CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
DIRETORIA DE GRADUAÇÃO
CURSO DE ENGENHARIA ELÉTRICA

RITA VITÓRIA BRAGA SILVA

DESENVOLVIMENTO DE SENSOR VIRTUAL EMBARCADO PARA ESTIMAÇÃO DE VAZÃO DE GÁS EM PROCESSO INDUSTRIAL UTILIZANDO MODELOS DE APRENDIZADO DE MÁQUINA

BELO HORIZONTE
2026

RITA VITÓRIA BRAGA SILVA

DESENVOLVIMENTO DE SENSOR VIRTUAL EMBARCADO PARA ESTIMAÇÃO DE VAZÃO DE GÁS EM PROCESSO INDUSTRIAL UTILIZANDO MODELOS DE APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso, apresentado no Curso de Graduação em Engenharia Elétrica do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Dr. Tulio Charles de Oliveira Carvalho

BELO HORIZONTE
2026

MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA - NG

ATA Nº 23 / 2026 - DEE (11.56.08)

Nº do Protocolo: NÃO PROTOCOLADO

Belo Horizonte-MG, 23 de junho de 2026.

ATA DE DEFESA DO PROJETO FINAL DE CURSO

Aos 19 dias do mês de julho do ano de dois mil e vinte e seis, realizou-se a sessão pública de defesa do projeto final de curso intitulado ***"Desenvolvimento de sensor virtual embarcado para estimação de vazão de gás em processo industrial utilizando modelos de aprendizado de máquina"***, apresentado por **Rita Vitória Braga Silva**, matrícula 20193022297 do **Curso de Engenharia Elétrica – CNG.** Os trabalhos foram iniciados às 11h pelo Professor (Orientador) Túlio Charles de Oliveira Carvalho, do Departamento de Engenharia Elétrica, presidente da banca examinadora, constituída pelos seguintes membros: Professor Raphael Paulo Braga Poubel do Departamento de Engenharia Elétrica e Professor Márcio Matias Afonso do Departamento de Engenharia Elétrica. Após a exposição oral, a candidata foi arguida pelos componentes da banca que decidiram por **aprovar** o trabalho. Para constar, redigiu-se a presente ata que, após aprovada por todos os presentes, será assinada pelos membros da banca e pela discente.

Belo Horizonte, 19 de junho de 2026

(Assinado digitalmente em 24/06/2026 09:19)
MARCIO MATIAS AFONSO
PROFESSOR DO MAGISTERIO SUPERIOR
DEE (11.56.08)
Matrícula: 1459994

(Assinado digitalmente em 24/06/2026 12:37)
RAPHAEL PAULO BRAGA POUBEL
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEE (11.56.08)
Matrícula: 2893447

(Assinado digitalmente em 25/06/2026 09:09)
TULIO CHARLES DE OLIVEIRA CARVALHO
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEE (11.56.08)
Matrícula: 2465363

(Assinado digitalmente em 23/06/2026 19:58)
RITA VITÓRIA BRAGA SILVA
DISCENTE
Matrícula: 20193022297

Processo Associado: 23062.027369/2026-88

Visualize o documento original em **https://sig.cefetmg.br/public/documentos/index.jsp** informando seu número: **23**, ano: **2026**, tipo: **ATA**, data de emissão: **23/06/2026** e o código de verificação: **e53a933c38**

AGRADECIMENTOS

Agradeço a Deus por estar ao meu lado durante toda a minha vida, especialmente nesta etapa, concedendo-me saúde, força e discernimento para superar os desafios ao longo desta jornada.

Aos meus pais e amigos, pelo apoio incondicional, incentivo constante e compreensão em todos os momentos da minha trajetória acadêmica.

Aos professores do CEFET-MG, que contribuíram direta ou indiretamente para minha formação, em especial ao meu orientador, Dr. Túlio Charles de Oliveira Carvalho, pela condução, paciência e autonomia concedida ao longo do desenvolvimento deste trabalho.

Por fim, agradeço às empresas LUMAR METALS e HYPELIFT AEROSPACE AND DEFENSE pela oportunidade e por viabilizarem a realização desta pesquisa.

RESUMO

Sensores virtuais constituem uma alternativa de baixo custo para a estimação de variáveis de processo, reduzindo a necessidade de instrumentação física adicional e ampliando as possibilidades de monitoramento em sistemas industriais. Nesse contexto, este trabalho propõe o desenvolvimento e a implementação embarcada de sensores virtuais baseados em aprendizado de máquina para estimação da vazão de gás em uma planta de bancada que emula um sistema de bicos injetores de alto-forno. Inicialmente, foi realizada a aquisição de dados experimentais da planta, seguida do pré-processamento e da construção de modelos preditivos utilizando as técnicas *Perceptron* Multicamadas (MLP) e as Florestas Aleatórias, do inglês *Random Forest* (RF). Posteriormente, os modelos foram convertidos e embarcados em uma plataforma ESP32. Em ambos os ambientes, computacional e embarcado, os modelos foram avaliados por meio das métricas coeficiente de determinação (R^2), erro absoluto médio (MAE) e erro quadrático médio (MSE), além do erro percentual absoluto médio (MAPE) e da análise dos recursos computacionais demandados durante a execução embarcada. Os resultados demonstraram que ambos os modelos foram capazes de estimar adequadamente a variável de interesse, apresentando elevados valores de R^2 e baixos erros de predição. Entretanto, o modelo RF apresentou desempenho superior ao MLP tanto em ambiente computacional quanto embarcado, mantendo maior estabilidade das métricas após a implementação no microcontrolador e demandando menor utilização de memória. Os resultados obtidos demonstram o potencial da aplicação de técnicas de aprendizado de máquina no desenvolvimento de sensores virtuais para sistemas embarcados, contribuindo para a estimação de variáveis de processo em ambientes sujeitos a restrições de processamento e memória.

Palavras-chave: sensor virtual; aprendizado de máquina; *Random Forest*; *Perceptron* Multicamadas; sistemas embarcados; ESP32.

SUMÁRIO

1 INTRODUÇÃO

Os medidores de vazão ou fluxo são instrumentos projetados para fornecerem dados relacionados à taxa de fluxo ou à quantidade de fluido que flui em determinado ponto de análise, sendo amplamente utilizados no setor industrial, incluindo aplicações na gestão de água, petróleo, gás natural, produtos químicos e outros fluidos industriais (BEZERRA, 2024). Sua importância está associada à capacidade de monitorar e controlar o fluxo desses materiais, garantindo eficiência operacional, segurança e maior confiabilidade do processo.

Atualmente, no setor siderúrgico, altos-fornos, reatores metalúrgicos responsáveis pela produção do ferro-gusa a partir da combinação de minério de ferro, coque e fundentes, utilizam injeção de combustíveis auxiliares gasosos, como carvão pulverizado e gás natural, através de bicos injetores localizados na região inferior do forno (BELO, 2019; MAJESKI et al., 2015). Esta injeção auxiliar permite uma redução do uso do coque e contribui para a economia de energia, redução de emissões de gases do efeito estufa e, em alguns casos, redução de custos (MAJESKI et al., 2015; WASEM; SATHLER; GENEROSO, 2023). Dessa forma, a medição e o controle adequados da vazão de gás tornam-se fundamentais para assegurar estabilidade operacional e eficiência térmica, sendo essa medição realizada, geralmente, por sensores físicos (neste caso, sensores de vazão).

Entretanto, sensores físicos estão sujeitos a erros de medição, atrasos na resposta, limitações relacionadas à disponibilidade e confiabilidade do fabricante, além de apresentarem custo elevado e potencial interferência no processo em que são inseridos (SILVA, 2022). No caso específico dos sensores de vazão empregados em altos-fornos, estes são posicionados próximos às regiões de injeção, onde o gás é introduzido em alta velocidade, sob temperaturas na ordem de 1100 a 1250 °C, caracterizando um ambiente adverso para observação e medição (BELO, 2019; MAJESKI et al., 2015). Nessas condições, a precisão dos medidores pode ser significativamente afetada por variações de temperatura, pressão e demais fatores ambientais, como umidade e vibração, exigindo cuidados adicionais na conversão e interpretação dos dados obtidos (BEZERRA, 2024).

Nesse contexto, a medição indireta de uma determinada grandeza (denominada variável de interesse) a partir de outras já conhecidas, obtidas por meio de modelos matemáticos ou dados reais provenientes de sensores físicos, e utilizando técnicas de aprendizado de máquina, configura-se como uma alternativa complementar aos sensores físicos (LIMA, 2022b). Desse modo, este trabalho propõe o desenvolvimento de um sensor virtual, do inglês *soft sensor*, para estimar a vazão de gás em uma planta de bancada que emula um sistema de bicos injetores de alto-forno. Para tal, são investigados e comparados dois modelos de aprendizado de máquina: o Perceptron de Múltiplas Camadas (ou Multicamadas) e as Florestas Aleatórias, do inglês *Multilayer Perceptron* (MLP) e *Random Forest* (RF) respectivamente, com o objetivo de avaliar o desempenho preditivo de cada abordagem frente a um mesmo conjunto de dados reais.

Além disso, os modelos são implementados em um sistema embarcado, por meio do

microcontrolador ESP32, do tipo *System on Chip* (SoC)[1], visando analisar seu desempenho e avaliar sua viabilidade de inserção na planta experimental como complemento ou redundância ao sensor físico existente.

2 DESENVOLVIMENTO

2.1 Fundamentação Teórica

2.1.1 Sensor Virtual na Indústria

As plantas industriais são dotadas de diversos instrumentos de medição, os sensores, responsáveis pela aquisição de dados em tempo real, possibilitando automação, monitoramento e controle dos processos (SILVA, 2022; JUVÊNCIO, 2023). Nos últimos anos, com o aumento significativo da quantidade de dados gerados e armazenados na indústria, essas informações passaram a ser exploradas no desenvolvimento de modelos preditivos orientados a dados, conhecidos como sensores virtuais (JUVÊNCIO, 2023).

Sensores virtuais são modelos computacionais capazes de estimar variáveis de difícil ou elevado custo de medição direta, a partir de dados acessíveis provenientes de outros sensores, sem necessidade de conexão física adicional ao processo (JUVÊNCIO, 2023; DRUMOND, 2025). No contexto industrial, esses modelos têm sido empregados no monitoramento e controle de processos por meio da estimativa de variáveis em tempo real.

