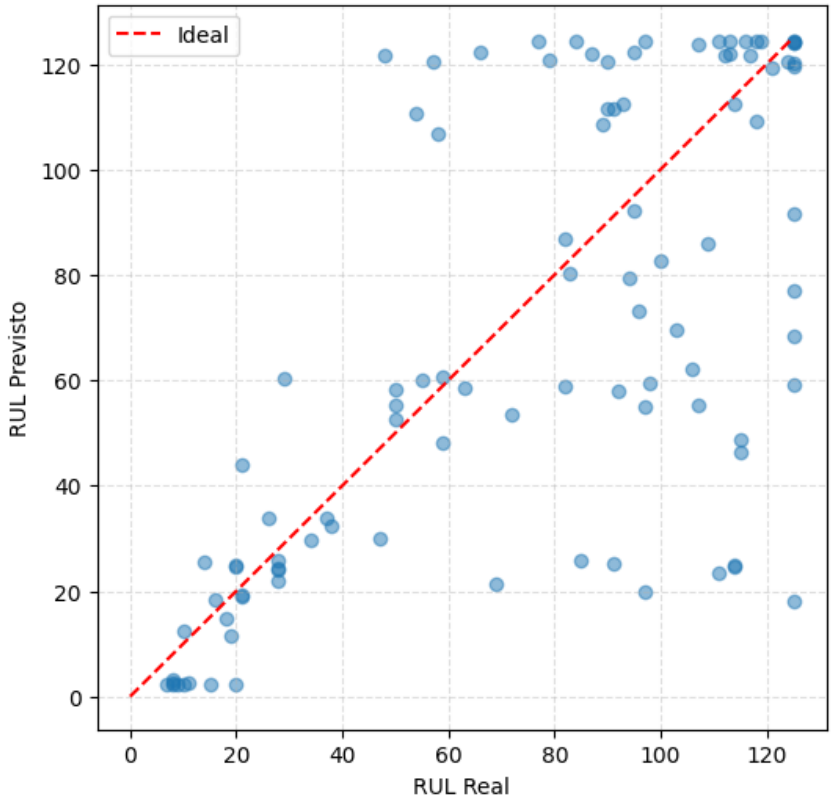
O desempenho quantitativo do modelo LSTM inicial, avaliado com base apenas na última janela de cada unidade do conjunto de teste, resultou em:

a) **MAE:** 23,38 ciclos;

b) **RMSE:** 34,23 ciclos.

Esses resultados demonstram uma pequena melhora em relação a abordagem inicial com redes MLP. No entanto, é possível observar na Figura 9 um nível considerável de erro, especialmente em motores com padrões de degradação abruptos, o que evidencia a necessidade de ajustes adicionais. Essas falhas ocorrem, sobretudo, em trechos em que o padrão de degradação do motor muda subitamente ou não segue uma tendência suave, o que sugere que a arquitetura atual ainda não captura bem variações locais abruptas.

Figura 9 – RUL real vs. predita pelo modelo LSTM inicial – última janela por motor (FD001).



Fonte: Elaborado pelo autor.

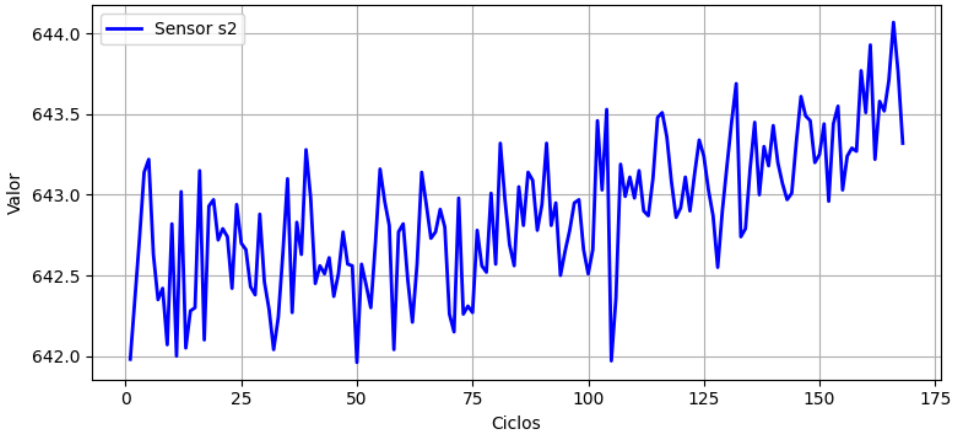
As próximas subseções descrevem os aprimoramentos incrementais realizados na arquitetura LSTM visando maior robustez e acurácia nas previsões.

4.2.2.1 Engenharia de Atributos

Conforme discutido na Seção 2.6.4, com o objetivo de enriquecer a representação dos dados e potencializar a capacidade preditiva da rede LSTM, foi aplicada uma etapa de engenharia de atributos aos sensores após a remoção das variáveis constantes. Esta técnica visa capturar informações dinâmicas da série temporal, como variação local e variabilidade, que nem sempre são evidentes nos valores brutos

A Figura 10 apresenta a série temporal do sensor s2 para o motor 23, exibindo seus valores originais.

Figura 10 – Série temporal original do sensor s2 para o motor 23 (FD001), sem aplicação de engenharia de atributos.



Fonte: Elaborado pelo autor.

A cada sensor foram adicionados três atributos derivados: a média móvel (mean5) e o desvio padrão (std5), ambos calculados sobre uma janela de cinco ciclos, e a inclinação instantânea (slope), calculada como a diferença entre ciclos consecutivos:

a) **Média móvel** (`mean5`): suaviza flutuações e representa o nível local do sinal;

b) **Desvio padrão** (`std5`): quantifica a variabilidade local;

c) **Inclinação** (`slope`): representa a variação imediata do sinal, calculada como a diferença entre ciclos consecutivos.

O processo foi aplicado individualmente a cada unidade, respeitando a estrutura sequencial de cada motor. A Tabela 2 resume os atributos resultantes:

Tabela 2 – Resumo dos atributos derivados gerados para cada sensor

Tipo de Atributo	**Descrição**
Original	Valor bruto do sensor
`mean5`	Média móvel sobre os 5 ciclos anteriores
`std5`	Desvio padrão sobre os 5 ciclos anteriores
`slope`	Diferença entre ciclos consecutivos

Fonte: Elaborado pelo autor.

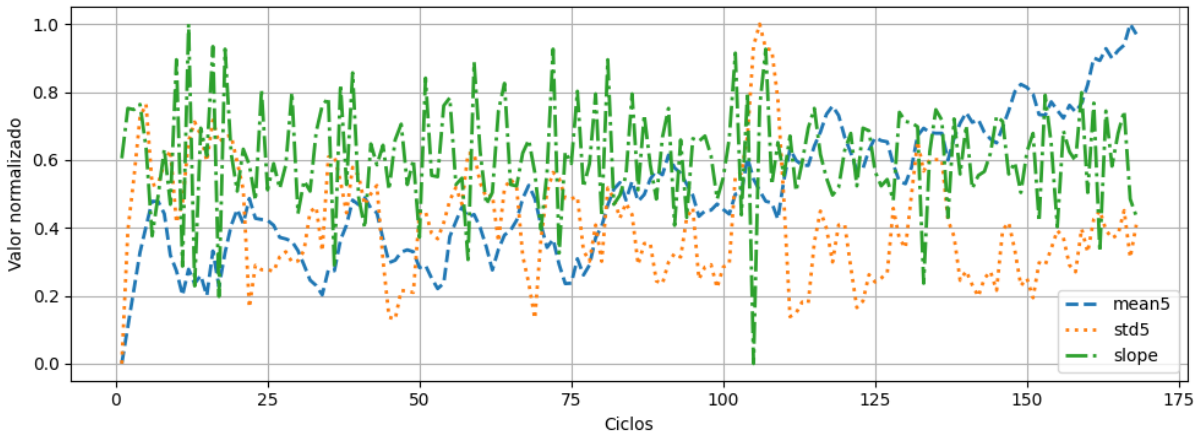
O Algoritmo 4 a seguir descreve o procedimento aplicado:

Algoritmo 4 Geração de atributos derivados (mean5, std5 com janela 5; slope com diferença entre ciclos)

Entrada: Matriz X de sensores por unidade, com T ciclos
Saída: Matriz estendida com atributos `mean5`, `std5` e `slope`
for all unidade u em X **do**
    for all sensor s **do**
        for $t = 5$ **até** T **do**
            $window \leftarrow s[t-5:t]$ ▷ Valores dos últimos 5 ciclos
            $mean5[t] \leftarrow \text{mean}(window)$
            $std5[t] \leftarrow \text{std}(window)$
            $slope[t] \leftarrow s[t] - s[t-1]$ ▷ Variação entre ciclos consecutivos
        end for
    end for
end for
return Matriz X com colunas adicionais para cada atributo derivado

A Figura 11 mostra os três atributos derivados desse sensor devidamente normalizados entre 0 e 1 para facilitar a comparação visual. Ressalta-se que a normalização dos atributos será discutida na seção seguinte.

