UNIVERSIDADE FEDERAL DE ALAGOAS
CENTRO DE TECNOLOGIA – CTEC
ENGENHARIA QUÍMICA

Mario Henrique Cosme Juvêncio

REDES NEURAIS E SVM APLICADAS NO DESENVOLVIMENTO DE SENSORES VIRTUAIS E DETECÇÃO E DIAGNÓSTICO DE FALHAS EM PROCESSO REACIONAIS COMPLEXOS

MACEIÓ – AL
2023

MARIO HENRIQUE COSME JUVÊNCIO

REDES NEURAIS E SVM APLICADAS NO DESENVOLVIMENTO DE SENSORES VIRTUAIS E DETECÇÃO E DIAGNÓSTICO DE FALHAS EM PROCESSOS REACIONAIS COMPLEXOS

Trabalho de conclusão de curso apresentado ao curso de Engenharia Química da Universidade Federal de Alagoas, como requisito parcial para obtenção do título de Bacharel em Engenharia Química.

Orientador: Prof. Dr. William Imamura.
Coorientador: Prof. Dr. Frede de Oliveira Carvalho.

MACEIÓ – AL
2023

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária: Antonia Izabel da Silva Meyer – CRB-4 – 1558

J97r Juvêncio, Mario Henrique Cosme.
Redes neurais e SVM aplicadas no desenvolvimento de sensores virtuais e detecção e diagnóstico de falhas em processo reacionais complexos / Mario Henrique Cosme Juvêncio. – 2023.
51 f. : il. tabs e grafs, color.

Orientador: William Imamura.
Coorientador: Frede de Oliveira Carvalho.
Monografia (Trabalho de Conclusão de Curso em Engenharia Química) – Universidade Federal de Alagoas. Centro de Tecnologia. Maceió, 2023.

Bibliografia: f. 49-51.

1. Redes Neurais (Computação). 2. Python (Linguagem de programação de computador). 3. Máquinas de Vetores de Suporte. 4. Engenharia química. I. Título.

CDU: 66.0+004.89

UNIVERSIDADE FEDERAL DE ALAGOAS - UFAL
UNIDADE ACADÊMICA CENTRO DE TECNOLOGIA
CURSO DE ENGENHARIA QUÍMICA

MARIO HENRIQUE COSME JUVENCIO

REDES NEURAIS E SVM APLICADAS NO DESENVOLVIMENTO DE SENSORES VIRTUAIS E DETECÇÃO E DIAGNÓSTICO DE FALHAS EM PROCESSOS REACIONAIS COMPLEXOS

BANCA EXAMINADORA

_____

Prof. Dr. William Imamura

_____

Prof. Dr. Frede de Oliveira Carvalho

_____

Prof. Dr. Rodolfo Junqueira Brandão

_____

Prof. Dr. João Batista Maia Rocha Neto

AGRADECIMENTOS

Primeiramente, gostaria de agradecer a Deus por ter me concedido força, sabedoria e resiliência ao longo desta jornada acadêmica.

Aos meus pais, Denize e Ananias, agradeço por todo o amor, apoio e incentivo incondicional que me deram ao longo da minha vida.

À minha namorada, Agda Guimarães, agradeço pelo carinho, amor, compreensão, incentivo e paciência durante os momentos difíceis enfrentados ao longo da minha vida acadêmica.

Expresso minha gratidão à minha tia Thereza, às minhas primas Liss Azevedo e Clara Azevedo, pelo apoio dado durante o período de estágio obrigatório. Vocês foram fundamentais e estiveram presentes em um dos momentos mais desafiadores da minha graduação.

Aos meus colegas de curso e colegas de estágio, agradeço pela parceria, aprendizado mútuo e amizade, que tornaram essa caminhada mais leve e enriquecedora.

Aos professores do curso de Engenharia Química da Universidade Federal de Alagoas, sou grato por todo o conhecimento transmitido, paciência e dedicação ao longo da minha formação.

Um agradecimento especial aos meus orientadores, Frede Carvalho e William Imamura, que me guiaram com sabedoria e paciência durante a elaboração deste trabalho.

Por fim, agradeço a todos que, direta ou indiretamente, contribuíram para a minha formação e realização deste sonho.

RESUMO

A evolução da indústria ao longo dos séculos tem sido marcada por revoluções que transformaram a forma como os produtos são fabricados e os processos são gerenciados. A Quarta Revolução Industrial, também conhecida como Indústria 4.0, representa um salto significativo em direção a sistemas de produção inteligentes e conectados, onde a integração de tecnologia de ponta desempenha um papel crucial no aumento da eficiência, produtividade e segurança dos processos industriais. Neste contexto, o desenvolvimento de sensores virtuais e técnicas de detecção e diagnóstico de falhas em processos se destaca como uma área de pesquisa vital, com potencial para melhorar o controle e a segurança dos processos. Nesse sentido, este trabalho teve como finalidade o desenvolvimento de algoritmos computacionais em Python com o objetivo de modelar um sistema reacional complexo com foco na criação de sensores virtuais e modelos de detecção e diagnóstico de falhas, utilizando Redes Neurais Artificiais (RNA) e Máquinas de Vetores de Suporte (SVM). A modelagem matemática de um reator CSTR, operando com a reação de Van de Vusse, foi implementada em Python e empregada na geração de dados históricos do processo que serviram de base para o desenvolvimento dos sensores virtuais e modelos de detecção e diagnóstico de falhas. Os resultados indicam que os modelos de RNA apresentaram alta precisão e eficácia na estimação da concentração do produto B, enquanto os modelos de SVM mostraram-se inadequados para essa tarefa. Na detecção e diagnóstico de falhas, o modelo de RNA obteve uma acurácia de 97%, com desempenho superior na detecção de falhas em comparação ao de SVM. Por outro lado, o SVM apresentou melhor desempenho no diagnóstico das falhas, identificando corretamente os tipos específicos em mais de 99,98% dos casos. Uma possível melhoria no processo seria a integração das técnicas, utilizando RNA na detecção e SVM na identificação do tipo de falha. Os modelos e estratégias propostas podem ser aplicadas para aprimorar o controle e monitoramento de processos reacionais complexos e otimizar o desempenho dos reatores.

Palavras-Chave: Redes Neurais Artificiais; Máquinas de Vetores de Suporte; Sensores Virtuais; Detecção e Diagnóstico de falhas; Reação de Van de Vusse; Python.

ABSTRACT

The evolution of industry throughout the centuries has been marked by revolutions that have transformed the way products are manufactured and processes are managed. The Fourth Industrial Revolution, also known as Industry 4.0, represents a significant leap towards intelligent and connected production systems, where the integration of cutting-edge technology plays a crucial role in increasing the efficiency, productivity, and safety of industrial processes. In this context, the development of virtual sensors and fault detection and diagnosis techniques in processes stands out as a vital area of research, with the potential to improve process control and safety. In this regard, this work aimed to develop computational algorithms in Python with the objective of modeling a complex reaction system focusing on the creation of virtual sensors and fault detection and diagnosis models, using Artificial Neural Networks (ANN) and Support Vector Machines (SVM). The mathematical modeling of a CSTR reactor, operating with the Van de Vusse reaction, was implemented in Python and used to generate historical process data that served as the basis for the development of virtual sensors and fault detection and diagnosis models. The results indicate that the ANN models showed high accuracy and effectiveness in estimating the concentration of product B, while the SVM models proved unsuitable for this task. In fault detection and diagnosis, the ANN model achieved 97% accuracy, with superior performance in fault detection compared to SVM. On the other hand, SVM showed better performance in fault diagnosis, correctly identifying the specific types in more than 99.98% of cases. A possible improvement in the process would be the integration of the techniques, using ANN for detection and SVM for identifying the type of fault. The proposed models and strategies can be applied to enhance the control and monitoring of complex reaction processes and optimize reactor performance.

Keywords: Artificial Neural Networks; Support Vector Machines; Virtual Sensors; Fault Detection and Diagnosis; Van de Vusse Reaction; Python.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

RNA	Redes Neurais Artificiais
SVM	Máquinas de Vetores de Suporte
CSTR	*Continuos Flow Stirred Tank Reactors*
PFR	*Plug Flow Reactor*
IDE	Ambiente de Desenvolvimento Integrado
MSE	*Mean Squared Error*
MAE	*Mean Absolute Error*
RELU	*Rectified Linear Unit*
SGD	*Stochastic Gradient Descent*
RMSProp	*Root Mean Square Propagation*
Adagrad	*Adaptive Gradient*
Adam	*Adaptive Moment Estimation*

SUMÁRIO

1 INTRODUÇÃO

Com o advento da Indústria 4.0 e o progresso das tecnologias para extração e armazenamento dos dados, as indústrias têm passado por um processo de modernização ao longo dos últimos anos, visando principalmente o aumento da produtividade e aprimoramento da segurança dos processos. Consequentemente, testemunhou-se um expressivo crescimento no volume de dados armazenados, originados do monitoramento online do processo por meio de sensores físicos. A expansão dos dados históricos gerados e armazenados, aliada ao desenvolvimento das técnicas de aprendizado de máquina, viabilizou pesquisas para a criação de ferramentas de sensores virtuais e detecção e diagnóstico de falhas (KADLEC et al., 2009).