Além disso, sensores virtuais podem atuar como sistemas de reserva, conhecido como *backup*, ou como fontes redundantes de informação, operando em paralelo com dispositivos de medição física e sendo acionados quando o *hardware* apresenta falhas ou necessita de intervenções de manutenção (JUVÊNCIO, 2023; MATHEUS, 2024). Seu uso pode contribuir para a prevenção de falhas, melhoria da eficiência dos processos, redução de custos e menor dependência de instrumentação física (DRUMOND, 2025; MATHEUS, 2024). Contudo, a eficácia desses modelos depende de adequada calibração, disponibilidade de dados confiáveis e recursos computacionais compatíveis com a aplicação, especialmente em cenários embarcados (DRUMOND, 2025).

O desenvolvimento de sensores virtuais pode ser realizado por diferentes abordagens, destacando-se a aplicação de técnicas de aprendizado de máquina, que conferem maior robustez e adaptabilidade aos modelos (SILVA, 2022).

2.1.2 Aprendizado de Máquina aplicado a Sensores Virtuais

Essas abordagens são comumente classificadas em modelos orientados por conhecimento físico do processo (caixa branca) ou modelos orientados a dados (caixa preta). Os modelos de caixa branca são fundamentados em equações matemáticas que descrevem os princípios físicos

[1] Circuito integrado que reúne em um único chip os principais componentes de um sistema computacional, como processador, memória e periféricos de comunicação.

e químicos do sistema, enquanto os modelos de caixa preta estabelecem relações entre variáveis a partir da análise de dados históricos do processo (LIMA, 2022a).

Com o aumento da disponibilidade de dados industriais e da capacidade computacional, abordagens orientadas a dados têm ganhado destaque na construção de sensores virtuais. Nesse contexto, insere-se o Aprendizado de Máquina (ou *Machine Learning* – ML), definida como um conjunto de algoritmos capazes de ajustar automaticamente seus parâmetros por meio de um processo de treinamento baseado em dados (SILVA, 2022). Esse aprendizado pode ocorrer de forma supervisionada, quando são fornecidos dados de entrada e suas respectivas saídas desejadas, ou não supervisionada, quando apenas os dados de entrada estão disponíveis e o algoritmo deve identificar padrões intrínsecos no conjunto de dados (SILVA, 2022).

Diversas técnicas de aprendizado de máquina têm sido empregadas na construção de sensores virtuais, incluindo Redes Neurais Artificiais (RNAs), *Random Forest*, Máquinas de Vetores de Suporte (SVM), métodos baseados em aprendizado extremo entre outras (SILVA, 2022). A escolha da técnica depende das características do processo, da disponibilidade e qualidade dos dados e dos requisitos da aplicação

2.2 Estado da Arte: Sensores Virtuais e Aprendizado de Máquina

Na literatura, a aplicação de técnicas de aprendizado de máquina no desenvolvimento de sensores virtuais tem sido amplamente investigada, especialmente em processos industriais que apresentam relações não lineares e dinâmica complexa (DRUMOND, 2025).

A partir do levantamento bibliográfico realizado, observa-se que as Redes Neurais Artificiais (RNAs) são amplamente empregadas na estimação de variáveis em diferentes contextos industriais, como para estimar a consistência de celulose, o teor de umidade em processos de granulação ou a concentração de oxigênio em sistemas de combustão; além disso suas distintas arquiteturas vêm sendo estudadas e comparadas entre si (MATHEUS et al., 2023; ZÁHONYI et al., 2025). No estudo apresentado em MATHEUS et al. (2023), por exemplo, as arquiteturas MLP, Rede de Função de Base Radial (RBF) e Máquina de Aprendizado Extremo (ELM) foram avaliadas na estimação da consistência de pasta termomecânica de celulose, utilizando dados reais de planta industrial. Os resultados indicaram que a MLP apresentou o menor erro percentual absoluto médio (EPAM $\approx$ 3,5%), enquanto RBF e ELM obtiveram desempenhos próximos (3,9% e 3,6%, respectivamente).

De forma complementar, em MATHEUS (2024), sensores virtuais baseados em RNAs foram aplicados à estimação de consistência e *freeness* de celulose. O autor destaca que arquiteturas como a ELM podem apresentar vantagens associadas ao tempo de treinamento e à simplicidade computacional, mantendo desempenho competitivo em relação a outras estruturas. Esses resultados evidenciam que múltiplas arquiteturas podem ser empregadas com desempenho satisfatório, sendo que a escolha da mais adequada depende do problema abordado, das características do conjunto de dados e dos critérios de avaliação adotados.

Apesar de a maioria dos trabalhos identificados empregar o MLP, devido à sua simplicidade estrutural e ampla consolidação na literatura, também foram encontradas abordagens baseadas em Redes Neurais Recorrentes (RNNs), modelos NARX (*Nonlinear Autoregressive with eXogenous inputs*) e LSTM (*Long Short-Term Memory*), exploradas como alternativas às arquiteturas tradicionais. Essas variações demonstram a diversidade de estratégias adotadas para a modelagem de sensores virtuais em aplicações industriais (DRUMOND, 2025; ZÁHONYI et al., 2025; THURUTHEL; GARDNER; ILDA, 2022).

Parte da literatura também apresenta análises comparativas entre RNAs e outras técnicas de aprendizado de máquina, como SVM e RF. Conforme observado em estudos como os de SILVA (2022), JUVÊNCIO (2023), modelos baseados em RNAs tendem a apresentar elevada acurácia em tarefas de detecção. Por outro lado, as outras técnicas demonstraram desempenho superior ou mais adequado em etapas de diagnóstico ou classificação mais detalhada de falhas. Esses resultados reforçam que a escolha da técnica está diretamente relacionada ao objetivo da modelagem e às características do conjunto de dados disponível.

Por fim, embora o uso de RNAs na construção de sensores virtuais esteja consolidado, a implementação embarcada desses modelos ainda aparece de forma mais recente na literatura. Trabalhos como os de DRUMOND (2025), ATANANE et al. (2023), CHAORAINGERN, TIPSUWANPORN e NUMSOMRAN (2023), EFFENDY et al. (2022), SILVA (2023) investigam a aplicação de conceitos associados ao TinyML [2] e a execução de modelos em microcontroladores de baixo custo. Entretanto, parte dessas pesquisas utiliza dados sintéticos provenientes de equações matemáticas ou não realiza efetivamente o embarque do modelo no hardware final, o que evidencia a necessidade de estudos que integrem modelagem, validação experimental com dados reais e implementação embarcada em microcontroladores.

2.3 Metodologia

A metodologia adotada foi desenvolvida a partir de uma abordagem experimental e computacional, abrangendo desde a obtenção dos dados por meio de ensaios em bancada até a implementação embarcada dos modelos preditivos selecionados.

2.3.1 Bancada de Teste de Bicos Injetores

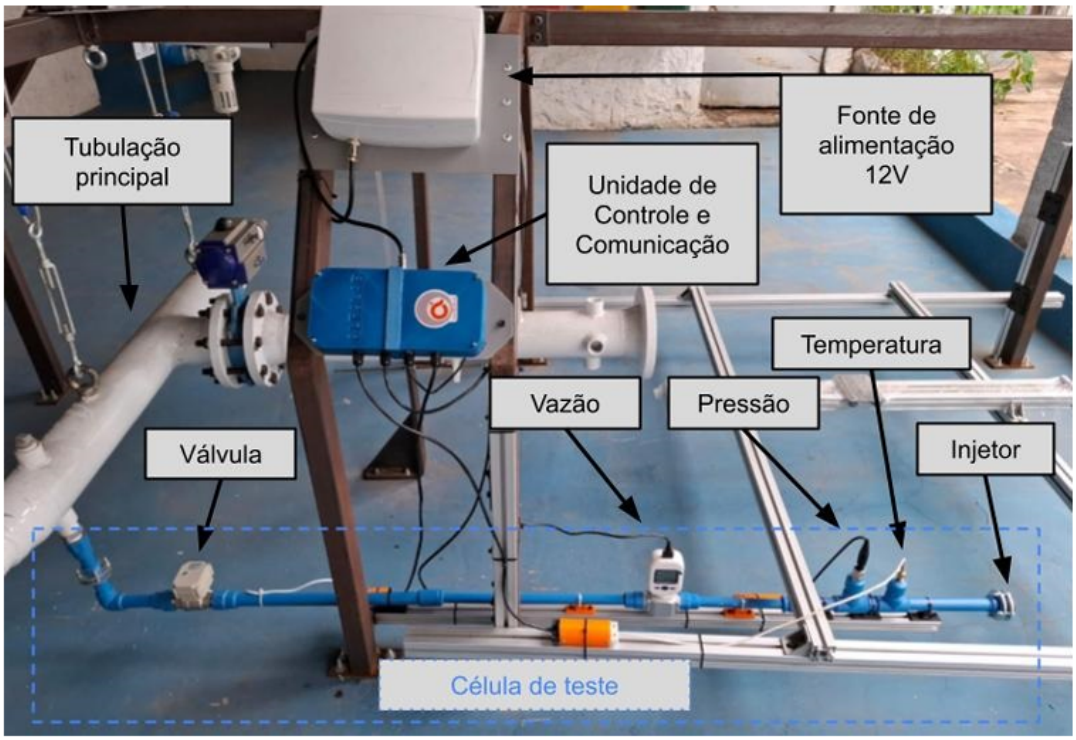
O conjunto de dados utilizado neste trabalho foi obtido por meio de ensaios em uma bancada de teste de bicos injetores de gás, projetada para emular, de forma controlada, condições operacionais associadas a fornos conversores de ferro-gusa em aço. Essa infraestrutura experimental foi desenvolvida pelo Prof. Dr. José Eduardo Mautone Barros, da Universidade Federal de Minas Gerais, e contou com a colaboração do Dr. Túlio Charles de Oliveira Carvalho na etapa de instrumentação física da planta. Dessa parceria resultou o trabalho apresentado em

[2] TinyML é o termo utilizado para designar o campo de pesquisa dedicado à execução de modelos de aprendizado de máquina em dispositivos com severas restrições de energia, memória e capacidade de processamento, como microcontroladores.