Figura 11 – Atributos derivados do sensor s2 do motor 23: média móvel (mean5) e desvio padrão (std5), ambos calculados sobre janela de 5 ciclos, e inclinação imediata (slope), obtida pela diferença entre ciclos consecutivos.



Fonte: Elaborado pelo autor.

A etapa seguinte envolveu a padronização do processo de normalização, que anteriormente era realizada por unidade, passando a ser aplicada de forma global por atributo, conforme descrito a seguir.

4.2.2.2 Normalização Global por Atributo

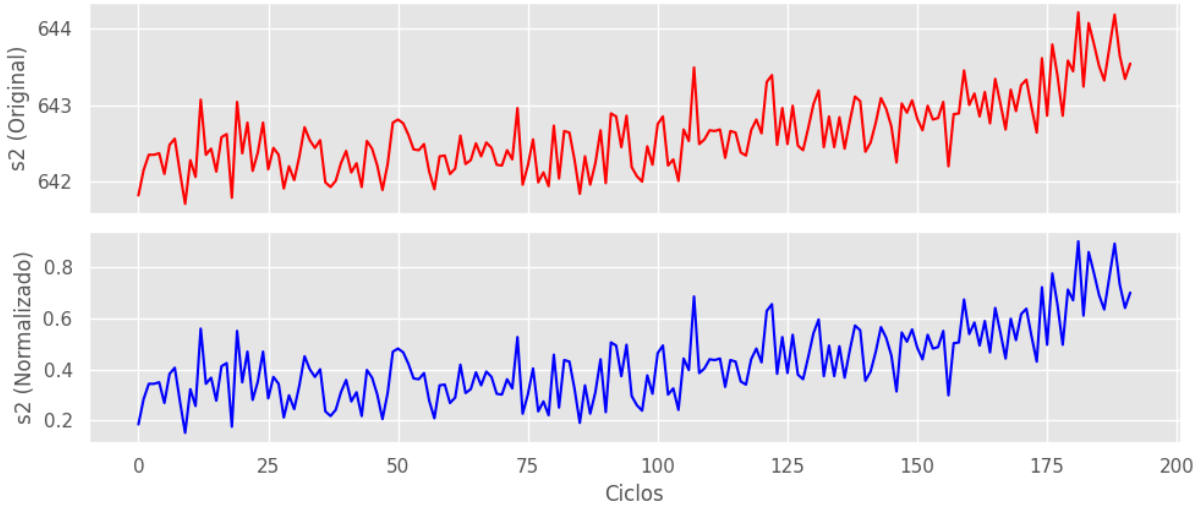
Durante os testes iniciais, foi adotada a normalização por unidade, na qual os valores de sensores e atributos derivados eram reescalonados com base nos mínimos e máximos de cada motor individualmente. Embora esse método garanta que os dados de cada unidade fiquem em uma mesma faixa, ele também elimina variações legítimas entre motores — o que pode dificultar a detecção de padrões generalizáveis de degradação pela rede LSTM.

Com o objetivo de preservar as diferenças reais entre unidades e fortalecer a capacidade do modelo em aprender comportamentos típicos de motores distintos, foi implementada a normalização global por atributo. Nessa abordagem, o `MinMaxScaler` foi ajustado com base na distribuição conjunta de todos os motores do conjunto de treino, garantindo que todos os sensores e atributos derivados sejam reescalonados para a faixa $[0, 1]$, sem perder suas relações relativas.

Essa estratégia substituiu a normalização por unidade utilizada nos experimentos iniciais, preservando diferenças entre motores e fortalecendo a capacidade de generalização do modelo.

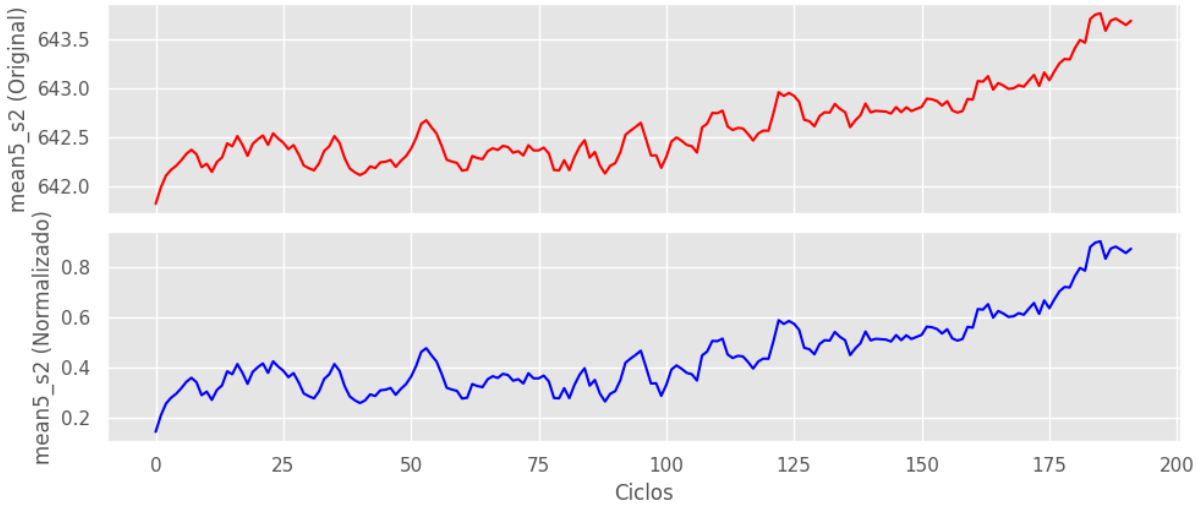
A Figura 12 ilustra visualmente os efeitos dessa transformação, utilizando como exemplo o sensor `s2`. Na Figura 13 pode-se observar a média móvel (`mean5`), na Figura 14 o desvio padrão local (`std5`) e na Figura 15 a inclinação (`slope`). Os gráficos comparam a distribuição das variáveis antes e depois da normalização, considerando o motor 1.

Figura 12 – Sensor `s2` antes e depois da normalização



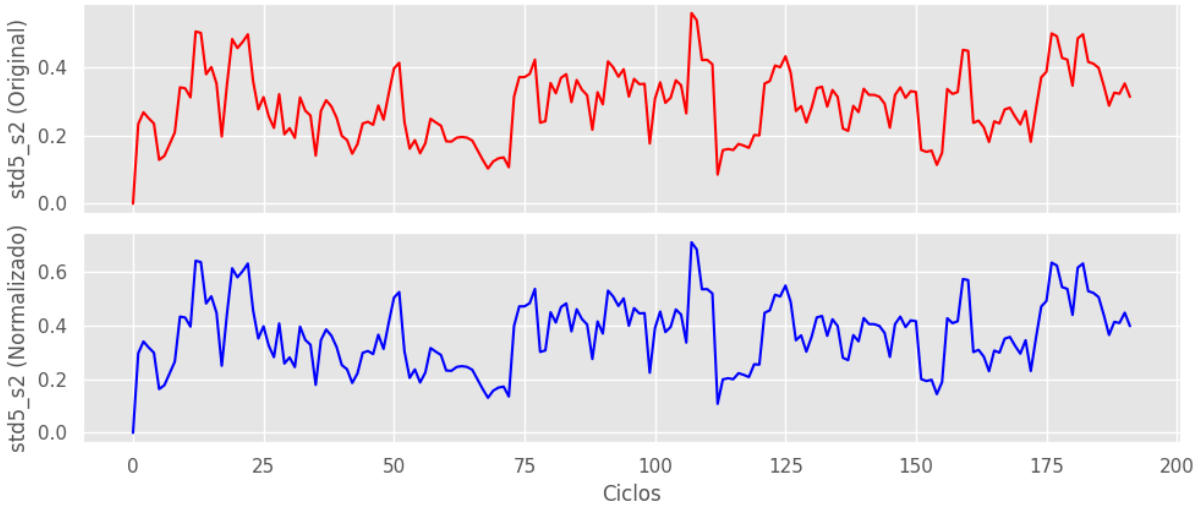
Fonte: Elaborado pelo autor.