Sensores virtuais são modelos preditivos que podem ser aplicados para prever variáveis que são cruciais para o gerenciamento e controle do processo, porém de difícil mensuração ou que demandam análises demoradas e custosas. Os sensores virtuais, em conjunto com detectores automáticos de falhas, desempenham um papel importante na detecção de falhas em processos químicos, uma vez que se fundamentam em dados históricos do processo. Assim, quaisquer desvios nas condições normais de operação são prontamente detectados e diagnosticados, embasando decisões rápidas e assertivas dos operadores da planta e contribuindo para a segurança do processo e a qualidade do produto (KADLEC et al., 2009). As técnicas de Redes Neurais Artificiais (RNA) e Máquinas de Vetores de Suporte (SVM) têm sido aplicadas nesta área, possibilitando a identificação de anomalias nos sistemas sem a necessidade de modelos matemáticos precisos do processo (TAQVI *et al*., 2021).

Os sensores virtuais, detectores automáticos de falhas e sistemas de diagnóstico de falhas podem ser construídos com base em técnicas de aprendizado de máquina supervisionado. Essas técnicas exploram o conhecimento histórico acerca do comportamento das variáveis do processo sob diversas condições de operação, tornando-as aptas a estimar valores de variáveis de interesse, identificar rapidamente os desvios e realizar o diagnóstico por meio de informações sobre as demais variáveis. A aplicação de técnicas de inteligência artificial, como as Redes Neurais Artificiais e Máquinas de Vetores de Suporte, tem sido fundamental no desenvolvimento desses modelos (CACCAVALE et al., 2009).

Neste contexto, o presente trabalho tem o objetivo de implementar, em linguagem Python, modelos de sensores virtuais e estratégias de detecção e diagnóstico de falhas para o reator CSTR (*Continuos flow Stirred Tank Reactors),* operando com a reação de Van de Vusse, empregando técnicas de Redes Neurais Artificiais e Máquinas de Vetores de Suporte.

2 OBJETIVOS

2.1 Geral

Desenvolver modelos computacionais em linguagem Python destinados à simulação de um sistema reacional complexo, visando o emprego de técnicas de aprendizado de máquina supervisionado no desenvolvimento de sensores virtuais e estratégias de detecção e diagnóstico de falhas.

2.2 Específicos

- Desenvolver e implementar uma modelagem matemática de um reator CSTR, que opere com a reação de Van de Vusse, utilizando a linguagem de programação Python;
- Criar modelos de sensores virtuais para predição de variáveis do processo a partir de técnicas de RNA e SVM;
- Aplicar estratégias de detecção e diagnóstico de falhas ao processo não-linear, recorrendo a técnicas de aprendizado de máquina supervisionado de RNA e SVM;
- Avaliar e validar os modelos de sensores virtuais e as estratégias de detecção e diagnóstico de falhas, por meio de simulações computacionais e análises de métricas de desempenho.

3 REVISÃO BIBLIOGRÁFICA

3.1 Indústria 4.0

Desde o fim do século XVIII, o setor industrial tem sido de extrema importância para o progresso econômica dos países. A indústria sofreu mudanças significativas que transformaram completamente a forma como os produtos são fabricados, trazendo consigo diversos benefícios, principalmente no que diz respeito à melhoria da eficiência produtiva.

Durante a Primeira Revolução Industrial, ocorrida a partir do final do século XVIII, duas importantes invenções transformaram o setor produtivo e de transporte: a descoberta da utilização do carvão como fonte de energia e o desenvolvimento das máquinas a vapor, aliadas às linhas férreas. Essas mudanças foram responsáveis por uma transição do trabalho manual para a utilização de máquinas industriais (VIAN, 2015).

No início do século XX, a introdução da eletricidade nos sistemas produtivos foi um marco na história da indústria e deu início à Segunda Revolução Industrial. Segundo Mata *et al*. (2018), a utilização da eletricidade proporcionou uma produção em larga escala de produtos padronizados, como os automóveis, o que antes era impossível com as antigas formas de produção.

A Terceira Revolução Industrial teve início na década de 1970 e representou uma importante evolução no sistema de produção. Segundo Brito (2017), essas tecnologias permitiram a criação de sistemas de automação mais sofisticados, com maior precisão e flexibilidade, capazes de se adaptar às demandas e mudanças do mercado. Além disso, permitiu uma maior interconexão entre os sistemas, aumentando a eficiência e produtividade (SCHWAB, 2016).

A Quarta Revolução Industrial, também conhecida como Indústria 4.0, foi apresentada pela primeira vez em 2011, na Alemanha, durante a feira de Hannover com o objetivo de promover uma nova tendência industrial baseada em tecnologias de ponta, como destacado por Mata *et al*. (2018). A Indústria 4.0 tem como foco principal os sistemas de produção, nos quais sensores orientam as máquinas sobre como operar e os processos são governados por si mesmos em um sistema modular descentralizado, com coleta de dados e informações em tempo real, como mencionado por Harrison, Vera e Ahmad (2016). Nesse sentido, a Indústria 4.0 visa transformar o sistema industrial em um ambiente inteligente e conectado, por meio de nove pilares principais: *Big Data*, Robôs Autônomos, Simulação, Integração Vertical e Horizontal,

Internet das Coisas (IoT), Segurança Cibernética, Nuvem, Manufatura Aditiva e Realidade Aumentada. Além desses pilares, a aplicação de tecnologias de inteligência artificial também desempenha um papel fundamental em sua implementação (CNI, 2017; RÜßMANN et al., 2015).

3.2 Sensores Virtuais

Os sensores virtuais são elementos cruciais em aplicações industriais, sendo empregados no monitoramento e controle de processos por meio da estimativa de variáveis em tempo real. Além disso, podem atuar como sistemas de *backup* para dispositivos de medição, sendo colocados em ação quando o hardware apresenta falhas ou requer intervenções de manutenção (FORTUNA et al.,2006).

Em geral, as plantas industriais são dotadas de muitos instrumentos denominados sensores, capazes de fornecer dados em tempo real sobre o processo e permitir o monitoramento e controle do processo. Nas últimas décadas, devido à grande quantidade de dados gerados e armazenados na indústria, os dados começaram a ser utilizados no desenvolvimento de modelos preditivos orientados a dados, denominados de sensores virtuais. Tais modelos são capazes de predizer variáveis de interesse a partir de informações sobre outras variáveis do processo. Os modelos podem ser construídos a partir de técnicas de máquina de aprendizado como RNA e SVM (KADLEC et al., 2009).

3.3 Detecção e Diagnóstico de Falhas

De acordo com Himmelblau (1978), o termo falha pode ser entendido como um desvio de um intervalo aceitável de uma variável observada ou um parâmetro calculado associado ao processo. Nesse contexto, a tarefa de detecção e diagnóstico de falhas envolve a identificação do desvio no processo, da causa e a localização das falhas detectadas. As técnicas de detecção e diagnóstico de falhas incluem o uso de modelos determinísticos e modelos baseados em dados históricos do processo, onde podem ser empregadas as técnicas de RNA e SVM. Os modelos determinísticos exigem o conhecimento da modelagem matemática do processo dinâmico, o que muitas vezes é difícil a identificação devido à complexidade do processo. As RNA e SVM, por sua vez, são algoritmos de aprendizado de máquina que podem ser treinados a partir de dados históricos para aprender o padrão de comportamento do processo e identificar anomalias

no sistema, não dependendo de modelos matemáticos precisos do processo. (FORTUNA *et al*.,2006).

De modo geral, os métodos de diagnóstico de falhas podem ser agrupados em três classes diferentes: métodos baseados em modelos quantitativos, métodos baseados em modelos qualitativos e métodos baseados nos dados históricos do processo, utilizando as máquinas de aprendizado supervisionado (VENKATASUBRAMANIAN *et al*., 2003).

A abordagem baseada em dados históricos é caracterizada pela necessidade de coleta e armazenamento de grandes quantidades de dados para serem utilizados no desenvolvimento de modelos baseados em aprendizado de máquina. Porém, nos últimos anos, esse problema tem sido minimizado com o uso de tecnologias de computação, bancos de dados e técnicas de mineração de dados. Além da quantidade, a eficácia dos modelos também depende da qualidade dos dados gerados, sendo necessário o tratamento adequado dos ruídos nos dados. No contexto das máquinas de aprendizado, o problema de detecção e diagnóstico de falhas pode ser modelados com um problema de classificação, onde cada amostra de dados é associada ao tipo específico de falha por meio de rótulos pré-definidos (TAQVI *et al*., 2021).