CARVALHO et al. (2025), que descreve a concepção da bancada e do sistema de aquisição de dados, originando a base experimental empregada neste estudo.

A bancada é composta por um sistema de suprimento de gás pressurizado, formado por cilindros conectados a reguladores responsáveis por manter a pressão constante na tubulação principal, e por uma célula de teste destinada à instalação do bico injetor e de bocais distintos, possibilitando variações nas condições de escoamento e, consequentemente, nos parâmetros fluidodinâmicos correspondentes, conforme ilustrado na Figura 1.

Figura 1 – Visão geral da bancada de teste desenvolvida para testes de controle de processos industriais



Fonte: CARVALHO et al. (2025).

O sistema de medição é composto pelos sensores PS-10B, LM35 e MF5700, responsáveis pelo monitoramento da pressão de entrada, temperatura e vazão mássica de gás, respectivamente. O sensor MF5700 comunica-se com o sistema embarcado por meio do padrão RS-485 e suas leituras são utilizadas como referência da vazão mássica na construção dos modelos preditivos desenvolvidos neste trabalho. A pressão de saída corresponde à pressão atmosférica do ambiente, não sendo medida por um sensor dedicado.

A coordenação da bancada é realizada por uma unidade embarcada baseada em um SoC da classe ESP32, desenvolvido pela *Espressif Systems*. Essa unidade opera sob o sistema operacional *FreeRTOS*[3], sendo responsável pelas tarefas de aquisição, processamento e comunicação

[3] Sistema operacional de tempo real, do inglês *Real-Time Operating System* (RTOS), com código livre e gratuito para microcontroladores e pequenos microprocessadores, projetado para lidar com tarefas embarcadas complexas.

dos dados experimentais.

Os dados utilizados neste estudo foram obtidos por meio de ensaios em degrau na abertura da válvula do bico injetor. Em cada patamar de abertura, o sistema era mantido até atingir regime permanente e, em seguida, ajustado para uma nova condição operacional. A repetição desse procedimento permitiu a obtenção de registros em diferentes pontos de operação, contemplando tanto regimes estacionários quanto transitórios.

Dessa forma, a bancada forneceu os dados experimentais utilizados na construção, avaliação e futura aplicação do sensor virtual proposto.

2.3.2 Estruturação e Tratamento dos Dados

2.3.2.1 Aquisição

Os dados provenientes dos sensores são estruturados em formato JSON [4], garantindo padronização e facilitando a integração entre os módulos do sistema. Para o gerenciamento e distribuição das mensagens, foi implementado um *broker* MQTT [5]em rede local, permitindo o encaminhamento das informações à interface *front-end* [6], onde são exibidas em tempo real.

De forma recorrente, quando requerido para fins de análise, os dados são armazenados localmente em arquivos no formato CSV, como ocorreu durante a geração do conjunto de dados utilizado neste estudo, possibilitando análises *offline*. Cada registro é associado a uma marca temporal (*timestamp*) gerado pela unidade de controle e comunicação, assegurando rastreabilidade temporal das medições.

2.3.2.2 Processamento

Para manipulação, leitura e organização dos dados experimentais, empregou-se a linguagem `Python` com a biblioteca `Pandas`, responsável pelo carregamento do arquivo e tratamento das informações utilizadas durante o desenvolvimento computacional deste trabalho (ver Apêndice B).

Pode-se afirmar que o objeto de estudo consiste em um problema de aprendizado supervisionado do tipo regressão, uma vez que o objetivo dos modelos é estimar valores contínuos de vazão mássica a partir de variáveis medidas no sistema físico.

---

[4] Sigla para *JavaScript Object Notation* sendo um formato leve de texto usado para armazenar e transportar dados entre sistemas.

[5] MQTT é sigla do protocolo *Message Queuing Telemetry Transport* que gerencia e roteia mensagens entre dispositivos em uma rede (Internet das Coisas - IoT) utilizando um modelo de publicação/assinatura.

[6] Parte visual de um site ou aplicativo com a qual o usuário interage diretamente.

2.3.2.3 Análise de Correlação e Seleção de Variáveis

Com o objetivo de compreender o comportamento das variáveis do processo e auxiliar na seleção das entradas dos modelos, realizou-se previamente uma análise de correlação entre os parâmetros disponíveis no conjunto de dados conforme Tabela 6 no Apêndice A.

A análise permitiu identificar o grau de associação linear entre cada parâmetro e a vazão mássica mostrando elevada correlação entre variáveis geométricas e fluidodinâmicas relacionada à válvula e ao escoamento, como abertura da válvula (*ValveApt* e *ValveAng*), área efetiva (*ValveArea*), coeficiente de descarga (CD) e número de *Reynolds*.

Entretanto, foram selecionadas como variáveis de entrada apenas a abertura da válvula *ValveAPt(%)*, temperatura *Temperature(K)*, pressão de entrada *IntakeP(Pa)* e pressão de saída *OutputP(Pa)* com a variável de interesse sendo a vazão mássica *Massflow (kg/s)*.

Essa decisão fundamenta-se nas condições específicas dos ensaios realizados neste estudo, conduzidos sem a utilização de bocais adicionais nos bicos injetores, o que reduz interferências associadas a variações geométricas no escoamento e permite tratar parâmetros como o CD e o número de *Reynolds* como aproximadamente constantes nas condições analisadas. Além disso, embora a variável temporal apresente correlação elevada com a vazão mássica, ela não foi utilizada como variável de entrada dos modelos por não representar uma grandeza física diretamente relacionada ao mecanismo de escoamento. Sua correlação decorre principalmente da sequência operacional adotada durante os ensaios, não constituindo uma variável causal do processo.

Por fim, optou-se por priorizar variáveis diretamente mensuráveis pelo sistema, evitando a inclusão de parâmetros calculados ou dependentes de relações teóricas específicas. Logo a análise de correlação serviu como instrumento complementar, sendo a definição das variáveis de entrada baseada também em critérios físicos e operacionais relacionados ao contexto experimental.

2.3.2.4 Análise Temporal

A análise temporal das variáveis de processo foi realizada por meio dos perfis temporais e de suas derivadas discretas (Figura 10, Apêndice A). Observa-se que a abertura da válvula foi aplicada em degraus sucessivos, resultando em diferentes pontos operacionais e regiões transitórias, com resposta direta na vazão mássica.

A temperatura apresentou oscilações de baixa amplitude em torno de um valor médio, enquanto a pressão de saída permaneceu aproximadamente constante e a pressão de entrada apresentou tendência decrescente ao longo do ensaio.

A análise das derivadas confirmou o comportamento observado, indicando regiões de baixa variação temporal intercaladas com picos associados às mudanças operacionais. De forma geral, os dados contemplam tanto regimes estacionários quanto transitórios, caracterizando adequadamente diferentes condições de operação do sistema.

2.3.2.5 Normalização

Separando-se as variáveis de entrada em X da variável de saída em Y, foi aplicada a normalização dos dados de entrada por meio da classe `MinMaxScaler` da biblioteca `Scikit-learn` (ver Apêndice B). Esse procedimento transforma os valores das variáveis para um intervalo entre 0 e 1, reduzindo discrepâncias de escala entre os parâmetros do processo e favorecendo o desempenho dos algoritmos de aprendizado de máquina.

A normalização foi ajustada apenas com os dados de treinamento e posteriormente aplicada aos dados de teste, evitando vazamento de informações entre as etapas do processo. Ressalta-se que os dados de saída não foram normalizados, pois já apresentavam valores no intervalo entre 0 e 1, além de possibilitarem melhor interpretabilidade dos resultados.

2.3.3 Modelos de Aprendizado de Máquina

Na etapa de desenvolvimento dos modelos de aprendizado de máquina empregados para inferência da vazão mássica do sistema selecionou-se modelos supervisionados capazes de representar relações não lineares entre as variáveis de entrada e a variável de interesse, compatíveis com a implementação no microcontrolador ESP32. Dessa forma, foram avaliadas duas abordagens distintas: o Perceptron Multicamadas e as Florestas Aleatórias.

2.3.3.1 Premissas

O conjunto de dados utilizado neste trabalho é composto por 482 amostras para cada variável monitorada. Em função desse tamanho reduzido, optou-se por limitar a complexidade dos modelos supervisionados, priorizando arquiteturas compactas compatíveis com a aplicação em sistema embarcado, com o objetivo de reduzir o risco de sobreajuste, do inglês *overfitting*, e evitar o uso de modelos superdimensionados.

A escolha dos modelos visa não apenas a comparação de desempenho preditivo nas etapas de treinamento e validação, utilizando o conjunto de dados obtido experimentalmente, mas também a análise de aspectos como custo computacional, estabilidade e viabilidade de execução no SoC ESP32. Dessa forma, busca-se avaliar a possibilidade de substituição do sensor físico de vazão por um sensor virtual baseado em aprendizado de máquina.

Os dados foram divididos em conjuntos de treinamento e teste na proporção de 80% e 20%, respectivamente, por meio de amostragem aleatória, garantindo a representatividade das diferentes condições operacionais observadas nos ensaios experimentais.

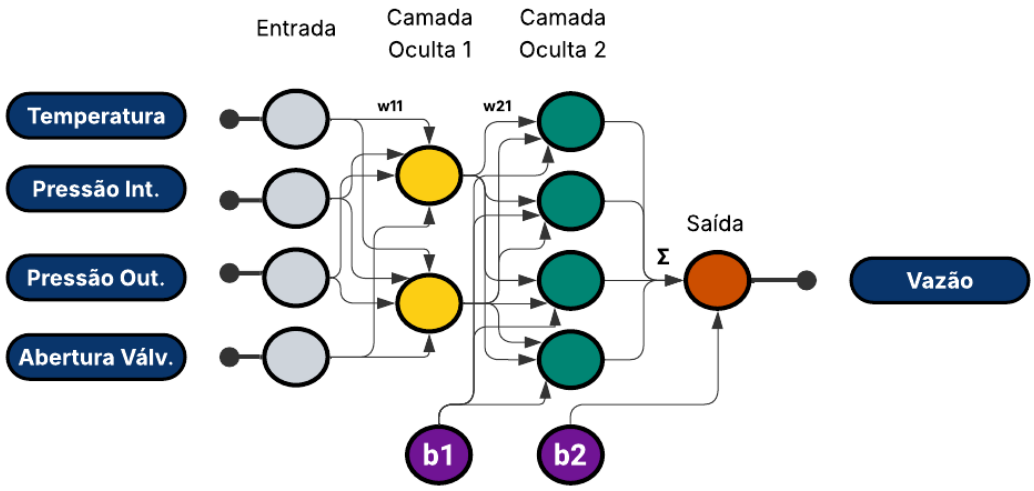
Embora o sistema apresente comportamento temporal associado às variações operacionais, o objetivo deste trabalho não é a previsão de séries temporais, mas a estimação instantânea da vazão mássica a partir de variáveis medidas no mesmo instante. Assim, a divisão aleatória dos dados é adequada para avaliar a capacidade de generalização dos modelos.