Figura 13 – Média móvel (`mean5`) do sensor `s2` antes e depois da normalização



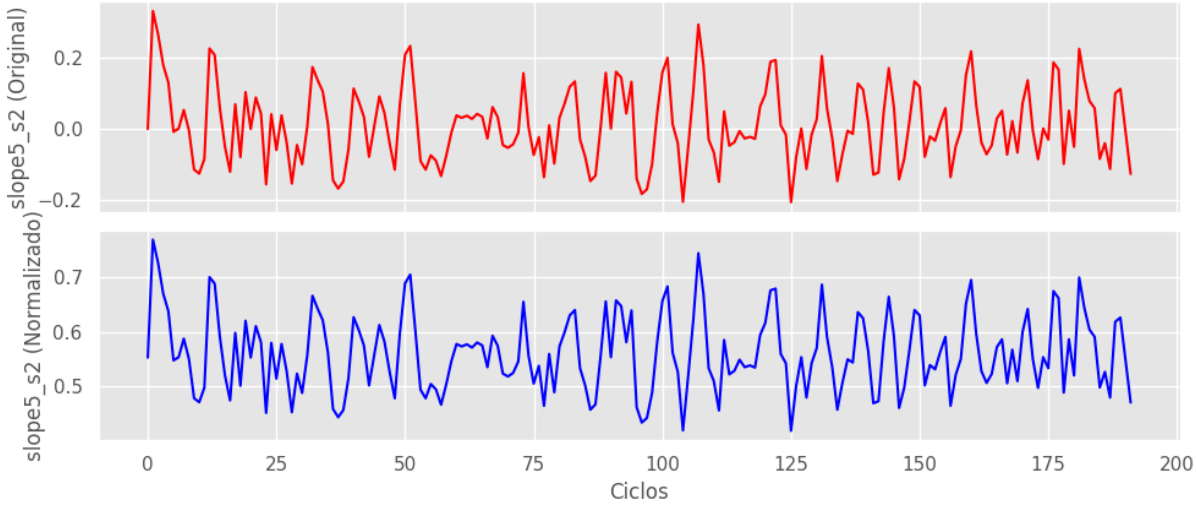
Fonte: Elaborado pelo autor.

Figura 14 – Desvio padrão local (`std5`) do sensor `s2` antes e depois da normalização



Fonte: Elaborado pelo autor.

Figura 15 – Inclinação local (`slope`) do sensor `s2` antes e depois da normalização



Fonte: Elaborado pelo autor.

Observa-se que a normalização preserva a forma dos sinais ao longo do tempo, respeitando as diferenças estruturais entre os motores. Isso é essencial para que a rede LSTM consiga aprender padrões temporais relevantes, independentemente da escala original dos dados.

Essa padronização contribui para uma entrada mais estável e homogênea no modelo, reduzindo o risco de sobreajuste e promovendo maior capacidade de generalização

frente a novos dados.

4.2.2.3 Arquitetura Residual Profunda

Após as melhorias iniciais de pré-processamento e enriquecimento dos dados por meio de engenharia de atributos, investigou-se o impacto do aprofundamento da arquitetura da rede neural. O objetivo nesta etapa foi ampliar a capacidade de extração de padrões temporais complexos, mantendo a estabilidade do treinamento por meio de mecanismos de regularização e conexões residuais.

A rede neural construída nessa fase consistiu em três camadas LSTM empilhadas, sendo as duas primeiras compostas por 128 unidades e configuradas para retornar sequências completas. A terceira camada continha 64 unidades e operava sobre a saída condensada da segunda camada. Para mitigar o risco de *overfitting*, foi aplicado um *dropout* de 0,2 após cada camada LSTM.

Adicionalmente, foi implementada uma conexão residual entre as duas primeiras camadas, o que permitiu preservar informações de baixo nível ao longo da rede e favorecer o fluxo de gradientes durante o treinamento. A saída da última camada LSTM era conectada a uma camada densa com uma única unidade e ativação ReLU, garantindo que os valores previstos para a vida útil remanescente fossem não negativos, em conformidade com a natureza não negativa da RUL.

A arquitetura foi escolhida como um ponto de partida por representar um equilíbrio inicial entre profundidade e custo computacional. Com base nessa arquitetura inicial, foi conduzido um processo de busca aleatória de hiperparâmetros com o objetivo de validar a eficácia da configuração adotada ou identificar arquiteturas alternativas com desempenho superior.

Arquitetura da rede:

a) **Entrada:** sequências de 30 ciclos com *n* atributos (sensores originais + derivados);

b) **Camada 1:** `LSTM(128)` com `return_sequences=True` + `Dropout(0.2)`;

c) **Camada 2:** `LSTM(128)` com `return_sequences=True` + `Dropout(0.2)`;

d) **Residual:** soma da saída da Camada 1 com a saída da Camada 2;

e) **Camada 3:** `LSTM(64)` + `Dropout(0.2)`;

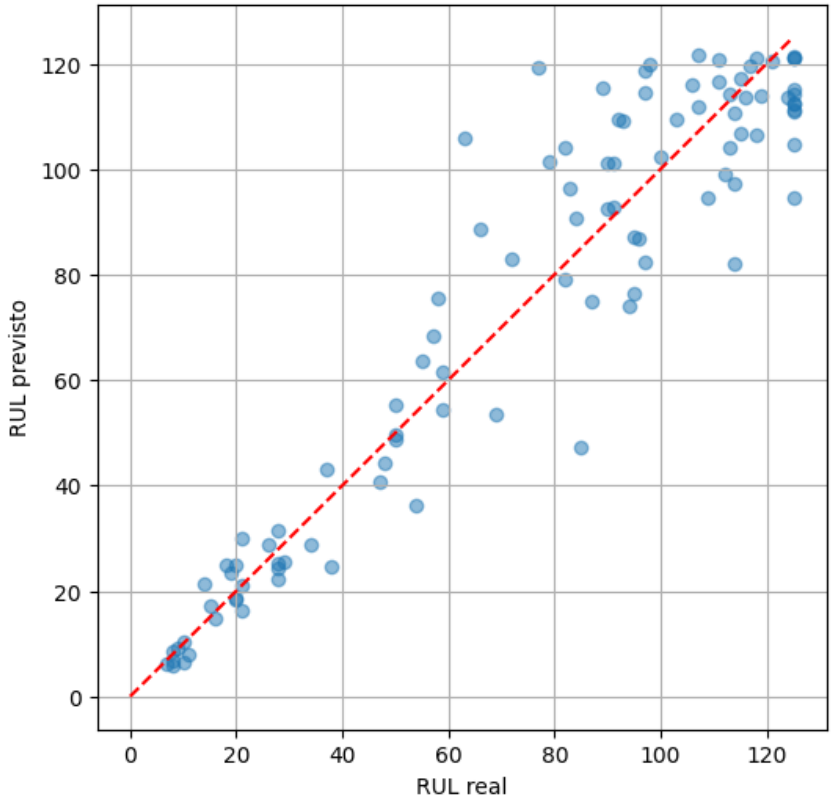
f) **Saída:** `Dense(1, activation="ReLU")`.

O modelo foi treinado com o otimizador Adam, taxa de aprendizado inicial de 10^{-3}, e função de perda MSE. Foi empregada a estratégia de parada antecipada com tolerância de 15 épocas e redução da taxa de aprendizado quando a validação estagnava. A validação seguiu a estratégia por unidade já descrita na Seção 3.6.1, com dados de motores distintos daqueles usados no treinamento.

Ao ser avaliado sobre a última janela de cada motor do conjunto de teste, o modelo apresentou um desempenho expressivo, com MAE de 9,74 ciclos e RMSE de 13,28 ciclos, representando um avanço expressivo em relação aos modelos anteriores, tanto em precisão quanto em robustez preditiva.

A Figura 16 apresenta o gráfico de dispersão entre os valores reais e previstos de RUL para os motores do conjunto de teste. Observa-se uma concentração de pontos ao longo da diagonal ideal, indicando boa capacidade preditiva do modelo.

Figura 16 – Dispersão entre os valores reais e previstos de RUL para os motores do conjunto de teste.



Fonte: Elaborado pelo autor.

Com base nos resultados promissores obtidos pela arquitetura residual profunda, decidiu-se seguir com investigação por meio do ajuste fino de hiperparâmetros. A robustez demonstrada pelo modelo serviu como ponto de partida sólido para essa nova etapa, cujo objetivo foi identificar combinações ainda mais eficazes de parâmetros que pudessem melhorar a capacidade de generalização da rede. O foco passou a ser a exploração sistemática de variações nos principais hiperparâmetros da arquitetura definida, visando potencializar seu desempenho sem comprometer a estabilidade do treinamento.

4.2.2.4 Modelo Final – Ajuste Fino de Hiperparâmetros

Após consolidar a arquitetura LSTM residual profunda, foram exploradas variações nos hiperparâmetros do modelo com o intuito de refinar ainda mais seu desempenho preditivo. O objetivo desta etapa foi investigar como pequenas alterações em parâmetros como número de unidades LSTM, taxa de *dropout*, taxa de aprendizado e tamanho do lote poderiam impactar a acurácia da rede.

Para isso, foi realizada uma busca aleatória (*random search*) com oito combinações distintas dos seguintes hiperparâmetros:

a) `lstm_units`: {96, 128, 160}

b) `dropout`: {0.1, 0.2, 0.3}

c) `learning rate`: {0.001, 0.0007, 0.0005}

d) `batch size`: {32, 64, 128}

A arquitetura da rede foi mantida fixa, conforme descrita na Seção 4.2.2.3, com três camadas LSTM empilhadas (sendo as duas primeiras com retorno de sequência e conexão residual), seguida de uma camada densa com ativação ReLU. O treinamento manteve a estratégia de divisão por unidade descrita na Seção 3.6.1, e a validação foi feita exclusivamente sobre a última janela de cada motor.