Dessa forma, o trabalho estudou o emprego das técnicas de RNA e SVM, aprendizado de máquina supervisionado, para detecção e diagnóstico de falhas em sensores físicos de um reator CSTR operando com a reação de Van de Vusse.

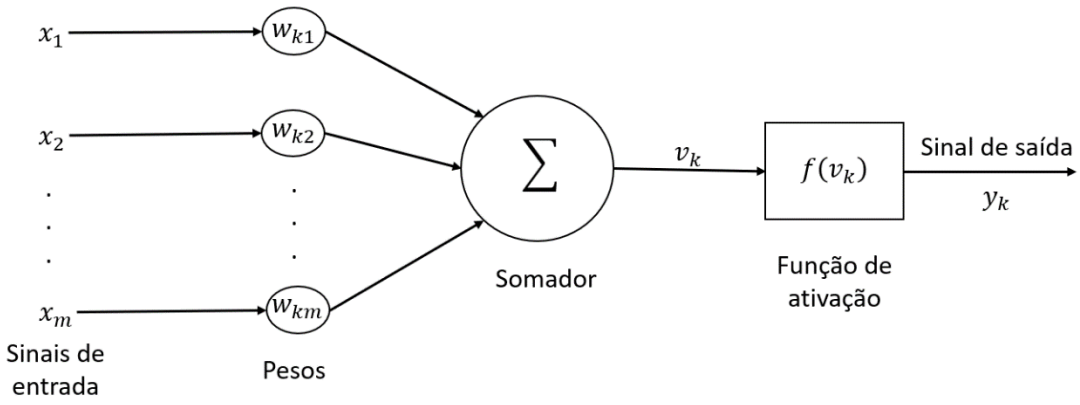
3.4 Rede Neural Artificial

Na literatura, as redes neurais artificiais têm sido amplamente estudadas e aplicadas em diversos campos de pesquisa. Conforme Haykin (2001), essas redes foram desenvolvidas com base no conhecimento do funcionamento do cérebro humano, buscando simular o processo de aprendizagem e processamento de informações realizados pelo cérebro. As redes neurais são compostas por unidades de processamento simples, denominadas neurônios, que se comunicam entre si através de conexões sinápticas.

Haykin (2001) destaca que as conexões sinápticas são caracterizadas por pesos sinápticos que têm a capacidade de armazenar o conhecimento adquirido durante o processo de aprendizagem supervisionada. Ajustando-se os pesos sinápticos, é possível alcançar objetivos pré-estabelecidos, permitindo que as redes neurais aprendam padrões e realizem generalizações. Isso resulta na capacidade de produzir saídas satisfatórias para entradas que não fazem parte do conjunto de treinamento na etapa de aprendizagem.

As redes neurais podem ser compostas por um ou mais neurônios, responsáveis pelo processamento de informações. O modelo neuronal é constituído por três componentes fundamentais: um conjunto de sinapses, associadas a pesos; um somador, responsável por somar os sinais de entrada ponderados pelos pesos sinápticos; e uma função de ativação que restringe a amplitude do sinal de saída do neurônio (Haykin, 2001). A Figura 1 ilustra um modelo de neurônio com os sinais de entrada $(x_1, x_2, x_3, \dots, x_m)$, as sinapses $(w_{k1}, w_{k2}, w_{k3}, \dots, w_{km})$ a função de ativação $f(v_k)$ e o sinal de saída do neurônio.

Figura 1 – Modelo de um neurônio artificial.



Fonte: Adaptado de Haykin (2001).

Matematicamente, um neurônio k pode ser representado pela equação (1):

$$y_k = f\left(\sum_{j=1}^{m} w_{kj} x_j\right) \tag{1}$$

em que x_j é o sinal de entrada j, w_{kj} são os pesos sinápticos associados ao sinal de entrada j do neurônio k, e y_k é o sinal de saída do neurônio k (Haykin, 2001).

Alguns hiperparâmetros importantes das RNA incluem a quantidade de camadas, o número de neurônios em cada camada, a função de ativação, o otimizador e o tamanho do lote. A quantidade de neurônios e o número de camadas definem as diferentes arquiteturas da RNA. A função de ativação é aplicada aos neurônios da rede para introduzir não-linearidades no modelo. As funções de ativação mais comuns são: '*tanh*' (tangente hiperbólica), '*sigmoid*' (função sigmoide), '*linear*' (função de identidade) e '*relu*' (*Rectified Linear Unit*) (GÉRON, 2019).

O otimizador é o algoritmo utilizado para atualizar os pesos da rede neural durante o treinamento, minimizado a função de custo. Alguns dos otimizadores mais comuns são: '*SGD*' (*Stochastic Gradient Descent*), '*RMSProp*' (*Root Mean Square Propagation*), '*Adagrad*' (*Adaptive Gradient*) e '*Adam*' (*Adaptive Moment Estimation*). Já o tamanho do lote (*batch size*) especifica a quantidade de amostras de treinamento empregadas para atualizar os pesos em cada iteração do algoritmo de otimização (GÉRON, 2019).

3.5 Máquinas de Vetores de Suporte

As técnicas de SVM são métodos de aprendizado de máquina amplamente aplicados em tarefas de classificação e regressão. Esses métodos são fundamentados em hiperplanos que separam as diferentes classes de decisão nos espaços de características (VAPNIK, 1998).

Em situações em que os conjuntos de dados são linearmente separáveis no espaço de característica, o SVM constrói um hiperplano que idealmente divide ambas as classes em duas regiões distintas. Quando os dados são não-lineares, o SVM recorre a funções do tipo kernel, responsáveis por mapear os dados em uma dimensão superior (MOHAMMADI *et al*., 2021). Adicionalmente, os algoritmos de SVM podem ser aplicados tanto em tarefas de classificação binárias quanto em multiclasses (GÉRON, 2019).

Na construção da SVM, tanto para problemas de regressão quanto para problemas de classificação, são definidos os parâmetros 'kernel', 'C' e 'gamma'. O kernel é uma função que permite transformar os dados em um espaço de características de maior dimensão, facilitando a separação entre as classes. Existem diferentes tipos de função kernel: linear, polinomial, sigmóide e *radial basis function* (*RBF*). A escolha do kernel adequado depende da natureza dos dados e do problema a ser resolvido (GÉRON, 2019).

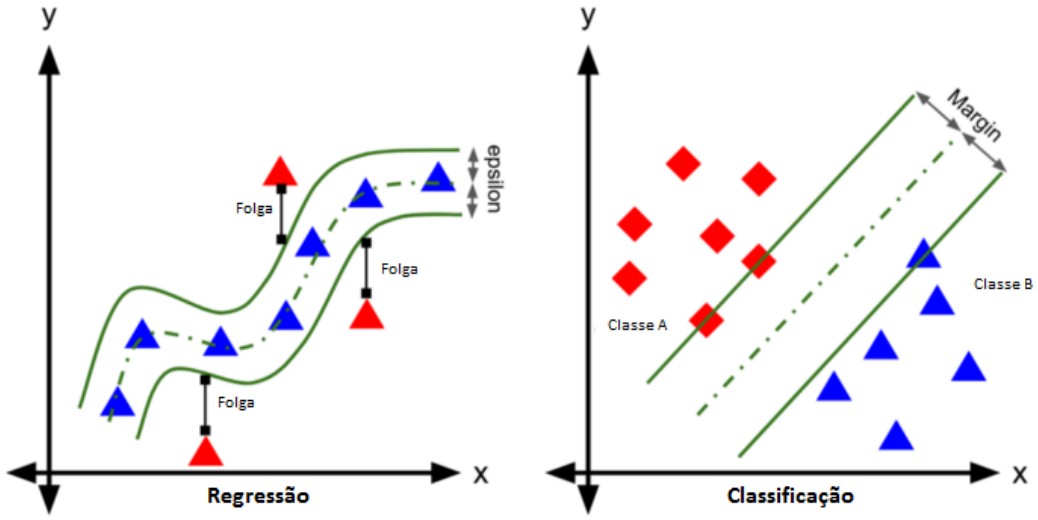
O parâmetro C controla a compensação entre a maximização da margem entre as classes e a minimização do erro de classificação. Um valor pequeno de C resulta em uma margem mais ampla, o que pode levar a uma melhor generalização. No entanto, um valor muito pequeno pode resultar em uma margem muito ampla, permitindo muitos erros de classificação. Um valor de C grande resulta em uma margem mais estreita, forçando o modelo a se ajustar aos dados de treinamento, o que pode levar ao *overfitting* (GÉRON, 2019).