Durante a etapa de treino da rede neural, o conjunto de treinamento foi novamente subdividido em 80% para ajuste dos parâmetros e 20% para validação, permitindo o monitoramento do processo de aprendizagem e a detecção de possíveis sinais de sobreajuste.

2.3.3.2 Perceptron de Multicamadas

O MLP é uma das arquiteturas mais utilizadas em RNAs, sendo amplamente aplicado em problemas de regressão e em sistemas embarcados devido à sua simplicidade estrutural, baixo custo computacional e capacidade de modelar relações não lineares (DRUMOND, 2025). Sua estrutura é organizada em camadas, sendo composta por uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, onde os neurônios de cada camada são conectados aos neurônios da camada subsequente (Figura 2).

Figura 2 – Rede Neural Artificial *Perceptron* Multicamadas.

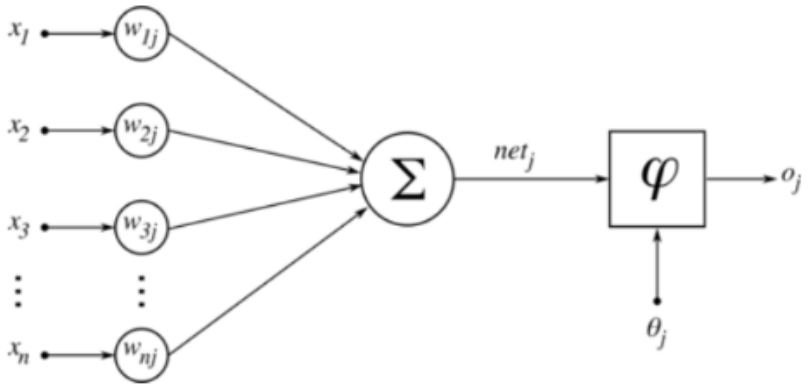


Fonte: Elaborado pelo autor (2026).

Cada neurônio realiza uma operação matemática baseada na soma ponderada das entradas, conforme ilustrado na Figura 3, em que cada variável é multiplicada por um peso sináptico associado $\omega_{11}, \omega_{12}, \omega_{31}, \ldots, \omega_{nj}$, onde j se refere ao neurônio em análise e n ao terminal de entrada correspondente. Esses pesos podem assumir diferentes intensidades e características excitatórias ou inibitórias, de acordo com seu valor e sinal. O resultado é submetido a uma função de ativação φ, responsável por introduzir não linearidades ao modelo, sendo posteriormente adicionado a um limiar de ativação θ_j [7] para produzir a saída do neurônio o_j, correspondente à resposta do neurônio às entradas $x_1, x_2, x_3, \ldots, x_n$ (JUVÊNCIO, 2023; DRUMOND, 2025; SILVA, 2023).

[7] O limiar de ativação é atualmente incorporado ao termo *bias* (viés), dado por $b_1, b_2, \ldots, b_j$, em modelos modernos de redes neurais artificiais. Portanto b_j = - θ_j.

Figura 3 – Estrutura do neurônio artificial.



Fonte: SILVA (2023).

A definição da arquitetura adotada neste trabalho foi baseada em DRUMOND (2025), que utilizou a mesma planta experimental empregada neste estudo como objeto de análise. Entretanto, no estudo de referência, os dados utilizados eram sintéticos, gerados a partir da modelagem matemática do escoamento de gás na válvula e implementados por meio de *scripts* em `Python` utilizando as bibliotecas `NumPy` e `Pandas`, devido à indisponibilidade de dados reais provenientes da planta durante o período de desenvolvimento.

Com o mesmo objetivo de substituição do sensor físico de vazão por um sensor virtual baseado em modelos de aprendizado de máquina implementados em um microcontrolador ESP32, diferentes arquiteturas de MLP foram desenvolvidas, treinadas e avaliadas tanto em ambiente computacional quanto em sistema embarcado.

As arquiteturas avaliadas foram implementadas por meio da biblioteca `TensorFlow`, responsável por criar as camadas, treinar e validar o modelo, e definidas a partir da variação dos hiperparâmetros, conforme apresentado na Tabela 1.

Tabela 1 – Variações dos hiperparâmetros avaliados em DRUMOND (2025)

Hiperparâmetros	**Valores utilizados**
Número de neurônios	2, 4, 8 e 16
Número de camadas ocultas	1 e 2
Volume de dados	10 000 e 40 000
Épocas	100, 200 e 400
Batch size [8]	4, 8, 16 e 32
Funções de ativação	*ReLU*, *ELU*, *SELU* e *Sigmoid*

Fonte: Elaborado pelo autor (2026).

Na pesquisa utilizada como base, o número de neurônios e de camadas ocultas constituiu o critério inicial para definição das arquiteturas testadas e analisadas nas próximas fases do

[8] Número de amostras processadas antes de cada atualização dos pesos da rede.

estudo, sendo selecionadas configurações com diferentes níveis de complexidade, abrangendo desde redes mais simples até modelos mais complexos.

Entretanto, embora o volume de dados tenha sido considerado no estudo anterior, esse parâmetro não faz parte do MLP propriamente dito, sendo tratado neste trabalho como uma característica fixa do conjunto de dados experimental obtido diretamente na planta.

Dessa forma, com base nas análises e resultados apresentados no estudo de referência, foram adotados a seguinte estrutura para o desenvolvimento do modelo neste trabalho:

1. Volume de dados: 482 amostras obtidas diretamente da bancada experimental;
2. Camadas e neurônios: duas camadas ocultas contendo 8 e 16 neurônios, respectivamente;
3. Épocas: 400;
4. *Batch size*: 16;
5. Função de ativação: *ReLU* (*Rectified Linear Unit*);
6. Otimizador: *Adam* (*Adaptive Moment Estimation*).

A escolha desses parâmetros foi baseada nos resultados obtidos no trabalho utilizado como base, no qual arquiteturas mais complexas apresentaram desempenho mais satisfatório. As configurações de número de camadas, número de neurônios, épocas e *batch size* foram selecionadas por apresentarem resultados consistentes durante as etapas de treinamento, validação e execução embarcada. A função de ativação (*ReLU*) foi adotada por apresentar desempenho equilibrado e baixo custo computacional em relação às demais funções avaliadas. Por fim, o otimizador *Adam* foi mantido padrão conforme utilizado no trabalho analisado.

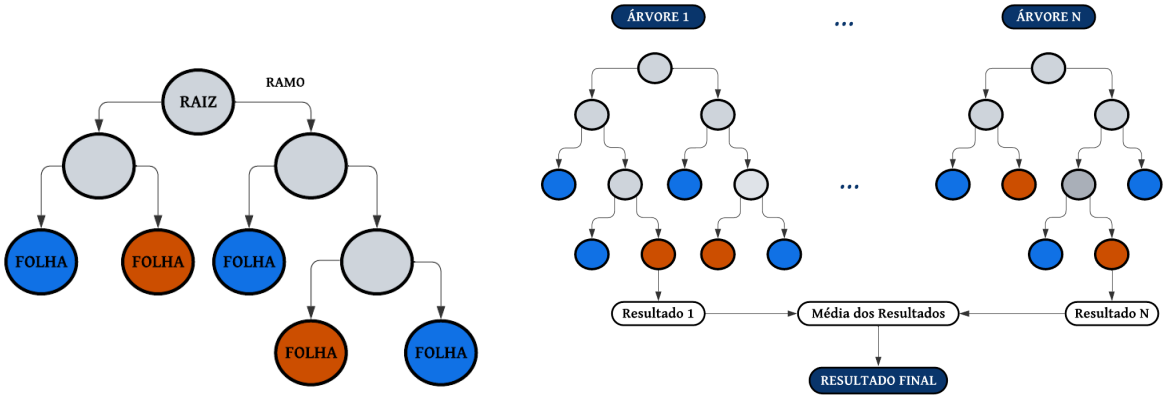
2.3.3.3 Florestas Aleatórias

Para a compreensão desse modelo de aprendizado de máquina é imprescindível entender o funcionamento da Árvore de Decisão, do inglês *Decision Tree* (DT). Esse modelo pode ser representado por uma estrutura hierárquica semelhante a um fluxograma, na qual os dados de entrada são sucessivamente subdivididos até que não seja mais possível realizar novas divisões, resultando na saída preditiva do modelo (SILVA, 2022; QUEIROZ, 2023).

A estrutura de uma DT é composta por diferentes tipos de nós, conforme ilustrado na Figura 4a. O primeiro nó, responsável por receber todo o conjunto inicial de dados, é denominado nó raiz (SILVA, 2022). A partir dele, são gerados novos nós e ramos de acordo com as divisões realizadas durante a etapa de treinamento. Nesse processo, a árvore aprende a organizar os dados por meio da definição das melhores divisões possíveis para o conjunto analisado, utilizando critérios baseados em medidas de impureza, como o índice de Gini, ou entropia e ganho de informação. (SILVA, 2022; QUEIROZ, 2023). Quando não é mais possível subdividir os dados, obtém-se o nó folha, responsável pela saída preditiva do modelo.

Com base nesse conceito, o RF consiste em um algoritmo fundamentado em métodos do tipo *Ensemble*, nos quais múltiplos modelos mais simples são combinados para obtenção de um modelo final mais robusto (SILVA, 2022). Nesse caso, o RF é formado pela combinação de diversas Árvores de Decisão independentes (CHENG; CHUNHONG; QIANGLIN, 2023), conforme Figura 4b.