A Tabela 3 apresenta os resultados obtidos nas oito combinações avaliadas.

Tabela 3 – Resultados da busca aleatória de hiperparâmetros (validação).

Unidades LSTM	Dropout	LR	Batch Size	MAE	RMSE
96	0.1	0.0010	32	9.24	**12.82**
160	0.2	0.0010	32	9.37	13.08
128	0.3	0.0007	32	9.33	13.10
128	0.3	0.0007	32	9.33	13.11
160	0.1	0.0010	64	9.53	13.24
96	0.1	0.0007	64	9.39	13.29
160	0.1	0.0010	128	9.68	13.43
128	0.3	0.0010	128	38.13	42.28

Fonte: Elaborado pelo autor.

A melhor configuração foi composta por 96 unidades LSTM, *dropout* de 0,1, taxa de aprendizado de 0,001 e batch size de 32, resultando em MAE = 9,24 e RMSE = 12,82 ciclos. Essa combinação foi utilizada no treinamento do modelo final.

É importante destacar que, apesar da arquitetura ter sido mantida constante, observou-se uma melhoria consistente em relação ao modelo anterior (RMSE = 13,28), evidenciando o impacto dos hiperparâmetros no desempenho final.

Esse desempenho, com RMSE inferior a 13 ciclos, posiciona o modelo entre os melhores resultados reportados na literatura para o subconjunto FD001, com a vantagem adicional de utilizar uma arquitetura relativamente compacta e de fácil generalização. Esses resultados reforçam a eficácia da arquitetura LSTM residual com engenharia de atributos e normalização global, indicando que ajustes finos de hiperparâmetros são determinantes para alcançar predições mais precisas da vida útil remanescente.

4.2.2.5 Resultados Finais

Com base na melhor combinação de hiperparâmetros identificada na Seção 4.2.2.4, foi realizado o treinamento do modelo final utilizando a arquitetura LSTM residual profunda descrita anteriormente. O modelo foi treinado conforme a divisão por unidade previamente estabelecida na Seção 3.6.1.

O modelo final foi composto pelas seguintes configurações:

a) **Arquitetura:** LSTM(96) $\rightarrow$ LSTM(96) com conexão residual $\rightarrow$ LSTM(48) $\rightarrow$ Dense(1, ReLU);

b) **Dropout:** 0,1 em todas as camadas LSTM;

c) **Taxa de aprendizado:** 0,001;

d) **Batch size:** 32;

e) **Função de perda:** Erro quadrático médio (MSE);

f) **Otimizador:** Adam;

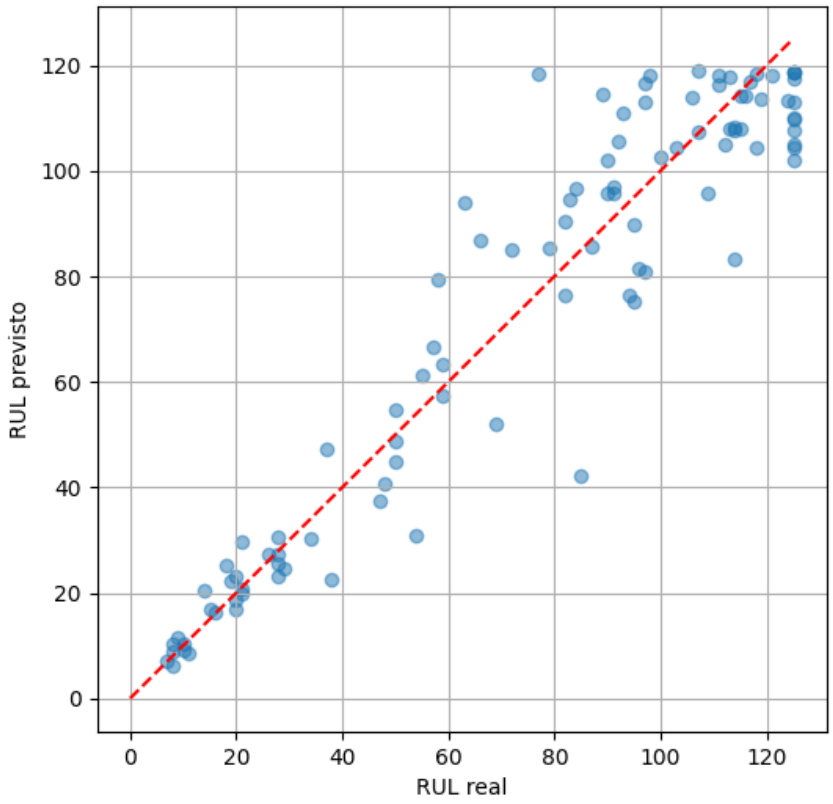
g) **Parada antecipada:** tolerância de 15 épocas, com redução da taxa de aprendizado automática.

A avaliação foi realizada exclusivamente sobre a última janela de cada motor do conjunto de teste FD001, em consonância com a literatura da área. O modelo obteve um desempenho final de:

MAE = 9,18 ciclos | RMSE = 12,6 ciclos

Esses resultados demonstram um avanço significativo em relação às configurações anteriores, confirmando a eficácia da arquitetura residual combinada com uma cuidadosa seleção de hiperparâmetros. A Figura 17 ilustra a dispersão entre os valores reais e previstos de RUL para os motores do conjunto de teste, evidenciando uma forte aderência à diagonal ideal.

Figura 17 – Dispersão dos valores reais e previstos de RUL no conjunto de teste FD001 com hiperparâmetros otimizados. (última janela).



Fonte: Elaborado pelo autor.

A viabilidade computacional da solução proposta é um aspecto fundamental para a aplicação prática. Para o modelo final (LSTM Residual), o uso de hardware otimizado permitiu que o processo completo de treinamento fosse concluído em 4,46 minutos no ambiente local (utilizando a GPU NVIDIA RTX 2060). Em termos comparativos, a execução do mesmo treinamento no ambiente compartilhado do Google Colab demandou 20,06 minutos. Essa diferença evidencia que a infraestrutura local dedicada proporcionou uma aceleração de aproximadamente 4,5 vezes, otimizando o ciclo de desenvolvimento e validando a eficiência da arquitetura proposta em hardware de alto desempenho.

4.3 BENCHMARKING COM TRABALHOS DA LITERATURA

Nesta seção, os resultados do modelo desenvolvido são comparados com trabalhos representativos da literatura que também avaliaram suas abordagens no subconjunto FD001 do C-MAPSS. A Tabela 4 consolida os valores de RMSE e Pontuação S reportados nos estudos analisados e do método proposto.

Tabela 4 – Resultados de RMSE (FD001) dos trabalhos analisados na literatura e do método proposto.

Autor (Ano)	**Método**	**RMSE**	**Pontuação S**
Jayasinghe et al. (2019)	Temporal CNN + LSTM	23,57	1220,00
Wang et al. (2023)	MLP	17,14	-
Hsu e Jiang (2018)	LSTM	16,74	388,68
Berghout et al. (2020)	Adaptive OSELM	46,03	-
Al-Dulaimi et al. (2019)	Hybrid Deep Neural Network (HDNN)	13,02	245,00
Elsheikh, Yacout e Ouali (2019)	Bidirectional Handshaking LSTM	13,74	**199,74**
Babu, Zhao e Li (2016)	Deep CNN	18,45	1286,70
Zheng et al. (2018)	TW-ELM	13,78	267,31
Peng et al. (2022)	Spatio-Temporal Attention Network	**11,98**	312,55
Método proposto	**LSTM residual profunda**	12,60	259,62

Fonte: Elaborado pelo autor.

Os resultados mostram que o método proposto apresenta desempenho competitivo, alcançando RMSE de 12,60 ciclos e Pontuação S de 259.62, valor próximo aos melhores modelos reportados na literatura para o FD001. Isso evidencia a eficácia da arquitetura residual empregada.

4.4 SÍNTESE FINAL DOS RESULTADOS

Os experimentos conduzidos ao longo deste capítulo demonstraram uma evolução consistente no desempenho dos modelos aplicados à previsão da RUL no subconjunto FD001. Após a análise do modelo *baseline* e das diferentes configurações de redes LSTM, verificou-se que a combinação entre engenharia de atributos, normalização global por feature e arquitetura LSTM residual profunda resultou no melhor desempenho, alcançando RMSE de 12,60 ciclos e Pontuação S de 259,62. A comparação com trabalhos representativos da literatura indica que o método proposto apresenta desempenho competitivo, situando-se entre os melhores resultados já reportados para o conjunto FD001. Esses achados consolidam a abordagem desenvolvida como uma solução eficaz e robusta para tarefas de prognóstico baseadas em séries temporais.