Por fim, o parâmetro gamma é usado especificamente no kernel *RBF* e determina a forma da função de base radial. Um valor pequeno de gamma resulta em uma função *RBF* mais flexível, permitindo que a fronteira de decisão se adapte melhor aos dados. No entanto, um valor muito pequeno de gamma pode levar ao *overfitting*. Por outro lado, um valor grande de

gamma resulta em uma função *RBF* mais rígida, o que pode levar a um modelo menos flexível e com menor capacidade de adaptação aos dados (GÉRON, 2019).

Uma representação do funcionamento da técnica de SVM, tanto em tarefas de classificação quanto em tarefas de regressão, é mostrada na Figura 2.

Figura 2 – SVM para regressão e classificação.



Fonte: Adaptado de ACHSAN (2019).

Conforme Smola *et al*. (2000), as SVM são técnicas bastante atrativas devido à sua notável capacidade de generalização, obtendo resultados satisfatórios em tarefas de classificação e regressão para conjuntos de dados que não fazem parte do conjunto utilizado no treinamento. Além disso, apresentam robustez em situações com alta dimensionalidade e possuem fundamentação teórica sólida nos campos da matemática e estatística.

No contexto de aprendizado de máquina supervisionado, o conceito de generalização se refere à habilidade de um modelo em adaptar-se e efetuar previsões precisas em dados não vistos durante o treinamento. Em outras palavras, um modelo com boa generalização é capaz de aplicar o conhecimento obtido durante o treinamento a novos exemplos, apresentando um bom desempenho (GÉRON, 2019).

3.6 Métricas de Desempenho

Métricas de desempenho são utilizadas para avaliar a qualidade dos modelos de aprendizado de máquina, permitindo compreender o desempenho deles na realização de

previsões (GÉRON, 2019). No contexto de problemas de regressão, algumas métricas incluem o Erro Médio Absoluto (MAE), o Erro Quadrático Médio (MSE) e o Coeficiente de Determinação.

O MAE é a média dos erros absolutos entre as previsões do modelo e os valores reais. Essa métrica é fácil de interpretar, pois fornece uma medida do erro com a unidade da variável dependente. Quanto menor o MAE, melhor o desempenho do modelo, com o valor zero indicando previsões perfeitas (GÉRON, 2019).

O MSE, por sua vez, é a média dos erros ao quadrado entre as previsões do modelo e os valores reais. O MSE penaliza erros maiores, o que o torna mais sensível a outliers do que o MAE. Um valor de zero indica previsões perfeitas, e, assim como no MAE, valores menores são desejáveis (GÉRON, 2019).

O Coeficiente de Determinação é uma métrica que avalia quão bem o modelo de regressão se ajusta aos dados. O seu valor varia de 0 a 1, sendo 1 um ajuste perfeito e 0 indicando que o modelo não tem capacidade preditiva. Um valor maior de R^2 sugere que o modelo explica uma maior proporção da variabilidade dos dados e, portanto, tem um melhor desempenho (BRUCE; BRUCE, 2019).

Em problemas de classificação utilizando máquinas de aprendizado, as principais métricas incluem acurácia, precisão, recall, F1 score e taxa de erro (HARRISON, 2019).

A acurácia mede a proporção de previsões corretas em relação ao total de previsões. É uma métrica comum para avaliar o desempenho geral de um modelo, porém pode ser enganosa em situações com classes desbalanceadas (GÉRON, 2019).

Precisão é a proporção de verdadeiros positivos (TP) em relação à soma dos verdadeiros positivos e falsos positivos (FP). Em outras palavras, a precisão indica a fração de previsões corretas da classe positiva em relação a todas as previsões feitas para essa classe (GÉRON, 2019).

Recall, também conhecido como sensibilidade, é a proporção de verdadeiros positivos (TP) em relação à soma dos verdadeiros positivos e falsos negativos (FN). O recall indica a fração de exemplos positivos corretamente identificados pelo modelo em relação a todos os exemplos positivo (HARRISON, 2019).

F1 score é uma métrica que combina precisão e recall em um único valor, dando igual importância a ambas. O F1 score é calculado como a média harmônica entre precisão e recall, e é uma medida útil quando se deseja equilibrar os dois aspectos, especialmente em casos de classes desbalanceadas (HARRISON, 2019).

Por fim, a taxa de erro é a proporção de previsões incorretas em relação ao total de previsões. Pode ser interpretada como a proporção de erros cometidos pelo modelo em relação a todas as previsões (GÉRON, 2019).

3.7 Reação de Van de Vusse

Segundo Fogler (2009), reações complexas são uma classe de reações químicas que englobam uma combinação de reações paralelas e em série. Nas reações paralelas ou competitivas, um reagente é consumido por diferentes vias reacionais, gerando produtos distintos. As reações em série, por sua vez, formam um produto intermediário que posteriormente participa de outra reação, originando um novo produto.

A reação de Van de Vusse é uma reação complexa que foi introduzida por Van de Vusse (1964) em seu trabalho, que tinha como objetivo comparar as características e eficiências dos reatores do tipo PFR (*Plug Flow Reactor*) e CSTR (*Continuous Stirred Tank Reactor*) em processos químicos operando com reações complexas. Essa reação envolve um sistema reacional altamente não-linear, com reações em série e em paralelo ocorrendo simultaneamente (VAN DE VUSSE, 1964). A reação de Van de Vusse é apresentada nas Equações (2) e (3).

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C \tag{2}$$

$$2A \xrightarrow{k_3} D \tag{3}$$

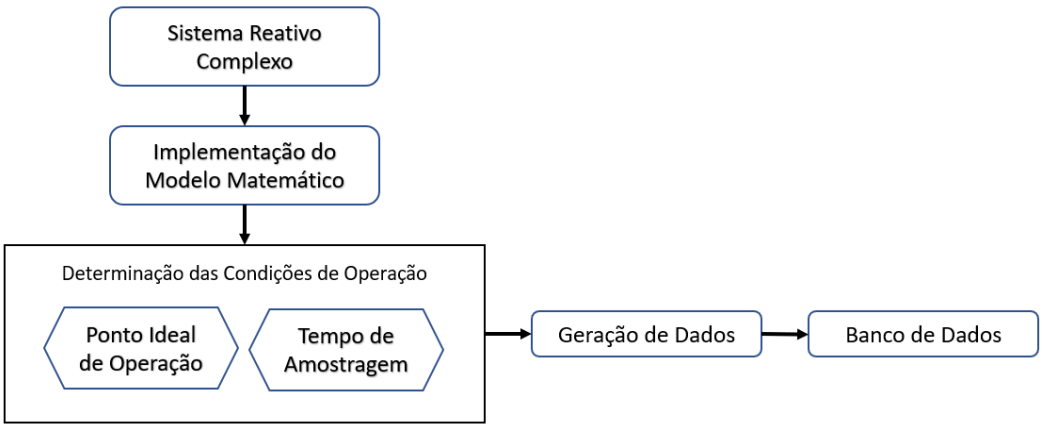
No trabalho de Vojtěšek (2007) sobre técnicas de controle adaptativo em reatores CSTR e PFR para reações de Van de Vusse, foi desenvolvida a modelagem matemática do processo reacional em um reator CSTR não-isotérmico. Esse modelo matemático foi utilizado no presente estudo para obtenção de dados estáticos e dinâmicos do processo, necessários à implementação dos sistemas de sensores virtuais e detecção e diagnóstico de falhas.

4 METODOLOGIA

A metodologia adotada foi organizada em três etapas principais: (1) implementação do modelo matemático e geração de dados; (2) desenvolvimento dos sensores virtuais; e (3) elaboração e implementação de estratégias de detecção e diagnóstico de falhas.

Na primeira etapa (Figura 3), o modelo matemático do reator CSTR, operando com a reação química de Van de Vusse, foi implementado e rotinas computacionais foram desenvolvidas para simular o comportamento do reator. Diferentes condições operacionais das variáveis Q_c e q_r foram simuladas para identificar os parâmetros de trabalho que maximizam a produção do composto B. Além disso, os tempos de amostragem adequados para monitoramento e controle do reator também foram determinados.