Figura 4 – Modelos baseados em árvores utilizados neste estudo.



(a) Árvore de Decisão

(b) Random Forest

Em problemas de classificação, a saída final do modelo é obtida por meio do critério de votação entre as árvores. Já em problemas de regressão, como o desenvolvido neste trabalho, o resultado final corresponde à média das saídas preditivas geradas por cada árvore da floresta (SILVA, 2022).

O RF apresenta maior robustez e menor tendência ao sobreajuste quando comparado a uma única DT (SILVA, 2022). Isso ocorre devido à utilização da técnica de *bagging*, dado pelo *bootstrap aggregating*, na qual cada árvore é treinada a partir de subconjuntos aleatórios do conjunto de dados selecionados com reposição, permitindo que uma mesma amostra possa ser escolhida mais de uma vez (CHENG; CHUNHONG; QIANGLIN, 2023).

Além disso, durante o treinamento de cada árvore, apenas uma parcela aleatória das variáveis de entrada é utilizada na definição das divisões realizadas em cada nó (CHENG; CHUNHONG; QIANGLIN, 2023). Em muitos casos, a quantidade de variáveis selecionadas corresponde à raiz quadrada do número total de atributos disponíveis. Essa aleatoriedade torna as árvores menos correlacionadas entre si, contribuindo para a redução da variância do modelo e mitigação do sobreajuste (SILVA, 2022).

Para a implementação do *Random Forest*, utilizou-se a classe `RandomForestRegressor`, pertencente ao módulo *Ensemble* da biblioteca `Scikit-learn`. Essa implementação permite o ajuste de diferentes hiperparâmetros, realizado por meio da técnica de validação cruzada associada ao método `GridSearchCV`, explorando combinações predefinidas de parâmetros, conforme apresentado na Tabela 2 (SILVA, 2022; MARTINS; REIS; PÁBON, 2025).

A arquitetura resultante da validação cruzada, denominada o modelo ótimo (*best model* - bm), encontra-se sublinhada na Tabela 2. Entretanto, visando também analisar configurações de menor complexidade computacional para o problema abordado, definiu-se uma segunda arquitetura comparativa, destacada em caixa na mesma tabela. Os hiperparâmetros não especificados foram mantidos com os valores padrão da biblioteca utilizada (ver Apêndices B e C).

Tabela 2 – Hiperparâmetros do modelo *Random Forest*.

Parâmetro	**Descrição**	**Faixa de valores**
n_estimators	Número de árvores no conjunto. Aumenta a capacidade preditiva do modelo.	10, 100, 200 e 300
max_depth	Profundidade máxima de cada árvore. Controla a complexidade e evita o sobreajuste.	*None*, 5, 10, 20, 30 e 40
min_samples_split	Número mínimo de amostras para dividir um nó interno. Reduz divisões irrelevantes.	2, 5, 10 e 20
min_samples_leaf	Número mínimo de amostras em um nó folha. Suaviza o modelo e reduz variância.	1,2 e 4
max_features	Número de atributos considerados na melhor divisão. Introduz diversidade entre as árvores.	*sqrt*, *log2*, 0.5, 0.7 e 1.0

Fonte: Adaptado de MARTINS, REIS e PÁBON (2025).

A fim de evitar confusão, adota-se, a partir deste ponto, a notação RF_{comp} para o modelo compacto e RF_{bm} para o modelo ótimo.

2.3.4 Conversão de modelos

Após o treinamento dos modelos, torna-se necessário convertê-los para formatos compatíveis com dispositivos embarcados de recursos limitados. No caso do MLP, utilizou-se o conversor *TensorFlow Lite*[9], por meio da API `TFLiteConverter`, responsável por transformar o modelo treinado em um arquivo com extensão `.tflite`. Esse processo requer a utilização de arquiteturas compatíveis com o interpretador *TensorFlow Lite Micro*[10], restringindo o modelo a operações e camadas suportadas pelo ambiente embarcado.

Posteriormente, o arquivo `.tflite` foi convertido para um arquivo de cabeçalho `.h`, permitindo sua incorporação direta ao *firmware* do ESP32 e viabilizando a execução embarcada da inferência em tempo real.

Para o modelo *Random Forest*, a conversão foi realizada por meio da extração das regras de decisão de cada árvore da floresta. Os nós internos foram representados por estruturas condicionais do tipo `if/else`, enquanto os nós terminais foram convertidos em valores de retorno correspondentes às predições da árvore. As árvores geradas foram armazenadas como funções independentes em um arquivo `.h`, sendo posteriormente agrupadas em uma função de predição responsável por calcular a média de suas saídas, reproduzindo o comportamento original do

[9] Ferramenta desenvolvida no *TensorFlow* para suporte a dispositivos embarcados.
[10] Versão do *TensorFlow Lite* destinada à execução em microcontroladores.

algoritmo durante a inferência. Dessa forma, o modelo pode ser executado diretamente no microcontrolador sem a necessidade de bibliotecas específicas de aprendizado de máquina em tempo de execução.

2.3.5 Implementação Embarcada e Protocolo MQTT

O *firmware* foi desenvolvido na plataforma `Arduino IDE`, integrando bibliotecas para conectividade *Wi-Fi*, comunicação MQTT e execução dos modelos embarcados. Para o modelo MLP, utilizou-se a biblioteca `ArduTFLite`, responsável pelo carregamento do modelo convertido para o formato `.tflite` e pela execução da inferência no ESP32. Já o modelo RF foi incorporado diretamente ao *firmware* por meio do arquivo `.h` gerado durante a etapa de conversão.

O ESP32 foi configurado para se conectar a uma rede *Wi-Fi* local e a um *broker* MQTT, por meio do qual recebe os dados de entrada e publica os resultados da inferência. Para a realização dos testes, foi desenvolvido um *script* em `Python` responsável por enviar sequencialmente as 482 amostras do conjunto de dados experimental ao microcontrolador, em formato JSON, com intervalo de 500 ms entre mensagens (ver Apêndice D). Esse intervalo foi adotado de forma a garantir a estabilidade da comunicação e permitir a execução completa da inferência e transmissão dos resultados antes do envio da amostra seguinte. Além disso, o período de 500 ms é compatível com a dinâmica do sistema e das variáveis físicas monitoradas, uma vez que corresponde ao intervalo de aquisição de dados da planta didática utilizada nos experimentos.

A cada mensagem recebida, os valores das variáveis de entrada são processados e fornecidos ao modelo embarcado para execução da inferência. Em seguida, a vazão mássica estimada é publicada em um segundo tópico MQTT juntamente com informações de desempenho obtidas durante a execução do modelo.

Os dados publicados pelo ESP32 são coletados pelo mesmo *script*, que armazena os resultados, calcula as métricas de desempenho e gera os gráficos utilizados na análise experimental. Dessa forma, o protocolo MQTT foi empregado como mecanismo de comunicação bidirecional entre o ambiente computacional e o sistema embarcado, permitindo a avaliação automatizada do desempenho embarcado dos modelos.

2.3.6 Métricas de Desempenho

A avaliação dos modelos foi realizada por meio de métricas associadas à precisão preditiva, custo computacional e desempenho embarcado. As métricas utilizadas em cada etapa do estudo são apresentadas na Tabela 3.

Tabela 3 – Métricas utilizadas na avaliação dos modelos.

Métrica	Etapa	Descrição
MSE	Computacional	Média dos quadrados dos erros entre os valores reais e preditos. Penaliza erros extremos.
MAE	Computacional e Embarcada	Média das diferenças absolutas entre os valores reais e estimados pelo modelo.
R^2	Computacional e Embarcada	Indica a capacidade do modelo em explicar a variabilidade dos dados observados.
Tempo de Treinamento	Computacional	Tempo necessário para o ajuste dos parâmetros do modelo durante o treinamento.
Tamanho do Modelo	Computacional	Espaço ocupado pelo modelo convertido para implementação embarcada.
Erro Percentual Médio (MAPE)	Embarcada	Erro relativo médio entre os valores estimados e os valores de referência.
Tempo de Inferência	Embarcada	Tempo necessário para que o microcontrolador produza uma predição após receber os dados de entrada.

Fonte: Elaborado pelo autor (2026).

As métricas computacionais foram utilizadas para comparar a capacidade preditiva e o custo computacional dos modelos durante o desenvolvimento em *Python*. Já as métricas embarcadas permitiram avaliar o comportamento dos modelos após sua implementação no ESP32, considerando aspectos de precisão e desempenho em tempo real.

3 RESULTADOS E DISCUSSÕES

Esta seção apresenta o desempenho dos modelos de aprendizado de máquina selecionados em ambiente computacional (treinamento, validação e teste) e embarcados bem como as discussões acerca dos resultados obtidos.

3.1 Ambiente Computacional

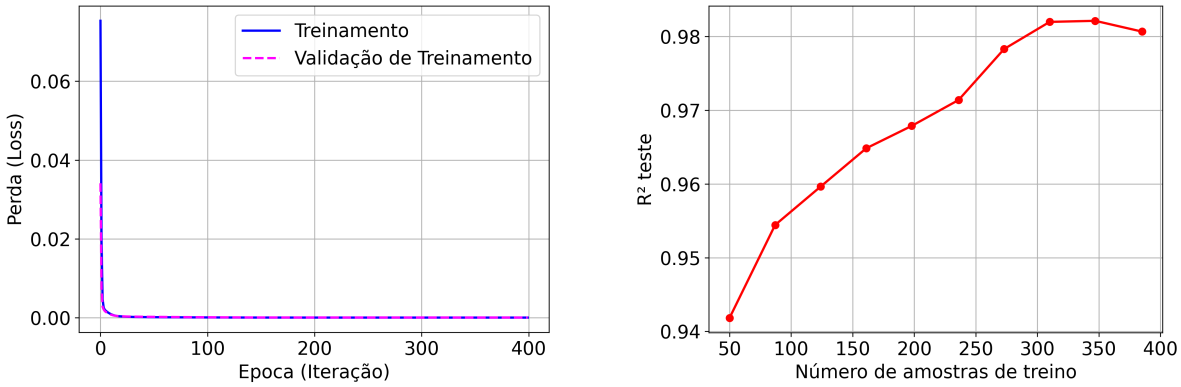
3.1.1 Aprendizado

Considerando o tamanho reduzido do conjunto de dados, buscou-se avaliar se a quantidade de amostras disponíveis era suficiente para que os modelos propostos, MLP e RF_{comp}, fossem capazes de aprender adequadamente os padrões presentes nos dados sem apresentar indícios de subajuste, do inglês (*underfitting*, ou sobreajuste.