4.5 DISPONIBILIZAÇÃO DO CÓDIGO

Com o objetivo de garantir transparência, reprodutibilidade e facilidade de consulta, todo o código desenvolvido neste trabalho foi disponibilizado em um repositório público no GitHub (2025). A organização do repositório é simples e reflete diretamente o fluxo de desenvolvimento da solução:

a) /app/ – contém o protótipo Web desenvolvido em Streamlit, responsável por carregar novos arquivos, aplicar o pré-processamento e executar a predição da RUL;

b) /model/ – armazena os artefatos utilizados na inferência, incluindo o modelo final treinado (model.h5) e o normalizador aplicado aos dados (scaler.pkl);

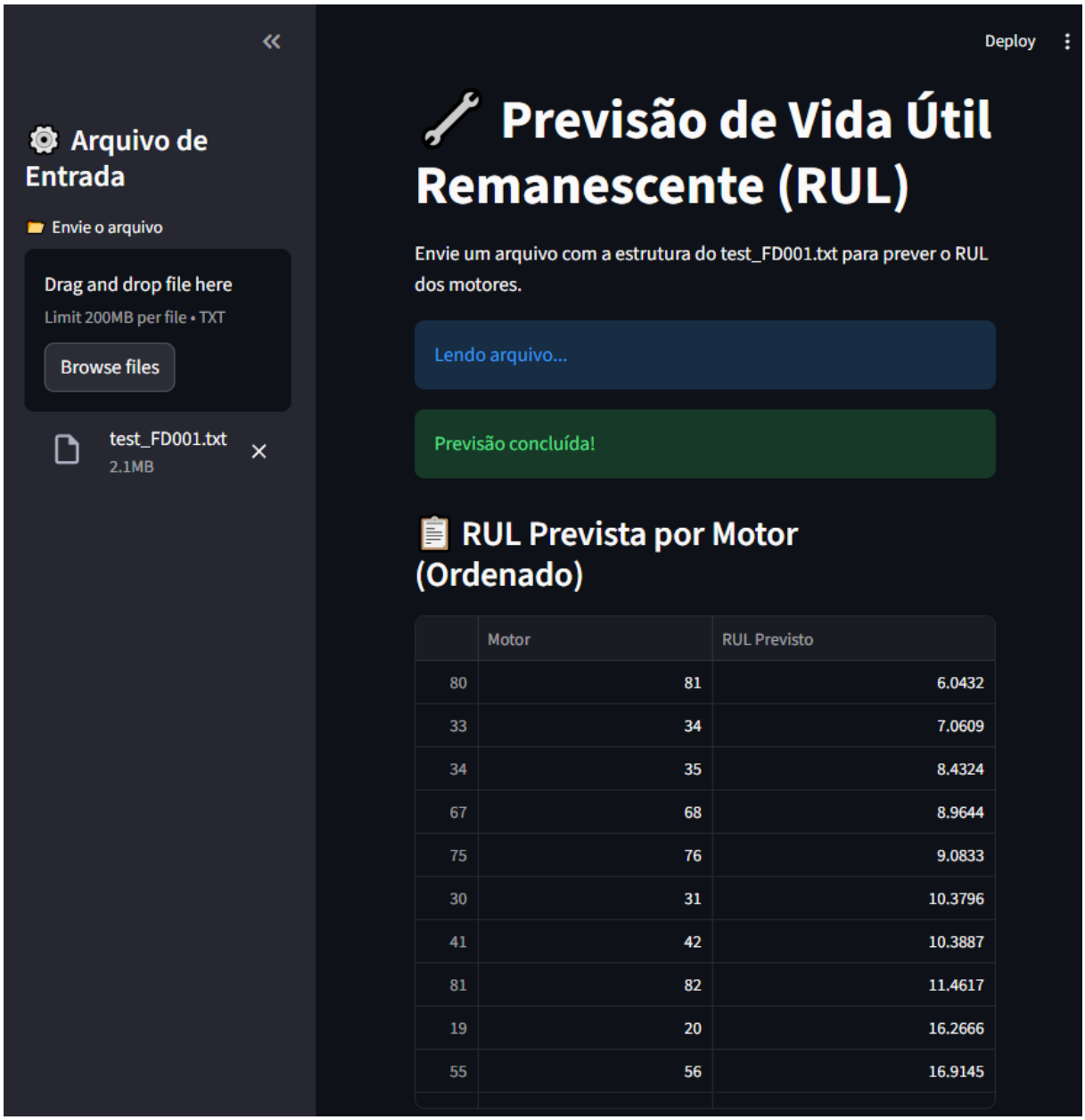
c) /test/ – reúne arquivos utilizados para demonstração e validação da aplicação;

d) /training/ – contém uma descrição do processo adotado no Google Colab para o treinamento do modelo.

A Figura 18 apresenta a interface inicial do protótipo Web, desenvolvida para demonstrar a aplicação prática do modelo no carregamento e análise de novos arquivos do conjunto

FD001. À esquerda, o usuário seleciona o arquivo de entrada; à direita, a aplicação exibe o status do processamento e apresenta a tabela com a RUL prevista para cada motor, ordenada de forma a destacar as unidades em pior condição.

Figura 18 – Interface inicial do protótipo Web, exibindo o upload do arquivo FD001 e a tabela com a RUL prevista para cada motor.

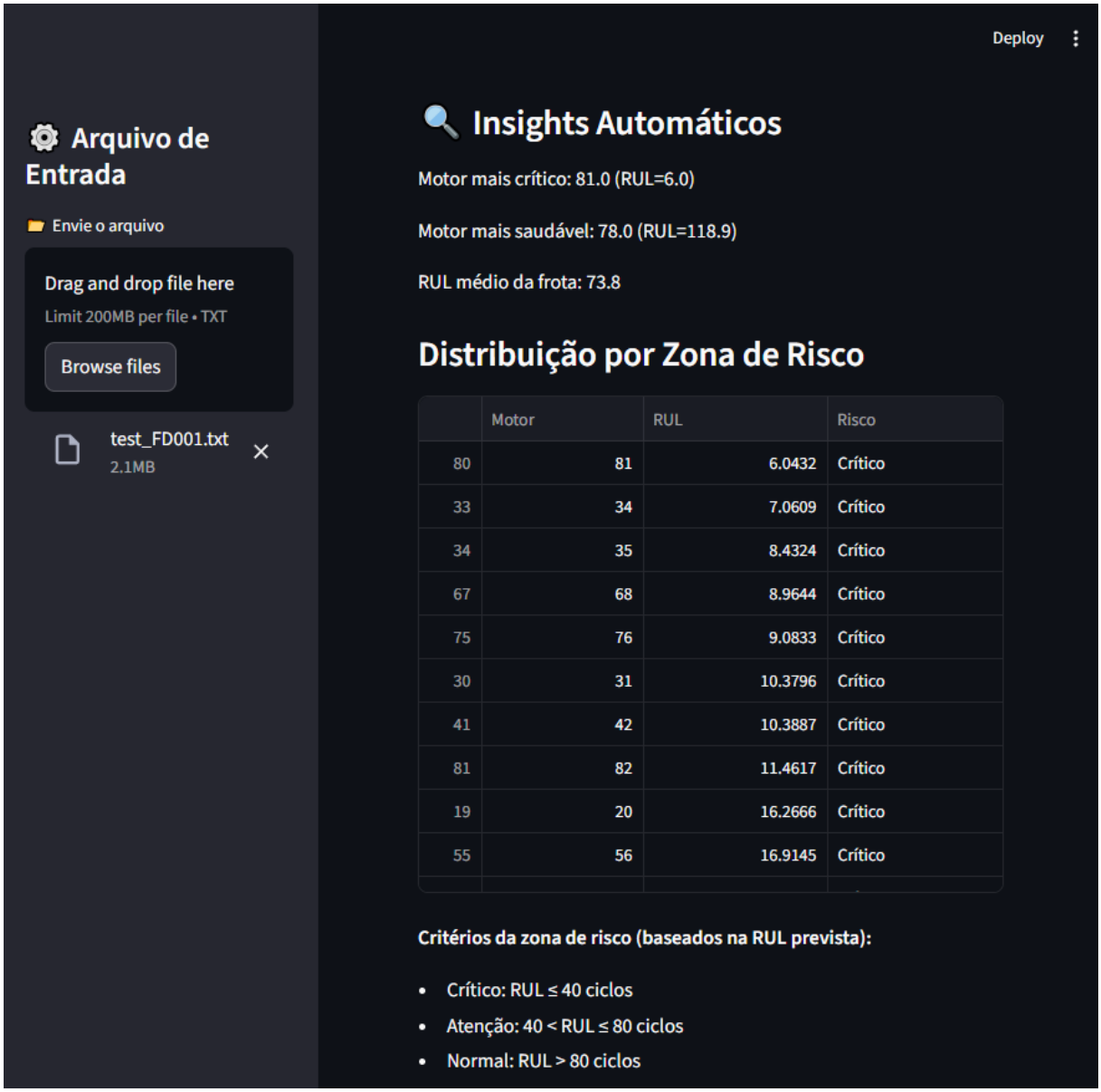


Fonte: Elaborado pelo autor.