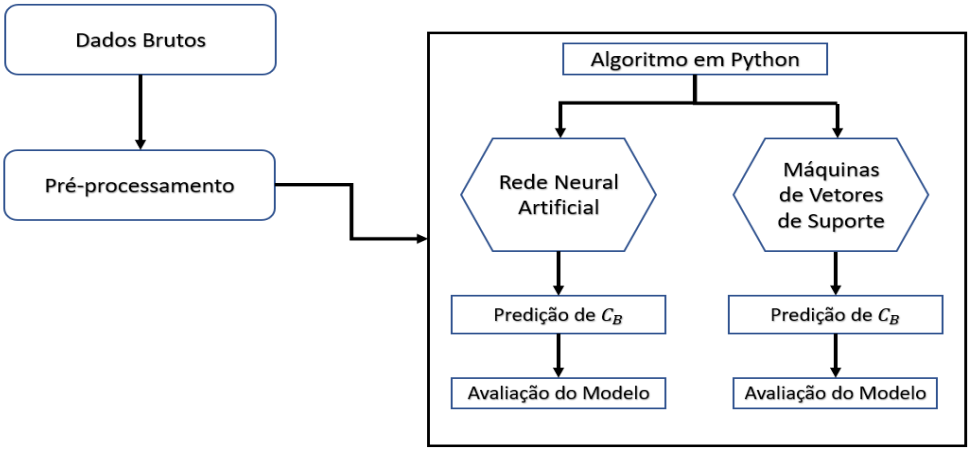
Figura 3 – Fluxograma da primeira parte do trabalho.



Fonte: Autor (2023).

A segunda etapa (Figura 4) envolveu a extração dos dados do processo em banco de dados relacional, pré-processamento, treinamento e validação dos modelos de sensores virtuais, utilizando as técnicas de RNA e SVM. Essa fase permitiu a criação de modelos capazes de realizar predições sobre a concentração de B.

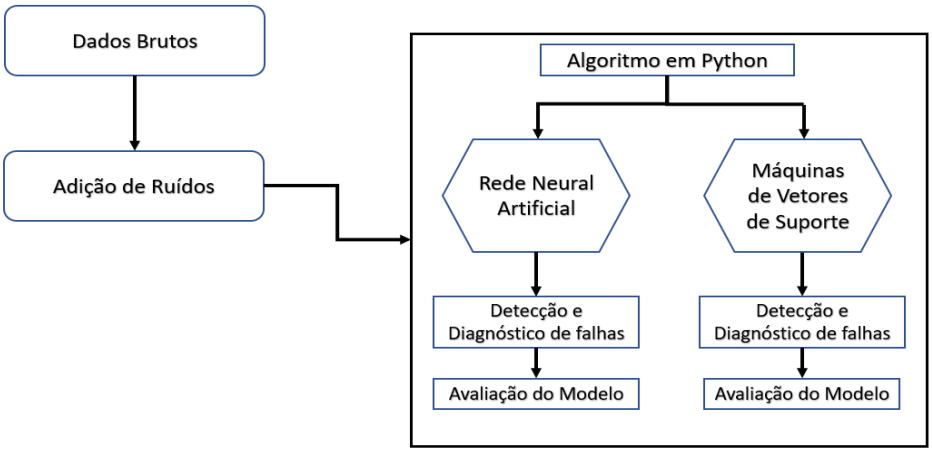
Figura 4 – Fluxograma da segunda parte do trabalho.



Fonte: Autor (2023).

Na terceira etapa (Figura 5), as técnicas de RNA e SVM foram aplicadas na detecção e diagnóstico de falhas, com base nos dados gerados e pré-processados por meio de rotinas computacionais em Python.

Figura 5 – Fluxograma da terceira parte do trabalho.



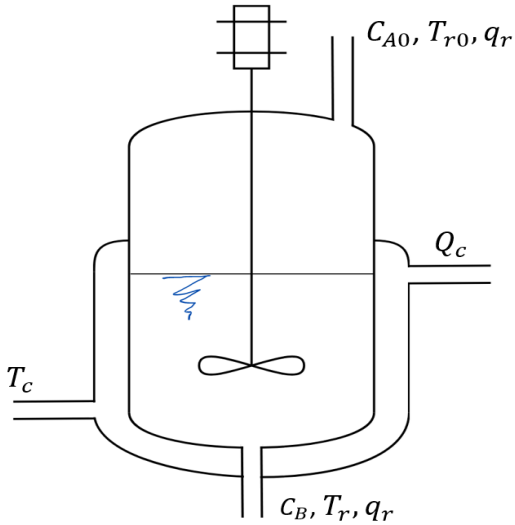
Fonte: Autor (2023).

4.1 Sistema Reativo Complexo

O sistema reativo complexo em análise (Figura 6) opera em regime transiente e é constituído por uma corrente de alimentação com vazão volumétrica q_r; concentração de reagente C_{A0} e temperatura T_{r0}; e produz o composto B a partir da reação de Van de Vusse. A corrente de saída do reator apresenta temperatura T_r e vazão volumétrica q_r. O controle de temperatura do reator é realizado por meio de uma jaqueta térmica.

Um trocador de calor externo é utilizado para controlar a vazão volumétrica e a temperatura da corrente de alimentação da jaqueta térmica, através do controle do calor removido do fluido refrigerante. A variável Q_c representa a taxa de troca térmica entre o fluido refrigerante do trocador de calor e o fluido refrigerante que alimenta o sistema do reator. A vazão volumétrica e a temperatura da corrente de alimentação do fluido refrigerante da jaqueta térmica são controladas pela variável Q_c. A corrente de saída da jaqueta térmica possui temperatura T_c.

Figura 6 – Esquema do reator CSTR.



Fonte: Adaptado de Vojtěšek (2007).

A Tabela 1 apresenta todas as variáveis associadas ao sistema reacional e as unidades de medida correspondentes.

Tabela 1 - Variáveis do sistema reacional.

Variável	Identificação	Unidade de Medida
C_{A0}	Concentração de reagente A na alimentação	kmol/m³
q_r	Vazão volumétrica da corrente de alimentação	m³/min
T_{r0}	Temperatura da corrente de alimentação	K
Q_c	Taxa de troca térmica no trocador de calor	kJ/min
T_c	Temperatura da corrente de saída da jaqueta térmica	K
T_r	Temperatura do meio reacional	K
C_A	Concentração de A no meio reacional	kmol/m³
C_B	Concentração de B no meio reacional	kmol/m³

Fonte: Vojtěšek (2007).

4.1.1 Construção do Modelo Matemático

A modelagem matemática do processo químico foi construída com base nos parâmetros do reator CSTR e nos parâmetros da reação de Van de Vusse, obtidos a partir do trabalho de Vojtěšek (2007). Estes parâmetros estão presentes na Tabela 2.

Tabela 2 – Parâmetros do reator CSTR e da reação de Van de Vusse.

Nome do parâmetro	Símbolo e valor do parâmetro
Volume do reator	V_r = 0,01 m³
Densidade do reagente	ρ_r = 934,2 kg m^{-3}
Capacidade calorífica do reagente	c_{pr} = 3,01 kJ kg^{-1} K^{-1}
Massa do fluido refrigerante	m_c = 5 kg
Capacidade calorífica do fluido refrigerante	c_{pc} = 2,0 kJ kg^{-1} K^{-1}
Superfície de troca térmica	A_r = 0,215 m²
Coeficiente de transferência de calor	U = 67,2 kJ min^{-1} m^{-2} K^{-1}
Fator pré-exponencial para a reação 1	k_{01} = 2,145x10^{10} min^{-1}
Fator pré-exponencial para a reação 2	k_{02} = 2,145x10^{10} min^{-1}
Fator pré-exponencial para a reação 3	k_{03} = 1,5072x10^{8} min^{-1} kmol^{-1}
Razão entre a energia de ativação e R para a reação 1	E_1/R = 9758,3 K
Razão entre a energia de ativação e R para a reação 2	E_2/R = 9758,3 K
Razão entre a energia de ativação e R para a reação 3	E_3/R = 8560 K
Entalpia da reação 1	h_1 = -4200 kJ kmol^{-1}
Entalpia da reação 2	h_2 = 11000 kJ kmol^{-1}
Entalpia da reação 3	h_3 = 41850 kJ kmol^{-1}

Fonte: Vojtěšek (2007).