Nesse contexto, a Figura 5 apresenta as curvas de aprendizado dos respectivos modelos, permitindo analisar o comportamento do erro de treinamento e validação (MLP) e a evolução do desempenho em função do número de amostras utilizadas no treinamento (RF_{comp}).

No caso da MLP, observa-se que após aproximadamente 20 iterações a perda durante o treinamento se estabiliza, indicando convergência do processo de otimização. Além disso,

Figura 5 – Curvas de aprendizado dos modelos avaliados.



(a) MLP

(b) RF_{comp}

nota-se que as curvas de treinamento e validação permanecem próximas ao longo do processo, sugerindo boa capacidade de generalização do modelo.

Para o RF_{comp}, a curva de aprendizado é obtida em função do número de amostras de treinamento, sendo o desempenho avaliado progressivamente a cada incremento de dados. Observa-se que, a partir de aproximadamente 300 amostras, o modelo atinge estabilidade no coeficiente de determinação (R^2), mantendo valores superiores a 98%, o que indica bom ajuste aos dados disponíveis.

3.1.2 Treinamento e Teste

A Tabela 4 apresenta as métricas de desempenho no ambiente computacional, nas etapas de treinamento e teste, para ambos os modelos.

Tabela 4 – Desempenho dos modelos em ambiente computacional.

Modelo	R^2_{treino}	R^2_{teste}	MSE	MAE	Tempo de Treino(s)	Tamanho (KB)
MLP	0,9223	0,9317	$7,92 \times 10^{-6}$	0,001995	37,80	2,71
RF_{comp}	0,9799	0,9807	$2,24 \times 10^{-6}$	0,001128	0,019	15,45
RF_{bm}	0,9802	0,9831	$1,96 \times 10^{-6}$	0,001087	16,30	597,55

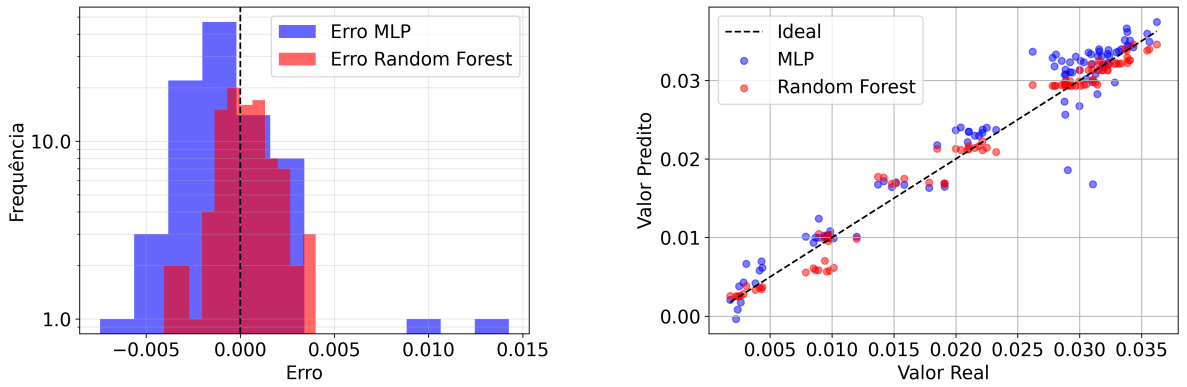
Observa-se que as arquiteturas do *Random Forest* apresentaram desempenhos semelhantes, sendo o RF_{bm} ligeiramente superior nas métricas de predição. Entretanto, esse ganho foi obtido à custa de um aumento significativo no tamanho do modelo. Como seu tamanho excede os 500 kB de memória disponíveis no microcontrolador utilizado, o RF_{bm} foi descartado para as etapas de implementação embarcada.

Dessa forma, o RF_{comp} mostrou-se a alternativa mais adequada, mantendo desempenho próximo ao do RF_{bm} com uma redução expressiva nos requisitos de armazenamento. Esse resultado evidencia que o modelo de melhor desempenho computacional não é necessariamente o mais adequado para aplicações embarcadas, nas quais as restrições de hardware devem ser consideradas juntamente com as métricas de predição.

Ao comparar a MLP e o RF_{comp}, verifica-se que o modelo baseado em árvores apresentou melhor desempenho geral. Embora o RF_{comp} possua tamanho aproximadamente seis vezes superior ao da MLP, ele permaneceu compatível com as limitações de memória do sistema embarcado.

As 97 amostras pertencentes ao conjunto de teste são apresentadas graficamente na Figura 6. Na Figura 6a, observa-se que ambos os modelos apresentaram erros reduzidos para a maior parte das amostras. Entretanto, a MLP apresentou maior dispersão e alguns desvios mais pronunciados, enquanto o RF_{comp} manteve erros mais uniformes ao longo do conjunto de teste. Esses desvios contribuem diretamente para o maior valor de MSE observado para a MLP.

Figura 6 – Desempenho na predição.



(a) Erro de predição.

(b) Valor real e preditivo

De forma complementar, a Figura 6b mostra que ambos os modelos acompanham adequadamente a tendência dos valores reais. Contudo, as predições do RF_{comp} permanecem visualmente mais próximas da linha de referência, corroborando o melhor desempenho observado nas métricas quantitativas.

3.2 Ambiente Embarcado

3.2.1 Verificação

A análise dos relatórios gerados pela função `Verify` da `Arduino IDE` permitiu avaliar a utilização dos recursos de memória da ESP32, que disponibiliza 1.310.720 bytes para armazenamento do programa em memória FLASH e 327.680 bytes de memória RAM dinâmica.

Para o *firmware* contendo o modelo MLP, observou-se uma ocupação de 93% da memória FLASH e 26% da memória RAM disponível (ver Apêndice E). Já o *firmware* baseado no modelo RF_{comp} utilizou 70% da memória FLASH e 14% da memória RAM (ver Apêndice F).

Os resultados evidenciam que ambos os modelos podem ser executados na plataforma embarcada adotada, permanecendo dentro dos limites de memória do microcontrolador. Entretanto, o modelo RF_{comp} apresentou menor consumo de memória em ambas as métricas

avaliadas, o que sugere maior adequação para implementação no microcontrolador empregado na planta experimental.

3.2.2 Desempenho

A Tabela 5 apresenta as métricas de desempenho no ambiente embarcado para ambos os modelos.

Tabela 5 – Desempenho dos modelos em ambiente embarcado.

Modelo	R^2	MAE	MSE	MAPE (%)	Tempo médio de inferência (ms)
MLP	0,9245	0,001794	$8,49 \times 10^{-6}$	16,44	0,700
RF_{comp}	0,9798	0,000926	$2,27 \times 10^{-6}$	7,97	0,624

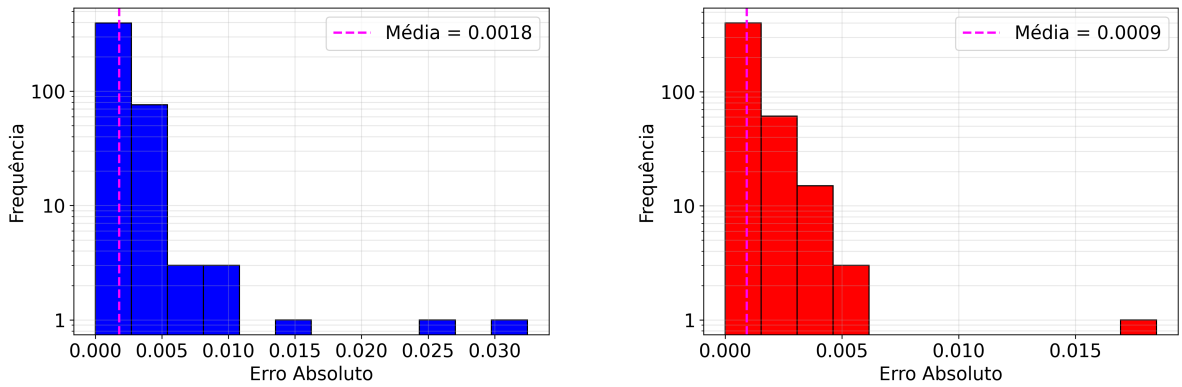
Observa-se que o modelo RF_{comp} apresentou desempenho superior ao MLP no ambiente embarcado, alcançando maior coeficiente de determinação (R^2) e menores valores de MAE, MSE e MAPE. Além disso, as métricas obtidas pelo RF_{comp} permaneceram próximas às observadas no ambiente computacional, indicando maior robustez ao processo de embarque.

Por outro lado, o modelo MLP apresentou aumento mais significativo dos erros, especialmente do MSE, sugerindo maior sensibilidade à implementação na plataforma embarcada e à presença de erros de maior magnitude nas predições. Como consequência, o MAPE obtido pelo MLP também foi superior ao observado para o RF_{comp}.

Quando comparados aos resultados obtidos em ambiente computacional, observa-se uma degradação mais pronunciada do desempenho do MLP após o embarque, evidenciada principalmente pelo aumento do MSE. Em contrapartida, o RF_{comp} manteve métricas muito próximas às originalmente obtidas, indicando maior capacidade de preservar o desempenho preditivo após a implementação no microcontrolador.

As Figuras 7 e 8 complementam os resultados apresentados na Tabela 5.