A Figura 19 apresenta a seção de análise automática do protótipo. Após o upload do arquivo, a aplicação identifica o motor mais crítico, o motor mais saudável e calcula a RUL média da frota. Em seguida, exibe uma tabela classificando cada motor em zonas de risco (Crítico, Atenção ou Normal), juntamente com os critérios adotados

para essa categorização. Essa categorização é uma etapa de classificação pós-processamento, onde o valor contínuo da RUL (obtido pela regressão) é mapeado para um status discreto. Isso traduz a previsão numérica em uma decisão acionável para a manutenção, utilizando os critérios definidos na interface.

Figura 19 – Seção de análise automática do protótipo, com identificação do motor mais crítico, mais saudável e classificação por zonas de risco.



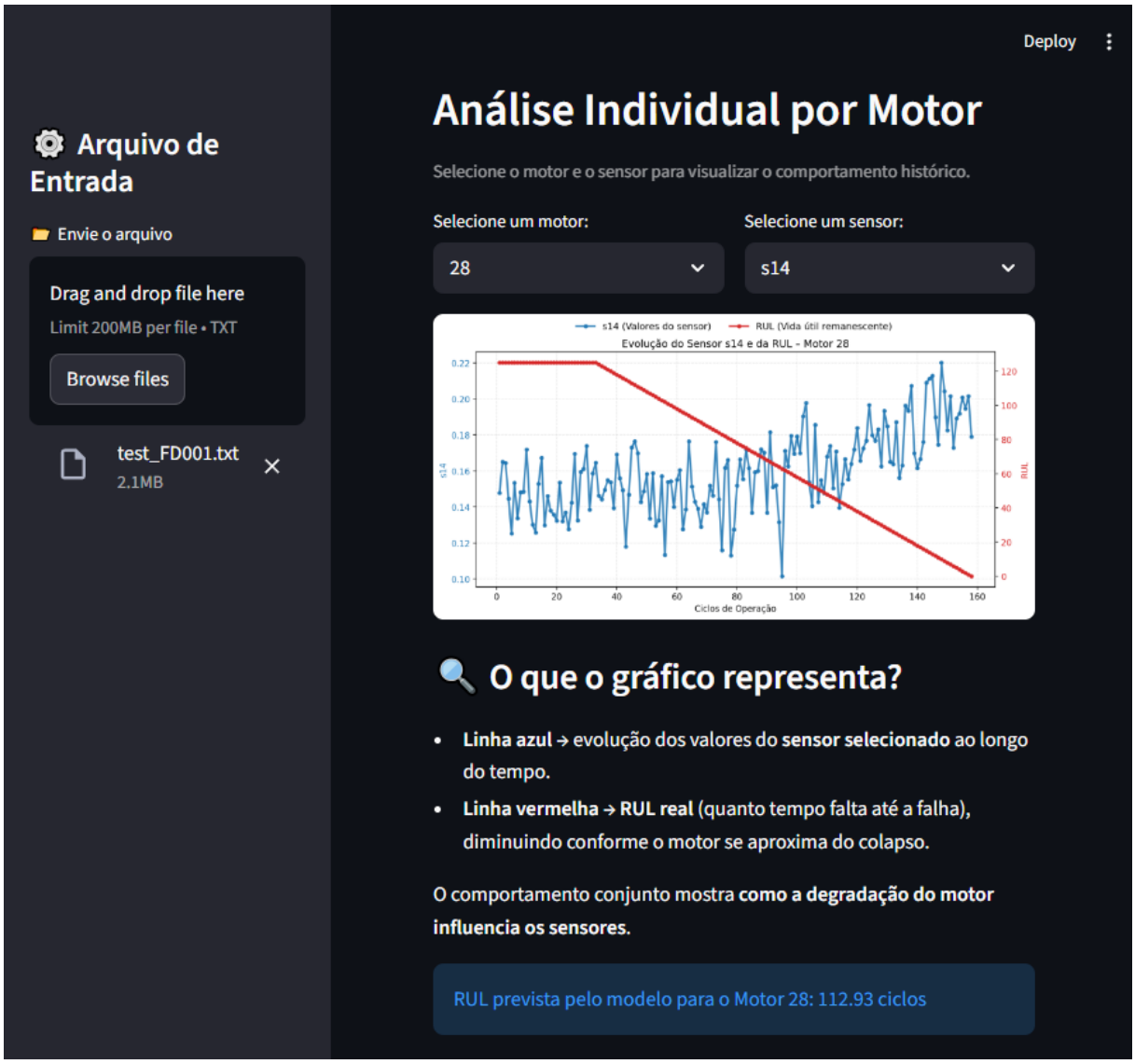
	Motor	RUL	Risco
80	81	6.0432	Crítico
33	34	7.0609	Crítico
34	35	8.4324	Crítico
67	68	8.9644	Crítico
75	76	9.0833	Crítico
30	31	10.3796	Crítico
41	42	10.3887	Crítico
81	82	11.4617	Crítico
19	20	16.2666	Crítico
55	56	16.9145	Crítico

Fonte: Elaborado pelo autor.

A Figura 20 mostra a seção de análise individual do protótipo, na qual o usuário seleciona um motor e um sensor para visualizar seu comportamento histórico. O gráfico apresenta a evolução do sensor escolhido (linha azul) ao longo dos ciclos de operação, juntamente com a trajetória da RUL real (linha vermelha), evidenciando

como a degradação do motor se reflete nas leituras dos sensores. Abaixo do gráfico, a interface exibe a RUL prevista pelo modelo para o motor selecionado.

Figura 20 – Seção de análise individual do protótipo, com a evolução do sensor selecionado e da RUL real para o motor escolhido.



Fonte: Elaborado pelo autor.

Os experimentos detalhados neste capítulo demonstraram a eficácia da abordagem proposta, evidenciando como a evolução incremental — passando pela engenharia de atributos, normalização global e arquitetura residual — resultou em um modelo competitivo com a literatura e em um protótipo funcional para inferência. Diante desses achados e da validação prática da solução, o capítulo final apresentará as conclusões gerais do estudo, sintetizando as contribuições alcançadas e apontando perspectivas para trabalhos futuros.

5 CONCLUSÃO

Este trabalho apresentou uma abordagem *data-driven* para o prognóstico da vida útil remanescente de turbinas *turbofan*, utilizando o subconjunto FD001 do repositório C-MAPSS da NASA. O estudo teve como foco o desenvolvimento e a avaliação de modelos de redes neurais, iniciando por um *baseline* MLP e evoluindo para arquiteturas mais adequadas à modelagem temporal, culminando em uma LSTM residual profunda combinada com engenharia de atributos e normalização global por *feature*.

Os resultados obtidos demonstraram uma evolução consistente ao longo do processo experimental. O *baseline* MLP forneceu um ponto de referência inicial, revelando as limitações de arquiteturas *feedforward* para séries temporais de degradação. Em seguida, a aplicação de uma busca aleatória de hiperparâmetros permitiu identificar configurações iniciais promissoras para redes LSTM, que serviram de base para refinamentos posteriores.

As melhorias incrementais tiveram impacto direto no aumento de desempenho. A engenharia de atributos — composta por média móvel, desvio padrão e inclinação local — ampliou a capacidade do modelo em capturar padrões dinâmicos de curto prazo. A adoção da normalização global por *feature* preservou diferenças estruturais entre as unidades, evitando que informações importantes fossem eliminadas pelo pré-processamento. Por fim, a arquitetura LSTM residual profunda proporcionou estabilidade e profundidade ao modelo, favorecendo o fluxo de gradientes e o aprendizado de padrões temporais complexos.

O modelo final alcançou um desempenho expressivo, obtendo RMSE de 12,60 ciclos e MAE de 9,18 ciclos na avaliação pela última janela de cada motor — estratégia alinhada às práticas mais adequadas em prognóstico. Os resultados obtidos mostram que o modelo apresenta desempenho comparável aos melhores valores reportados na literatura analisada para o conjunto FD001, mesmo quando comparado com arquiteturas mais complexas, incluindo CNNs profundas e modelos híbridos. Essa proximidade com o estado da arte reforça a eficácia da abordagem desenvolvida, especialmente conside-

rando sua relativa simplicidade, interpretabilidade e aderência à natureza sequencial dos dados.

De forma geral, os resultados evidenciam que a estimativa antecipada da vida útil remanescente é viável no contexto de turbinas turbofan e pode contribuir significativamente para estratégias de manutenção preditiva. Modelos baseados em aprendizado profundo mostraram-se capazes de identificar irregularidades nos sinais sensoriais e de captar tendências de degradação ao longo do tempo. Esses achados destacam o potencial do prognóstico para reduzir custos, mitigar falhas inesperadas e aumentar a confiabilidade operacional de sistemas industriais.