A reação de Van de Vusse ocorre no interior do reator CSTR. As equações de balanço de massa e energia que descrevem o comportamento do reator, construído e validado com os parâmetros fornecidos por Vojtěšek (2007), são apresentados a seguir.

$$\frac{dC_A}{dt} = \frac{q_r}{V_r}(C_{A0} - C_A) - k_1 C_A - K_3 C_A^2 \quad (4)$$

$$\frac{dC_B}{dt} = -\frac{q_r}{V_r} C_B + k_1 C_A - k_2 C_B \quad (5)$$

$$\frac{dT_r}{dt} = \frac{q_r}{V_r}(T_{r0} - T_r) - \frac{h_r}{\rho_r c_{pr}} + \frac{A_r U}{V_r \rho_r c_{pr}}(T_c - T_r) \quad (6)$$

$$\frac{dT_c}{dt} = \frac{1}{m_c c_{pc}}[Q_c + A_r U(T_r - T_c)] \quad (7)$$

As taxas de reação são descritas pela Lei de Arrhenius, conforme Equação (8), enquanto as entalpias de reação são calculadas conforme a Equação (9).

$$k_j(T_r) = k_{0j} \cdot exp\left(\frac{-E_j}{RT_r}\right), para\ j = 1{,}2{,}3 \quad (8)$$

$$h_r = h_1 k_1 c_A + h_2 k_2 c_B + h_3 k_3 c_A^2 \quad (9)$$

Para resolver as equações do sistema, foi empregada a implementação computacional do método de Runge-Kutta em linguagem Python. Este método numérico permite obter soluções aproximadas para as equações diferenciais que descrevem o comportamento do reator ao longo do tempo.

4.1.2 Implementação em Python

As rotinas computacionais responsáveis pela geração, extração e transformação de dados, bem como a implementação das máquinas de aprendizado aplicadas aos sensores virtuais e à detecção e diagnóstico de falhas, foram desenvolvidas utilizando a linguagem de programação Python. O ambiente de desenvolvimento integrado (IDE) *Spyder* foi empregado para facilitar a criação e a manutenção dessas rotinas.

Para resolver o conjunto de equações diferenciais do modelo matemático do sistema, optou-se pelo emprego do método numérico de Runge-Kutta de segunda ordem.

A análise e a visualização dos dados gerados foram conduzidas com o auxílio das bibliotecas *Pandas*, *Matplotlib* e *Seaborn*. A primeira facilitou a manipulação e a análise dos dados, enquanto as últimas permitiram a criação de gráficos.

No que diz respeito ao armazenamento e à extração dos dados obtidos nas simulações computacionais, a biblioteca *psycopg2* foi utilizada para integrar a linguagem Python ao sistema gerenciador de banco de dados PostgreSQL, permitindo a conexão do algoritmo Python com o banco de dados relacional responsável pelo armazenamento dos dados. O banco de dados relacional foi utilizado para facilitar o armazenamento dos dados durante a simulação computacional, uma vez que cada dado gerado era prontamente armazenado no banco de dados e deletado da memória RAM do computador.

As tarefas de extração, preparação dos dados, pré-processamento, treinamento e validação das máquinas de aprendizado supervisionado, na etapa de desenvolvimento dos sensores virtuais e detectores de falhas, foram realizadas empregando-se rotinas computacionais em Python. Nessa etapa, foram utilizadas as bibliotecas *Scikit-Learn* e *TensorFlow* para implementação e avaliação das métricas de desempenho dos modelos de RNA e SVM.

O Quadro 1 apresenta as ferramentas e as versões utilizadas na etapa de implementação das rotinas computacionais.

Quadro 1 – Ferramentas computacionais utilizadas.

Linguagem de Programação	Python 3.10
Bibliotecas Python	*Numpy 1.23.5*
	Matplotlib 3.7.0
	Seaborn 0.12.2
	Pandas 1.5.3
	Scikit-learn 1.2.1
	Tensorflow 2.12.0
	Psycopg 2.9.5
Ambiente de Desenvolvimento	*Spyder 5.4.1*

Fonte: Autor (2023).

4.1.3 Determinação da Condição Ideal de Operação

O estado estacionário para o sistema é alcançado quando os valores das variáveis de entrada (C_{A0}, T_{r0}, q_r, Q_c) permanecem constantes e o tempo tende ao infinito. Esse estado é

atingido quando as variações das variáveis C_A, C_B, T_r e T_c ao longo do tempo são nulas (VOJTĚŠEK, 2007).

De acordo com Vojtěšek (2007), o estado estacionário do sistema reativo investigado neste trabalho é definido apenas pelos valores das variáveis q_r, C_{A0}, T_{r0} e Q_c. As condições iniciais da concentração do reagente A, concentração do produto B, temperatura inicial do reator e temperatura inicial do fluido refrigerante não influenciam os valores das variáveis do estado estacionário final do sistema. Portanto, para analisar os estados estacionários alcançados pelo sistema em diferentes condições operacionais e definir o ponto ótimo de operação do reator, foram realizadas simulações considerando diversos cenários de Q_c e q_r. A escolha de apenas duas variáveis se deve ao fato de que os valores de C_{A0} e T_{r0} já são pré-determinados. Os cenários analisados são apresentados na Tabela 3.

Tabela 3 – Cenários de análise estacionária.

Cenários de Simulação de Estado Estacionário	
Q_c	$\langle -500; 500 \rangle\ kJ.min^{-1}$
q_r	$\langle 0{,}001; 0{,}030 \rangle\ m^3.min^{-1}$
c_{A0}	5,1 kmol.m^{-3}
T_{r0}	387,05 K

Fonte: Autor (2023).

Com o auxílio do *software OriginPro* foram construídas as superfícies de C_B, C_A, T_r e T_c em função de q_r e Q_c. O objetivo das superfícies é possibilitar a visualização do comportamento do reator para várias condições operacionais e auxiliar na determinação das regiões ótimas de operação.

O ponto ótimo de operação será selecionado com base na concentração do produto B, C_B. Esta condição do sistema é caracterizada pela maximização da produção do composto B. A partir desse ponto, serão desenvolvidos algoritmos de sensores virtuais e detecção e diagnóstico de falhas, como estratégia para o controle e monitoramento do processo reacional. Esta abordagem visa melhorar a eficiência do sistema, garantindo um desempenho otimizado.

4.2 Geração de Dados

A primeira etapa para a geração de dados envolve a determinação do tempo de amostragem para coleta de dados do processo. O tempo de amostragem refere-se ao intervalo entre as medições consecutivas realizadas por um sensor. Em geral, o tempo de amostragem mais curto permite uma resposta mais rápida a variações no processo, sendo benéfico para o controle em tempo real. Por outro lado, leva a um maior consumo de recursos computacionais. Tempos de amostragem longos, por sua vez, resultam em informações insuficientes para o controle do processo, especialmente em sistemas com dinâmicas rápidas (ÅSTRÖM; HÄGGLUND, 2006). Para determinação dos tempos de amostragem, foram aplicadas perturbações de -10% e +10% nas variáveis C_{A0}, Q_c e q_r, e -2,58% e +2,58% na variável T_{r0}, individualmente, em torno do ponto ideal de operação, a fim de avaliar a influência de cada variável no comportamento do reator após sofrerem perturbações. As porcentagens de perturbação, e os limites inferiores e superiores associados, foram selecionados de forma conveniente, de acordo com o julgado como razoável.

Para cada variável perturbada, foi registrado o tempo que o sistema leva para atingir um novo estado estacionário após cada perturbação e o comportamento de C_B após as perturbações. O objetivo desta análise é determinar o melhor tempo de amostragem, de modo a garantir que não haja perdas de informações sobre a dinâmica do processo e a representatividade dos dados.

Os limites superiores e inferiores das perturbações para cada variável são apresentados no Quadro 2.

Quadro 2 – Limites das variáveis de entrada e variáveis de perturbação.

Variáveis Distúrbio		
Limites	$C_{A0}: (4{,}59: 5{,}61)$	$T_{r0}: (377{,}05: 397{,}05)$
Unidade	$kmol/m^3$	K
Variáveis Manipuladas		
Limites	$Q_c: (130{,}05: 158{,}95)$	$q_r: (0{,}00288: 0{,}00352)$
Unidade	kJ/min	m^3/min

Fonte: Autor (2023).

4.2.1 Dados para o Sensor Virtual

Após a simulação computacional, foram gerados quatro conjuntos de dados, cada um correspondendo a um tempo de amostragem diferente. Durante a etapa de geração de dados, a operação dinâmica do processo é simulada, com perturbações randômicas inseridas nas variáveis C_{A0}, T_{r0}, Q_c e q_r ao longo do tempo. As perturbações aplicadas são do tipo degrau e foram geradas de forma aleatória dentro do intervalo especificado no Quadro 2 em torno do valor do ponto ótimo de operação de cada variável. Ao longo da simulação os dados gerados a cada tempo de amostragem foram armazenados em tempo real em um banco de dados relacional, contribuindo para o aumento do desempenho computacional.

Os quatro conjuntos de dados provêm da mesma simulação computacional, e a única diferença entre eles é o tempo de coleta das amostras. Essa abordagem permite comparar a eficiência dos sensores virtuais para diferentes tempos de amostragem, possibilitando a escolha do tempo mais adequado para o desenvolvimento de sensores virtuais que proporcionem uma melhor performance no monitoramento do processo químico em estudo.