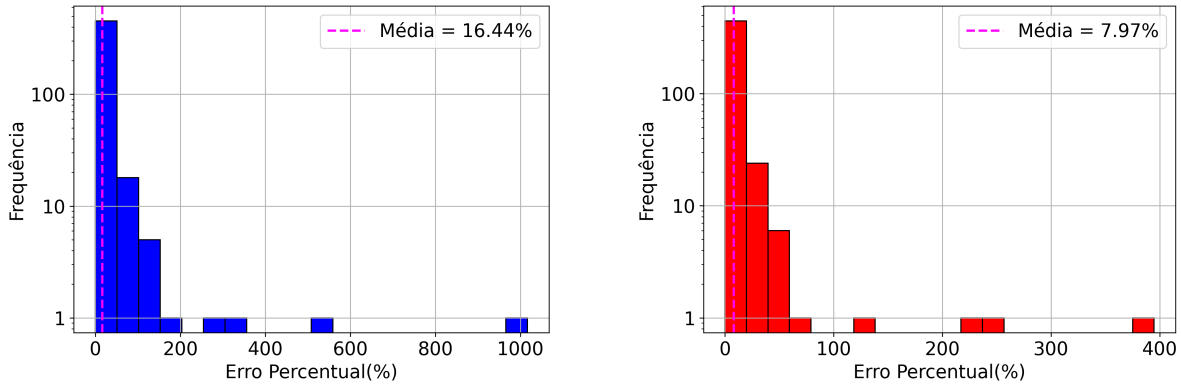
Figura 7 – Histograma do erro absoluto no ambiente embarcado.



(a) MLP

(b) RF_{comp}

Figura 8 – Histograma do erro percentual no ambiente embarcado.

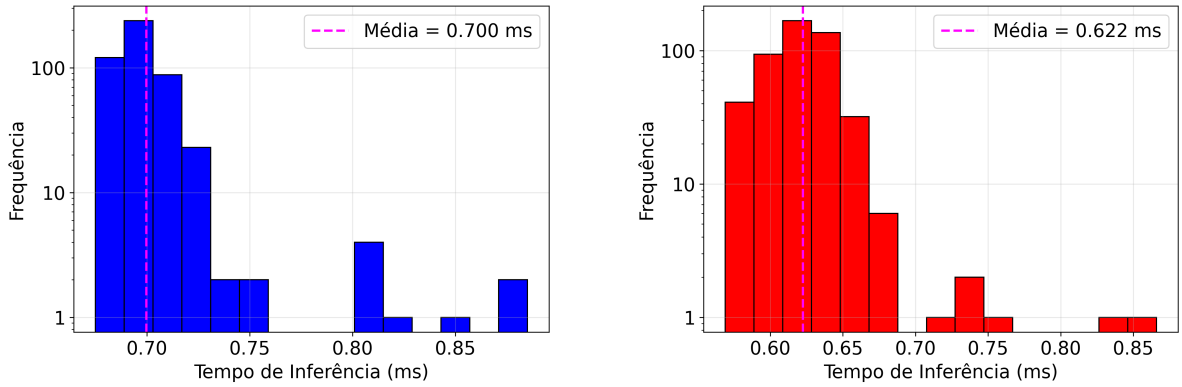


(a) MLP (b) RF_{comp}

Observa-se que o modelo MLP apresenta maior ocorrência de erros de elevada magnitude, evidenciada pela distribuição dos erros absolutos e percentuais. Além disso, nota-se que a amplitude dos erros do MLP é superior à observada para o RF_{comp}, justificando a apresentação dos histogramas em escalas distintas para facilitar a visualização e a interpretação dos resultados.

Em relação ao tempo de inferência, ambos os modelos apresentaram tempos médios superiores ao período de amostragem de 500 ms adotado na planta experimental. A análise das distribuições na Figura 9 mostra que os tempos de execução concentram-se predominantemente entre aproximadamente 0.5 ms e 1 ms . Dessa forma, o tempo de inferência não constitui uma limitação para a implementação do sensor virtual na plataforma embarcada utilizada. Assim, ambos os modelos mostraram-se viáveis para implementação na plataforma utilizada, embora o RF_{comp} tenha apresentado melhor equilíbrio entre desempenho preditivo e custo computacional.

Figura 9 – Tempo de inferência no ambiente embarcado.



(a) MLP (b) RF_{comp}

4 CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo desenvolver e avaliar um sensor virtual para estimação da vazão de gás em uma planta de bancada que emula um sistema de bicos injetores de alto-forno, comparando o desempenho de modelos baseados em aprendizado de máquina em ambiente computacional e embarcado. Para isso, foram investigados e comparados os modelos *Perceptron* Multicamadas e *Random Forest*, considerando tanto a qualidade das predições quanto os requisitos de implementação embarcada.

Os resultados obtidos demonstraram que ambos os modelos foram capazes de estimar adequadamente a variável de interesse, apresentando elevados coefientes de determinação (superiores a 90%) e baixos erros de predição. Entretanto, o modelo *Random Forest* apresentou desempenho superior nos ambientes computacional e embarcado em relação ao *Perceptron* Multicamadas, demonstrando maior robustez frente ao conjunto de dados disponível e às restrições computacionais da plataforma embarcada. Além de apresentar menores valores de MAE, MSE e MAPE, o *Random Forest* manteve desempenho preditivo praticamente inalterado após o embarque, enquanto o MLP apresentou degradação mais significativa das métricas de erro.

A análise dos recursos computacionais também evidenciou a adequação do modelo *Random Forest* para aplicações embarcadas, uma vez que demandou menor utilização de memória *FLASH* e RAM quando comparado ao MLP. Além disso, ambos os modelos apresentaram tempos médios de inferência significativamente inferiores ao período de amostragem de 500 ms adotado na planta experimental, demonstrando que o processamento pode ser realizado com ampla margem temporal em relação à aquisição dos dados. Dessa forma, verificou-se a viabilidade de sua implementação na plataforma embarcada utilizada.

Como principal contribuição científica, este trabalho demonstrou a aplicação de técnicas de aprendizado de máquina no desenvolvimento de sensores virtuais para estimação de vazão de gás em uma planta experimental, contemplando desde a etapa de aquisição e tratamento dos dados até a implementação e validação dos modelos em *hardware* real. Além disso, a comparação entre diferentes algoritmos em ambientes computacional e embarcado fornece subsídios para a seleção de modelos em aplicações sujeitas a restrições de processamento e memória.

Como propostas para trabalhos futuros, destacam-se: (i) a avaliação de modelos de aprendizado de máquina voltados à representação de dinâmicas temporais, como NARX e LSTM; (ii) a obtenção de conjuntos de dados mais abrangentes, contendo diferentes condições operacionais distribuídas de forma mais uniforme ao longo da série temporal, permitindo uma avaliação mais robusta da capacidade de generalização dos modelos; e (iii) a adoção de modelos adaptativos capazes de atualizar seus parâmetros com base nos dados coletados ao longo da operação, bem como o emprego de técnicas de quantização para reduzir ainda mais o tamanho dos modelos embarcados.

REFERÊNCIAS

ATANANE, O. et al. Smart buildings: Water leakage detection using tinyml. *Sensors*, 2023. Disponível em: <https://www.mdpi.com/1424-8220/23/22/9210>. Acesso em: 09 abr. 2026.

BELO, E. d. O. *Análise de falhas dos equipamentos de um alto forno*. Curitiba: [s.n.], 2019. Monografia (Especialização em Engenharia da Confiabilidade). Disponível em: <https://riut.utfpr.edu.br/jspui/bitstream/1/18633/1/CT_CEECVIT_II_2019_07.pdf>. Acesso em: 09 abr. 2026.

BEZERRA, A. R. d. C. F. *Estudo comparativo de medidores de vazão: características e aplicações*. Natal: [s.n.], 2024. Trabalho de Conclusão de Curso (Graduação em Engenharia Química). Disponível em: <https://repositorio.ufrn.br/server/api/core/bitstreams/b51653b4-f90e-479f-81cc-512c5661c78a/content>. Acesso em: 09 abr. 2026.

CARVALHO, T. C. O. et al. Desenvolvimento e análise de desempenho de um sistema iot para cálculo do coeficiente de descarga de bicos injetores. In: *Simpósio Brasileiro de Automação Inteligente (SBAI)*. [S.l.: s.n.], 2025. Disponível em: <https://www.sba.org.br/open_journal_systems/index.php/sbai/article/view/5303>. Acesso em: 03 mai. 2026.

CHAORAINGERN, J.; TIPSUWANPORN, V.; NUMSOMRAN, A. Artificial intelligence for the classification of plastic waste utilizing tinyml on low-cost embedded systems. *International Journal on Advanced Science Engineering Information Technology*, v. 13, n. 6, 2023. Disponível em: <https://pdfs.semanticscholar.org/e26b/626ec32d6dc39ce167b4657af22e365cf4e3.pdf>. Acesso em: 09 abr. 2026.

CHENG, Q.; CHUNHONG, Z.; QIANGLIN, L. Development an application of random forest regression soft sensor model for treating domestic wastewater in a sequencing batch reactor. *Scientific Reports*, v. 13, n. 9149, 2023. Disponível em: <https://www.nature.com/articles/s41598-023-36333-8>. Acesso em: 27 mai. 2026.

DRUMOND, J. G. M. *Avaliação de Modelos de TinyML para Inferência Embarcada de Vazão de Gás no ESP32*. Belo Horizonte: [s.n.], 2025. Trabalho de Conclusão de Curso em Engenharia Elétrica. Material disponibilizado pelo orientador. Uso interno.

EFFENDY, N. et al. The prediction of the oxygen content of the flue gas in a gasfired boiler system using neural networks and random forest. *IAES International Journal of Artificial Intelligence*, v. 11, n. 3, 2022. Disponível em: <https://www.researchgate.net/publication/361163729>. Acesso em: 09 abr. 2026.

JUVÊNCIO, M. H. C. *Redes Neurais e SVM Aplicadas no Desenvolvimento de Sensores Virtuais e Detecção e Diagnóstico de Falhas em Processo Reacionais Complexos*. Alagoas: [s.n.], 2023. Trabalho de Conclusão de Curso em Engenharia Química. Disponível em: <https://www.repositorio.ufal.br/handle/123456789/11872>. Acesso em: 09 abr. 2026.

LIMA, J. d. S. *Modelagem de Sensor Virtual para Medição de Vazão em uma Usina do Setor Sucroenergético Baseado em Redes Neurais Artificiais*. Dissertação (Mestrado) — Universidade Federal da Paraíba, João Pessoa, 2022. Dissertação de Mestrado em Engenharia Elétrica. Disponível em: <https://repositorio.ufpb.br/jspui/handle/123456789/25791?locale=pt_BR>. Acesso em: 09 abr. 2026.