Por fim, o trabalho desenvolvido oferece uma contribuição relevante para a área de PHM, ao apresentar uma abordagem metodológica clara, reprodutível e alinhada às melhores práticas experimentais. Os resultados obtidos constituem uma base sólida para avanços futuros. Como perspectivas de continuidade, sugere-se a aplicação da arquitetura proposta nos subconjuntos mais complexos do C-MAPSS (FD002, FD003 e FD004), que envolvem múltiplas condições operacionais e modos de falha simultâneos, desafiando a capacidade de generalização do modelo.

Outra linha promissora é a incorporação de mecanismos de atenção (*Attention Mechanisms*) e a investigação de arquiteturas baseadas em *Transformers*, visando aprimorar a captura de dependências de longo prazo e a interpretabilidade das predições. Recomenda-se, ainda, a exploração de abordagens probabilísticas que forneçam intervalos de confiança para a RUL estimada, bem como a aplicação de técnicas de Inteligência Artificial Explicável (XAI) para identificar a contribuição individual dos sensores. Por fim, estudos futuros podem focar na otimização do modelo para execução em dispositivos de borda (*Edge Computing*), visando a implementação embarcada em sistemas reais.

Com o intuito de garantir transparência e possibilitar a continuidade das pesquisas, todo o código desenvolvido, incluindo o modelo final, o pré-processamento e o protótipo Web, foi disponibilizado em um repositório público, permitindo que outros pesquisadores reproduzam os experimentos e explorem novas extensões da abordagem proposta.

REFERÊNCIAS

AL-DULAIMI, A. et al. Hybrid deep neural network model for remaining useful life estimation. In: IEEE. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.], 2019. p. 3872–3876.

ALMEIDA, P. S. D. *Manutenção Mecânica Industrial–Conceitos Básicos e Tecnologia Aplicada*. [S.l.]: Saraiva Educação SA, 2018.

ALOMARI, Y.; ANDÓ, M.; BAPTISTA, M. L. Advancing aircraft engine rul predictions: an interpretable integrated approach of feature engineering and aggregated feature importance. *Scientific Reports*, Nature Publishing Group, v. 13, n. 1, p. 13466, 2023. Disponível em: <https://doi.org/10.1038/s41598-023-40315-1>.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. *NBR 5462: Confiabilidade e mantenabilidade*. Rio de Janeiro, 1994. Norma brasileira.

AUTOR, D. *Polanyi's Paradox and the Shape of Employment Growth*. [S.l.], 2014. (Working Paper Series, 20485). Disponível em: <http://www.nber.org/papers/w20485>.

BABU, G. S.; ZHAO, P.; LI, X.-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. In: *Database Systems for Advanced Applications*. Cham: Springer International Publishing, 2016. p. 214–228. ISBN 978-3-319-32025-0.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, v. 5, p. 157–66, 02 1994.

BENHANIFIA, A. et al. Systematic review of predictive maintenance practices in the manufacturing sector. *Intelligent Systems with Applications*, v. 26, p. 200501, 2025. ISSN 2667-3053. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2667305325000274>.

BERGHOUT, T. et al. Aircraft engines remaining useful life prediction with an improved online sequential extreme learning machine. *Applied Sciences*, MDPI, v. 10, n. 3, p. 1062, 2020.

BIANCHINI, Â. R. Arquitetura de redes neurais para o reconhecimento facial baseado no neocognitron. Universidade Federal de São Carlos, 2001.

BISHOP, C. M.; NASRABADI, N. M. *Pattern recognition and machine learning*. [S.l.]: Springer, 2006. v. 4.

CHAI, T.; DRAXLER, R. R. Root mean square error (rmse) or mean absolute error (mae)? – arguments against avoiding rmse in the literature. *Geoscientific Model Development*, v. 7, n. 3, p. 1247–1250, 2014. Disponível em: <https://gmd.copernicus.org/articles/7/1247/2014/>.

CHAO, M. A. et al. Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, v. 6, n. 1, 2021. ISSN 2306-5729. Disponível em: <https://www.mdpi.com/2306-5729/6/1/5>.

CHAOUB, A. et al. Learning representations with end-to-end models for improved remaining useful life prognostics. In: *Annual Conference of the PHM Society*. [s.n.], 2021. v. 13, n. 1, p. 1–9. Disponível em: <https://arxiv.org/abs/2104.05049>.

CHARNIAK, E.; MCDERMOTT, D. *Introduction to Artificial Intelligence*. Addison-Wesley, 1985. (Addison-Wesley series in computer science and information processing). ISBN 9780201119459. Disponível em: <https://books.google.com.br/books?id=TpJQAAAAMAAJ>.

CHASSAGNON, G. et al. Deep learning: definition and perspectives for thoracic imaging. *European Radiology*, Springer, v. 30, n. 4, p. 2021–2030, 2020. Disponível em: <https://doi.org/10.1007/s00330-019-06564-3>.

CHRIST, M. et al. Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package). *Neurocomputing*, v. 307, p. 72–77, 2018. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231218304843>.

{de Oliveira da Costa}, P. et al. Attention and long short-term memory network for remaining useful lifetime predictions of turbofan engine degradation. *International Journal of Prognostics and Health Management*, PHM Society, v. 10, 2019. ISSN 2153-2648.

EKANEM, U. A.; USORO, M. U.; BARIDAM, D. Maintenance management and organizational performance in selected manufacturing firms, akwa ibom state. *International Journal of Business and Management Review*, v. 10, n. 4, p. 37–59, 2022.

ELSHEIKH, A.; YACOUT, S.; OUALI, M.-S. Bidirectional handshaking lstm for remaining useful life prediction. *Neurocomputing*, Elsevier, v. 323, p. 148–156, 2019.

ELSHERIF, S. M. et al. A deep learning-based prognostic approach for predicting turbofan engine degradation and remaining useful life. *Scientific Reports*, v. 15, p. 26251, 2025. Disponível em: <https://doi.org/10.1038/s41598-025-09155-z>.

FANUCCHI, R. Z.; OLESKOVICZ, M.; BARBOSA, D. Análise da detecção de faltas de alta impedância utilizando redes neurais artificiais com topologias baseadas em perceptron multicamadas e redes rbf. *Simpósio Brasileiro de Automação Inteligente (SBAI), Fortaleza*, 2013.

FREDERICK, D.; DECASTRO, J.; LITT, J. User's guide for the commercial modular aero-propulsion system simulation (c-mapss). *NASA Technical Manuscript*, v. 2007–215026, 01 2007.

GITHUB. *tcc-cmapss-rul-lstm*. 2025. <https://github.com/hencabral/tcc-cmapss-rul-lstm>. Acesso em: 24 nov. 2025.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GREFF, K. et al. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, v. 28, n. 10, p. 2222–2232, 2017. Disponível em: <https://doi.org/10.1109/TNNLS.2016.2582924>.

HAUGELAND, J. *Artificial Intelligence: The Very Idea*. USA: Massachusetts Institute of Technology, 1985. ISBN 0262081539.

HE, K. et al. Deep residual learning for image recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 770–778.

HEIMES, F. O. Recurrent neural networks for remaining useful life estimation. In: *2008 International Conference on Prognostics and Health Management*. [S.l.: s.n.], 2008. p. 1–6.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, v. 9, n. 8, 1997.

HOTA, H.; HANDA, R.; SHRIVAS, A. K. Time series data prediction using sliding window based rbf neural network. *International Journal of Computational Intelligence Research*, v. 13, n. 5, p. 1145–1156, 2017.

HSU, C.-S.; JIANG, J.-R. Remaining useful life estimation using long short-term memory deep learning. In: IEEE. *2018 ieee international conference on applied system invention (icasi)*. [S.l.], 2018. p. 58–61.

ISO13381-1. *ISO 13381-1: Condition monitoring and diagnostics of machines – Prognostics – Part 1: General guidelines*. Geneva, Switzerland, 2015.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine learning and deep learning. *Electronic Markets*, Springer, v. 31, n. 3, p. 685–695, 2021. Published online 8 April 2021. Disponível em: <https://doi.org/10.1007/s12525-021-00475-2>.

JARDINE, A. K.; LIN, D.; BANJEVIC, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, v. 20, n. 7, p. 1483–1510, 2006. ISSN 0888-3270. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0888327005001512>.

JAYASINGHE, L. et al. Temporal convolutional memory networks for remaining useful life estimation of industrial machinery. In: IEEE. *2019 ieee international conference on industrial technology (icit)*. [S.l.], 2019. p. 915–920.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. *Science*, v. 349, n. 6245, p. 255–260, 2015. Disponível em: <https://www.science.org/doi/abs/10.1126/science.aaa8415>.

JOST, I. Aplicação de deep learning em dados refinados para mineração de opiniões. Universidade do Vale do Rio dos Sinos, 2015.

KARDEC, A.; NASCIF, J. *Manutenção - Função estratégica*. 3ª edição. ed. [S.l.]: QualityMark, 2009.

KIM, N.-H.; CHOI, J.-H.; AN, D. *Prognostics and Health Management of Engineering Systems*. Cham, Switzerland: Springer International Publishing, 2017. ISBN 978-3-319-44740-7. Disponível em: <https://doi.org/10.1007/978-3-319-44742-1>.