Para validar e verificar o desempenho dos modelos de sensores virtuais de RNA, foi empregada uma segunda base de dados, em vez de usar o mesmo conjunto de dados nas etapas de treinamento e teste, pois os dados são séries temporais do processo. Para isso, realizou-se uma segunda simulação dinâmica do reator a fim de gerar um novo conjunto de dados para validação do modelo de aprendizado de máquina, que não está relacionado ao utilizado no treinamento dos modelos. Tal abordagem foi adotada com base na experiência obtida no trabalho, por tentativa e erro.

Os dados de validação são empregados para monitorar o desempenho durante a fase de treinamento, auxiliando na prevenção de sobreajuste (*overfitting*) e assegurando a adequada generalização para dados não vistos pelo algoritmo na etapa de treinamento. A validação dos modelos é uma tarefa crucial para verificar o desempenho do modelo em novas instâncias de dados, que é o objetivo principal (GÉRON, 2019).

4.2.2 Dados para Detecção e Diagnóstico de Falhas

Os dados foram gerados por meio de rotinas computacionais implementadas em Python e categorizados em diferentes classes, sendo os dados de operação normal rotulados como “normal”.

As amostras foram geradas a partir da base de dados com o tempo de amostragem de melhor desempenho na etapa de desenvolvimento do sensor virtual. As falhas impostas simulam erros no funcionamento dos sensores físicos. A Tabela 4 apresenta as falhas impostas aos sensores e as classes de diagnóstico de falhas correspondentes.

Tabela 4 – Falhas de sensores do processo.

Variável Medida	Falha de Sensor	Diagnóstico
-	-	Normal
C_{A0}	[-90%; 90%]	Falha 1
T_{r0}	[-90%; 90%]	Falha 2
Q_c	[-90%; 90%]	Falha 3
q_r	[-90%; 90%]	Falha 4
T_r	[-90%; 90%]	Falha 5
T_c	[-90%; 90%]	Falha 6

Fonte: Autor (2023).

Para gerar a base de dados com falhas para o treinamento e validação dos algoritmos de detecção e diagnóstico de falhas, foi desenvolvido um código em Python. O código iterou sobre todas as linhas da base de dados original e, para cada linha, decidiu aleatoriamente se seria inserida a falha ou não. Nos casos em que o código decidiu pela não inserção, a linha recebeu o rótulo “normal”. Nos casos em que o código decidiu pela inserção de falhas, o código selecionou de forma randômica a variável a receber a falha.

A falha inserida consiste na multiplicação do valor real da variável por um número randômico, que está no intervalo de -0,9 a 0,9, e adição do resultado ao valor real da variável, simulando um aumento ou diminuição de até 90% (SILVA, 2022). Dessa forma, o código gera um conjunto de dados com falhas induzidas aleatoriamente em diferentes variáveis do processo. Esses dados são então utilizados para treinar e validar os algoritmos de detecção e diagnóstico de falhas, permitindo que os modelos aprendam a identificar e classificar diferentes tipos de falhas a partir das características dos dados.

A abordagem da seleção aleatória da inserção ou não de falhas e seleção aleatória do tipo de falha inserida possibilitou a construção de uma base de dados balanceada. Nessa condição, cada classe contém quantidade de dados similares, proporcionando maior precisão na tarefa de detecção e diagnóstico e evitando o problema de desbalanceamento de classes na etapa de treinamento.

4.3 Construção do Sensor Virtual

Os conjuntos de dados de treinamento e validação, obtidos na etapa de geração de dados e estruturados como séries históricas, foram pré-processados e preparados para serem utilizados no desenvolvimento do sensor virtual. Conforme Schnitman (1998), a utilização de informações históricas das variáveis de entrada e saída do processo aprimora a capacidade de aprendizado das máquinas e permite a captura dinâmica do processo. Assim, os conjuntos de dados foram reorganizados utilizando a biblioteca Pandas do Python, a fim de construir entradas com atrasos temporais, de modo que as entradas dos modelos considerem os valores atuais e anteriores das variáveis para a tarefa de predição da concentração do composto B.

Desta forma, foi realizado o estudo com 1, 2 e 3 atrasos temporais, com o objetivo de avaliar o desempenho preditivo do sensor virtual utilizando dados passados do processo (SARAIVA, 2017). Os Quadros 3, 4 e 5 resumem a organização das variáveis de entrada e saída empregados nos modelos de sensores virtuais, com os atrasos nas variáveis representados por $(k-3)$, $(k-2)$, e $(k-1)$.

Quadro 3 – Organização do conjunto de dados com 1 tempo de atraso.

Conjunto	Entrada – 1 Atraso	Saída
3 segundos e 1 tempo de atraso	$C_B(k-1)$ $T_r(k-1)$ $T_C(k-1)$ $T_{r0}(k-1)$ $C_{A0}(k-1)$ $Q_c(k-1)$ $q_r(k-1)$	$C_B(k)$
6 segundos e 1 tempo de atraso		
9 segundos e 1 tempo de atraso		
12 segundos e 1 tempo de atraso		

Fonte: Autor (2023).

Quadro 4 – Organização do conjunto de dados com 2 tempos de atraso.

Conjunto	Entrada – 2 Atrasos		Saída
3 segundos e 2 tempos de atraso	$C_B(k-1)$ $T_r(k-1)$ $T_C(k-1)$ $T_{r0}(k-1)$ $C_{A0}(k-1)$ $Q_c(k-1)$ $q_r(k-1)$	$C_B(k-2)$ $T_r(k-2)$ $T_C(k-2)$ $T_{r0}(k-2)$ $C_{A0}(k-2)$ $Q_c(k-2)$ $q_r(k-2)$	$C_B(k)$
6 segundos e 2 tempos de atraso			
9 segundos e 2 tempos de atraso			
12 segundos e 2 tempos de atraso			

Fonte: Autor (2023).

Quadro 5 – Organização do conjunto de dados com 3 tempos de atraso.

Conjunto	Entrada – 3 Atrasos			Saída
3 segundos e 3 tempos de atraso	$C_B(k-1)$	$C_B(k-2)$	$C_B(k-3)$	$C_B(k)$
6 segundos e 3 tempos de atraso	$T_r(k-1)$	$T_r(k-2)$	$T_r(k-3)$	
9 segundos e 3 tempos de atraso	$T_C(k-1)$	$T_C(k-2)$	$T_C(k-3)$	
12 segundos e 3 tempos de atraso	$T_{r0}(k-1)$	$T_{r0}(k-2)$	$T_{r0}(k-3)$	
	$C_{A0}(k-1)$	$C_{A0}(k-2)$	$C_{A0}(k-3)$	
	$Q_c(k-1)$	$Q_c(k-2)$	$Q_c(k-3)$	
	$q_r(k-1)$	$q_r(k-2)$	$q_r(k-3)$	

Fonte: Autor (2023).

Após inserção dos atrasos temporais, os dados de entrada foram pré-processados para garantir que os atributos previsores (variáveis que são utilizadas para a predição de uma variável objetivo) estivessem na mesma escala. Para isso, aplicou-se o escalonamento min-max, também conhecido como normalização, técnica na qual os valores são deslocados e redimensionados para variarem entre 0 e 1. A função *MinMaxScaler* da biblioteca *Scikit-Learn* foi utilizada para normalizar os dados de entrada (GÉRON, 2019).

4.3.1 Rede Neural Artificial

Os conjuntos de dados de treinamento e validação foram carregados separadamente a partir de um banco de dados relacional, pré-processados e normalizados de maneira análoga. Posteriormente, os modelos de aprendizado de máquina foram treinados utilizando o conjunto de treinamento. Durante o treinamento, o desempenho foi monitorado com base nos dados de validação.

A RNA foi implementada utilizando a biblioteca *TensorFlow*. A RNA foi construída empregando diferentes hiperparâmetros para avaliação e seleção do conjunto mais adequado. A Tabela 5 apresenta os conjuntos de hiperparâmetros utilizados na construção dos modelos.

Tabela 5 – Hiperparâmetros utilizados na construção dos modelos de RNA.

Hiperparâmetro	Valores
Nº de neurônios nas camadas	[16,8,1], [8,4,1], [32,16,1], [64,32,1]
Função de ativação	*'tanh', 'sigmoid', 'linear' e 'relu'*
Otimizador	*'SGD', 'RMSprop', 'Adagrad' e 'adam'*
Tamanho do lote (*batch_size*)	16, 32, 64 e 128

Fonte: Autor (2023).

A fim de evitar o ajuste excessivo (*overfitting*), foi empregado o método de parada antecipada (*early stopping*), monitorando a perda de desempenho do modelo no conjunto de validação e interrompendo o treinamento após 10 épocas sem melhora. O modelo foi validado com o conjunto de dados de validação normalizados.