LIMA, R. P. G. *Desenvolvimento de um soft sensor para estimação da vazão em sistemas de abastecimento de água utilizando redes neurais artificiais*. Tese (Doutorado) — Universidade Federal da Paraíba, João Pessoa, 2022. Disponível em: <https://repositorio.ufpb.br/jspui/handle/123456789/24485>. Acesso em: 09 abr. 2026.

MAJESKI, A. et al. Injection of pulverized coal and natural gas into blast for iron-making: lance positioning and design. *ISIJ International*, v. 55, 2015. Disponível em: <https://www.jstage.jst.go.jp/article/isijinternational/55/7/55_1377/_pdf/-char/en>. Acesso em: 09 abr. 2026.

MARTINS, T. M.; REIS, A. J. d. R.; PÁBON, R. E. C. Uso de sensores virtuais para estimativa do fluxo de minério de ferro em correias transportadoras. In: *Congresso Brasileiro de Inteligência Computacional (CBIC)*. [S.l.: s.n.], 2025. Disponível em: <https://sbia.org.br/eventos/cbic_2025/cbic2025-1191340/>. Acesso em: 25 mai. 2026.

MATHEUS, R. C. M. *Sensores Virtuais Utilizando Redes Neurais Artificiais para a Estimação da Consistência e do Freeness de Celulose*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2024. Dissertação de Mestrado em Engenharia de Produção. Disponível em: <https://repositorio.utfpr.edu.br/jspui/handle/1/35614>. Acesso em: 09 abr. 2026.

MATHEUS, R. C. M. et al. Sensor virtual para estimação de consistência de pasta termomecânica de celulose. In: *Brazilian Conference on Computational Intelligence*. Salvador: CBIC, 2023. Disponível em: <https://sbia.org.br/eventos/cbic_2023/cbic2023-153/>. Acesso em: 09 abr. 2026.

QUEIROZ, D. G. *Desenvolvimento de Sensores Virtuais para Predição da Composição dos Efluentes de uma Unidade de Tratamento de Águas Ácidas*. Rio de Janeiro: [s.n.], 2023. Trabalho de Conclusão de Curso em Engenharia Química. Disponível em: <https://pantheon.ufrj.br/handle/11422/23246>. Acesso em: 25 mai. 2026.

SILVA, J. D. d. *Aprendizagem de máquina aplicada ao desenvolvimento de sensores virtuais e na detecção e diagnóstico de falhas para processos químicos reativos em linguagem Python*. Alagoas: [s.n.], 2022. Trabalho de Conclusão de Curso (Graduação em Engenharia Química). Disponível em: <https://www.repositorio.ufal.br/handle/123456789/16101>. Acesso em: 25 mai. 2026.

SILVA, M. T. B. d. *Aprendizado de Máquina em Microcontroladores utilizando TinyML: Exploração, Otimização e Portabilidade*. Salvador: [s.n.], 2023. Trabalho de Conclusão de Curso em Engenharia Elétrica. Disponível em: <https://repositorio.ufba.br/handle/ri/39091>. Acesso em: 09 abr. 2026.

THURUTHEL, T. G.; GARDNER, P.; ILDA, F. Closing the control loop with time-variant embedded soft sensors and recurrent neural networks. *Soft Robotics*, v. 9, n. 6, 2022. Disponível em: <https://journals.sagepub.com/doi/full/10.1089/soro.2021.0012>. Acesso em: 09 abr. 2026.

WASEM, L. A.; SATHLER, F.; GENEROSO, L. G. Co-injeção de gás natural e carvão pulverizado no alto-forno 3 da arcelormittal tubarão: resultados operacionais. In: *51° Seminário de Redução de Minérios e Matérias-Primas*. São Paulo: [s.n.], 2023. Disponível

em: <https://abmproceedings.com.br/pt-br/publications/anais-dos-seminarios-de-reducao-minerio-de-ferro-e-aglomeracao/event/anais-dos-seminarios-de-reducao-minerio-de-ferro-e-aglomeracao/article/co-injeo-de-gs-natural-e-carvo-pulverizado-no-alto-forno-3-da-arcelormittal-tubaro-resultados-operacionais>. Acesso em: 09 abr. 2026.

ZÁHONYI, P. et al. Explainable artificial neural network as a soft sensor to predict the moisture content in a continuous granulation line. *European Journal of Pharmaceutical Sciences*, Elsevier, 2025. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0928098725001721>. Acesso em: 09 abr. 2026.

APÊNDICES

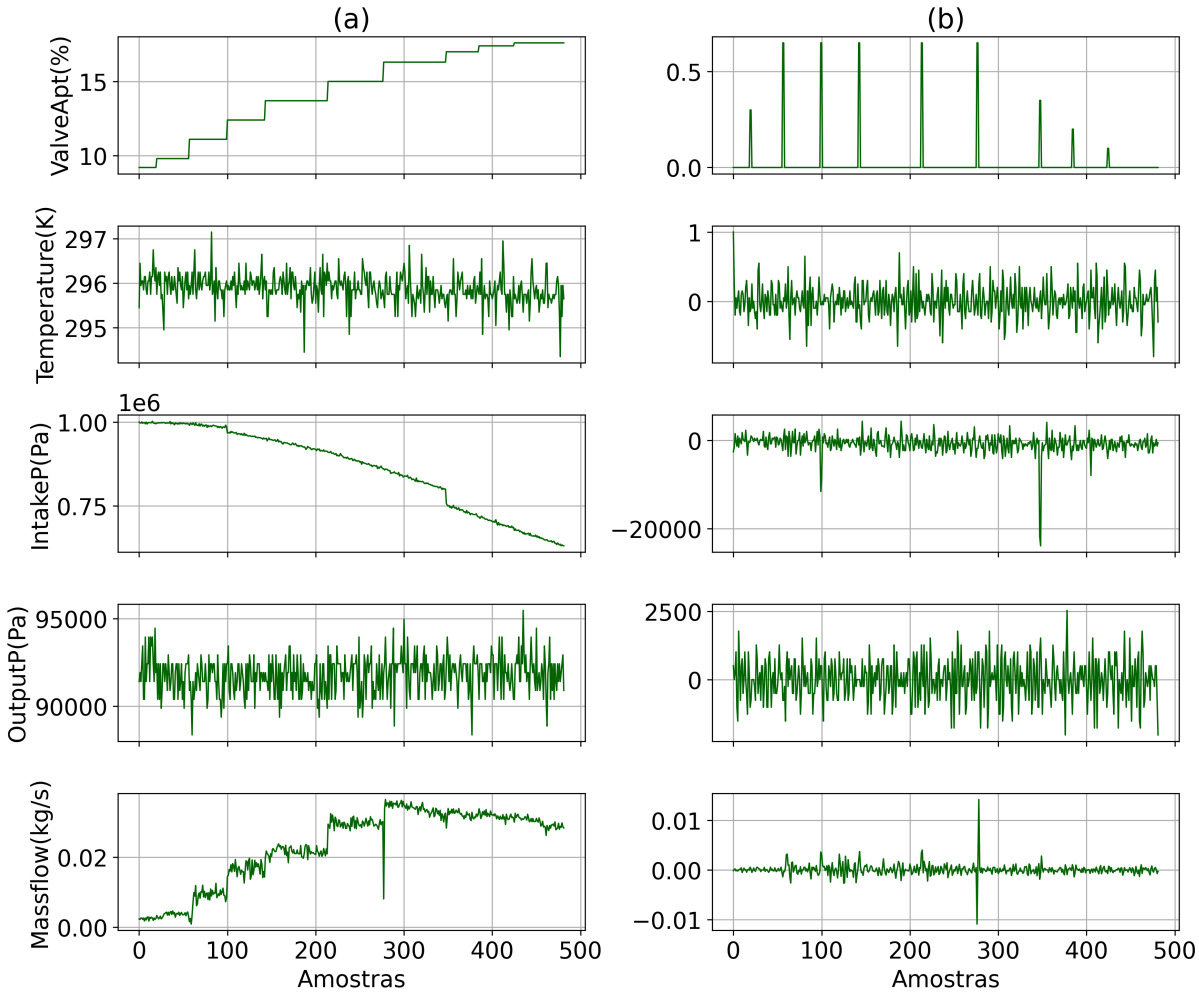
A Análise Estatística e Temporal dos Dados

Tabela 6 – Coeficientes de correlação entre a vazão mássica e as variáveis monitoradas

Variável	**Correlação**
Time (s)	0.843282
Massflow (kg/s)	1.000000
ValveApt (%)	0.944307
ValveAng (°)	0.944307
ValveArea (m^2)	0.942071
Temperature (K)	-0.196469
IntakeP (Pa)	-0.758393
OutputP (Pa)	0.107944
ReynoldsD	0.999998
CD	0.990847
KV	0.827325

Fonte: Elaborado pelo autor (2026).

Figura 10 – Análise temporal: (a) Perfis Temporais e (b) Derivadas discretas.



Fonte: Elaborado pelo autor (2026).

B *Script* em `Python` para processamento e treinamento dos dados

O código-fonte completo referente ao treinamento dos modelos utilizados neste trabalho encontra-se disponível em: <https://github.com/Sapalhi/SensorVirtual/blob/main/Treino_modelos_real.py>

C *Script* do modelo `Random Forest`

O código-fonte completo da implementação do modelo *Random Forest* por meio da validação cruzada encontra-se disponível em: <https://github.com/Sapalhi/SensorVirtual/blob/main/RF_completo.py>

D *Script* de comunicação MQTT

O código-fonte completo referente à implementação da comunicação MQTT encontra-se disponível em: <https://github.com/Sapalhi/SensorVirtual/blob/main/ProtocoloMQTT.py>

E Código ESP32 para o modelo *Perceptron* Multicamadas

O código-fonte completo da implementação embarcada encontra-se disponível em: <https://github.com/Sapalhi/SensorVirtual/blob/main/embarcado_mlp/embarcado_mlp.ino>

F Código ESP32 para o modelo *Random Forest*

O código-fonte completo da implementação embarcada encontra-se disponível em: <https://github.com/Sapalhi/SensorVirtual/blob/main/embarcado_rf/embarcado_rf.ino>