KURZWEIL, R. *The Age of Intelligent Machines*. Kurzweil Foundation, 1990. ISBN 9780262111218. Disponível em: <https://books.google.com.br/books?id=SI1QAAAAMAAJ>.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015. Disponível em: <https://doi.org/10.1038/nature14539>.

LEE, J. et al. Intelligent prognostics tools and e-maintenance. *Computers in Industry*, v. 57, n. 6, p. 476–489, 2006. ISSN 0166-3615. E-maintenance Special Issue. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0166361506000522>.

LI, C. et al. Small data challenges for intelligent prognostics and health management: a review. *Artificial Intelligence Review*, Springer, 2024. ISSN 0269-2821. Disponível em: <https://doi.org/10.1007/s10462-024-10820-4>.

LI, X.; DING, Q.; SUN, J.-Q. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety*, v. 172, p. 1–11, 2018. ISSN 0951-8320. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0951832017307779>.

LIU, Y.; WEN, J.; WANG, G. A comprehensive overview of remaining useful life prediction: From traditional literature review to scientometric analysis. *Machine Learning with Applications*, v. 21, p. 100704, 2025. ISSN 2666-8270. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666827025000878>.

LU, J. et al. Learning under concept drift: A review. *CoRR*, abs/2004.05785, 2020. Disponível em: <https://arxiv.org/abs/2004.05785>.

LUGER, G. *Inteligência Artificial*. PEARSON BRASIL, 2013. ISBN 9788581435503. Disponível em: <https://books.google.com.br/books?id=UNEKvQEACAAJ>.

MITICI, M. et al. Dynamic predictive maintenance for multiple components using data-driven probabilistic rul prognostics: The case of turbofan engines. *Reliability Engineering & System Safety*, v. 234, p. 109199, 2023. ISSN 0951-8320. Disponível em: <https://www.sciencedirect.com/science/article/pii/S095183202300114X>.

NOR, A. K. B. M.; PEDAPAIT, S. R.; MUHAMMAD, M. Explainable ai (xai) for phm of industrial asset: A state-of-the-art, prisma-compliant systematic review. *arXiv preprint arXiv:2107.03869*, 2021. Preprint; versão de revisão de XAI para PHM. Disponível em: <https://arxiv.org/abs/2107.03869>.

OLAH, C. *Understanding LSTM Networks*. 2015. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Acesso em: 01 dez. 2023.

PASCHOAL, D. et al. Disponibilidade e confiabilidade: Aplicação da gestão da manutenção na busca de maior competitividade. *Revista da Engenharia de Instalações no mar da FSMA*, p. 1–14, 2009.

PATRO, S.; SAHU, K. Normalization: A preprocessing stage. *arXiv preprint arXiv:1503.06462*, 2015. Disponível em: <https://arxiv.org/abs/1503.06462>.

PENG, C. et al. A spatio-temporal attention mechanism based approach for remaining useful life prediction of turbofan engine. *Computational Intelligence and Neuroscience*, v. 2022, n. 1, p. 9707940, 2022. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1155/2022/9707940>.

PHI, M. *Illustrated Guide to LSTMs and GRUs: A step by step explanation*. 2018. <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>. Acesso em: 02 dez. 2023.

POOLE, D.; MACKWORTH, A.; GOEBEL, R. *Computational Intelligence: A Logical Approach*. [S.l.]: Oxford University Press, 1998.

RUSSELL, S. et al. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010. (Prentice Hall series in artificial intelligence). ISBN 9780136042594. Disponível em: <https://books.google.com.br/books?id=8jZBksh-bUMC>.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, v. 3, n. 3, p. 210–229, 1959.

SANTOS, A. M. d. et al. Usando redes neurais artificiais e regressão logística na predição da hepatite a. *Revista Brasileira de Epidemiologia*, SciELO Brasil, v. 8, p. 117–126, 2005.

SAXENA, A. et al. Damage propagation modeling for aircraft engine run-to-failure simulation. In: *Proceedings of the 1st International Conference on Prognostics and Health Management (PHM 2008)*. [S.l.]: IEEE, 2008. p. 1–9.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural Networks*, v. 61, p. 85–117, 2015. ISSN 0893-6080. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0893608014002135>.

SI, X.-S. et al. Remaining useful life estimation – a review on the statistical data driven approaches. *European Journal of Operational Research*, v. 213, n. 1, p. 1–14, 2011. ISSN 0377-2217. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0377221710007903>.

SIKORSKA, J.; HODKIEWICZ, M.; MA, L. Prognostic modelling options for remaining useful life estimation by industry. *Mechanical Systems and Signal Processing*, v. 25, n. 5, p. 1803–1836, 2011. ISSN 0888-3270. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0888327010004218>.

SILVER, D. et al. Mastering the game of go without human knowledge. *Nature*, v. 550, p. 354–359, 10 2017.

SIMEÓN, E. J. A. *Prognóstico de falhas baseado em redes neurais com estados de eco*. Tese (Doutorado) — University of Brasília, Brazil, 2015. Disponível em: <http://repositorio.unb.br/handle/10482/20510>.

SIMON, H. A. Why should machines learn? In: ____. *Machine Learning: An Artificial Intelligence Approach*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1983. p. 25–37. ISBN 978-3-662-12405-5. Disponível em: <https://doi.org/10.1007/978-3-662-12405-5_2>.

SOCIETY, P. Phm08 challenge: Challenge data description. 2016.

VASILACHE, A. et al. Low-power vibration-based predictive maintenance for industry 4.0 using neural networks: A survey. *arXiv preprint arXiv:2408.00516*, 2024. Disponível em: <https://arxiv.org/abs/2408.00516>.

WANG, C. et al. Towards time-series feature engineering in automated machine learning for multi-step-ahead forecasting. *Engineering Proceedings*, v. 18, n. 1, 2022. ISSN 2673-4591. Disponível em: <https://www.mdpi.com/2673-4591/18/1/17>.

WANG, F. et al. A deep-learning method for remaining useful life prediction of power machinery via dual-attention mechanism. *Sensors*, v. 25, n. 2, 2025. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/25/2/497>.

WANG, H. et al. Remaining useful life prediction of aircraft turbofan engine based on random forest feature selection and multi-layer perceptron. *Applied Sciences*, v. 13, n. 12, 2023. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/13/12/7186>.

WILLMOTT, C. J.; MATSUURA, K. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, v. 30, n. 1, p. 79–82, 2005.

WU, F. et al. Remaining useful life prediction based on deep learning: A survey. *Sensors*, v. 24, n. 11, 2024. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/24/11/3454>.

WU, Y. et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

XENOS, H. G. *Gerenciando a Manutenção Produtiva*. [S.l.]: Editora Dg, 1998.

YANG, X.; LI, J.; JIANG, X. Research on information leakage in time series prediction based on empirical mode decomposition. *Scientific Reports*, v. 14, p. 28362, 2024. Disponível em: <https://doi.org/10.1038/s41598-024-80018-9>.

ZHANG, L. et al. A review on deep learning applications in prognostics and health management. *IEEE Access*, v. 7, p. 162415–162438, 2019.

ZHENG, C. et al. A data-driven approach for remaining useful life prediction of aircraft engines. In: IEEE. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. [S.l.], 2018. p. 184–189.

ZHENG, S. et al. Long short-term memory network for remaining useful life estimation. In: *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/ICPHM.2017.7998311>.

APÊNDICE A – DESCRIÇÃO DOS ATRIBUTOS DO CONJUNTO FD001

Tabela A.1 – Descrição dos settings e sensores disponíveis no conjunto FD001 do C-MAPSS.

Coluna	Descrição aproximada
setting_1	Modificador de velocidade do fan (fan speed setting)
setting_2	Condição ambiental / sangria de ar (bleed)
setting_3	Parâmetro operacional estático (ex: altitude, fixo em FD001)
sensor_1	Pressão do fan inlet
sensor_2	Velocidade do bypass
sensor_3	Temperatura do fan inlet
sensor_4	Pressão da câmara de combustão
sensor_5	Temperatura do bleed de alta pressão
sensor_6	Pressão do bleed de alta pressão
sensor_7	Velocidade do fan (nível do motor)
sensor_8	Temperatura do estágio de alta pressão
sensor_9	Velocidade do compressor
sensor_10	Temperatura do estágio de baixa pressão
sensor_11	Vibração do fan
sensor_12	Vibração do booster
sensor_13	Corrente do atuador
sensor_14	Pressão na saída do motor (exaustão)
sensor_15	Temperatura da exaustão
sensor_16	Velocidade angular do eixo principal
sensor_17	Temperatura do estágio intermediário
sensor_18	Posição do atuador
sensor_19	Torque ou nível de combustível
sensor_20	Temperatura do óleo
sensor_21	Pressão hidráulica ou fluxo de óleo

Fonte: Adaptado de Saxena et al. (2008)