Após a etapa de treinamento, o modelo foi avaliado com base em suas previsões no conjunto de validação. Métricas como Erro Quadrático Médio (MSE), Coeficiente de Determinação (R^2) e Erro Médio Absoluto (MAE) foram calculadas para avaliar o desempenho do sensor virtual, proporcionando uma compreensão de quão bem ele generaliza para novos conjuntos de dados (GÉRON, 2019).

Por fim, foram gerados gráficos de dispersão e perfis temporais da dinâmica do processo para visualizar a relação entre os valores reais e as estimativas do modelo.

4.3.2 Máquina de Vetores de Suporte

O conjunto de dados de treinamento foi importado a partir do banco de dados relacional empregado no estudo. Os atributos previsores e os atributos objetivos, correspondente à variável CB(k), foram extraídos do conjunto de dados. Em seguida, os dados foram divididos em dois subconjuntos, treinamento e teste, utilizando a função '*train_test_split*' do *Scikit-Learn*. A proporção da divisão foi estabelecida em 80% para treinamento e 20% para teste conforme recomendado por Géron (2019).

O modelo SVM foi construído utilizando a classe 'SVR' do *Scikit-Learn*. Os hiperparâmetros do modelo, como o tipo de kernel, o parâmetro de regularização C e o parâmetro gamma, foram selecionados por meio da técnica de validação cruzada e pesquisa em grade (*grid search*). Na validação cruzada os dados de treinamento são divididos em diferentes subconjuntos (partições) e cada modelo diferente é treinado utilizado uma combinação desses subconjuntos e validado com os demais. Após a seleção dos melhores hiperparâmetros, o modelo final é treinado e o desempenho avaliado com os dados do conjunto de teste (GÉRON, 2019). A Tabela 6 apresenta os hiperparâmetros e os conjuntos de valores que foram avaliados para determinar os hiperparâmetros mais adequados ao problema.

Tabela 6 – Hiperparâmetros utilizados na construção dos modelos de SVM.

Hiperparâmetro	Valores
Kernel	‘linear’, ‘rbf’ e ‘poly’
Parâmetro C	0,5; 1; 10 e 100
Parâmetro gamma	‘scale’; ‘auto’; 0,1; 1 e 10

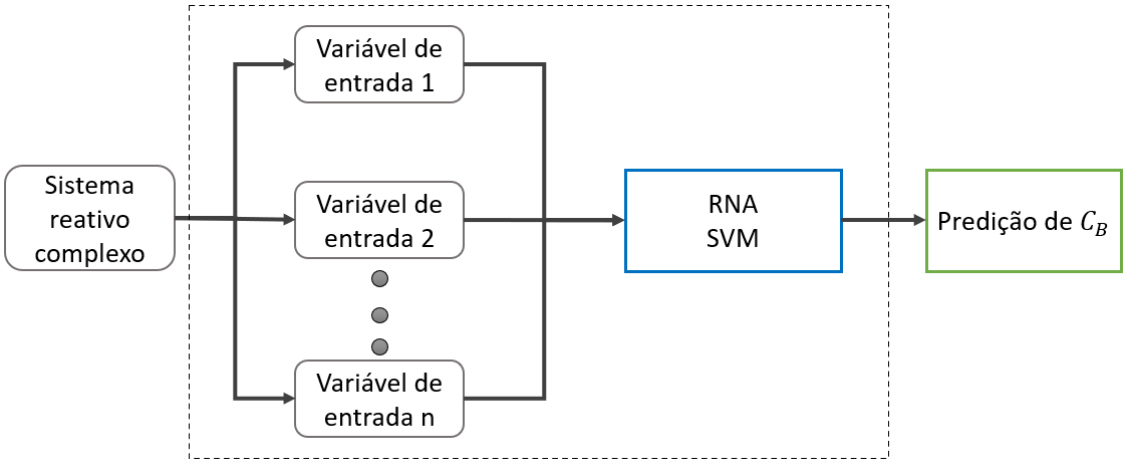
Fonte: Autor (2023).

O modelo SVM foi treinado utilizando validação cruzada com 5 partições, a fim de encontrar a melhor combinação de hiperparâmetros. O desempenho do modelo treinado foi verificado no conjunto de teste empregando métricas de desempenho como MSE, Coeficiente de Determinação (R^2) e MAE. As métricas foram calculadas pelo algoritmo utilizando as funções internas ‘*mean squared error*’, ‘*r2 score*’ e ‘*mean absolute error*’ da biblioteca *Scikit-Learn*.

Um gráfico de dispersão foi gerado para comparar visualmente as estimativas do modelo com os valores reais no conjunto de teste, permitindo a identificação de padrões e inconsistências no modelo de sensor virtual.

Uma representação simplificada do funcionamento do sensor virtual proposto no trabalho é ilustrada na Figura 7.

Figura 7 – Esquema simplificado do sensor virtual.



Fonte: Autor (2023).

4.4 Construção do Detector de Falhas

O conjunto de dados de treinamento e teste para o desenvolvimento do detector de falhas foi importado a partir do banco de dados relacional. Os dados foram divididos em atributos previsores e atributos objetivos, contendo as classes de diagnóstico. Em seguida, foram separados em conjunto de treinamento, com 70% dos dados, e teste, com 30%, utilizando a função '*train test split*'. A separação dos realizados é realizada para a verificação do desempenho do modelo em novas instâncias de dados (GÉRON, 2019).

Os dados de entrada foram padronizados utilizando a função '*StandardScaler*' do *Scikit-Learn*, para evitar que as diferenças de escala entre as variáveis prejudicassem o desempenho dos modelos (GÉRON, 2019).

4.4.1 Rede Neural Artificial

Para realizar a detecção e diagnóstico de falhas, foi empregado um classificador de RNA utilizando a classe '*MLP Classifier*' da biblioteca *Scikit-Learn*. A RNA foi treinada com o conjunto de dados de treinamento e, posteriormente, utilizada para realizar previsões no conjunto de teste.

A fim de avaliar o desempenho do modelo, foram calculadas diversas métricas, incluindo acurácia, precisão, *recall*, *F1 Score*, e taxa de erro. Essas métricas foram obtidas por meio de funções específicas do Scikit-Learn, como '*accuracy score*', '*precision score*' e '*recall score*'. A matriz de confusão foi gerada e exibida em forma de gráfico, utilizando a biblioteca *Seaborn*, permitindo uma análise visual das previsões do classificador em relação aos dados verdadeiros (GÉRON, 2019).

Adicionalmente, o desempenho nas tarefas de detecção e diagnóstico de falhas foi avaliado de forma individual. Esta análise buscou verificar o quão eficaz o modelo é na detecção das falhas e, ao detectá-las, o quão bem identifica o tipo específico de cada falha. Essa avaliação foi realizada a partir da matriz de confusão gerada.

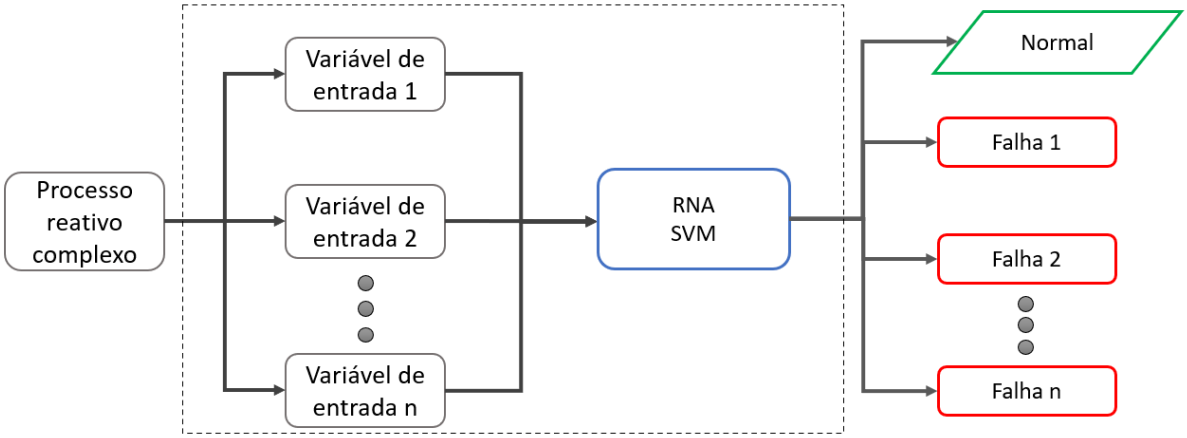
4.4.2 Máquina de Vetores de Suporte

Com os dados adequadamente preparados, o classificador SVM (implementado pela classe SVC do *Scikit-Learn*) foi treinado utilizando o conjunto de treinamento. A fim de avaliar o desempenho do modelo, foram realizadas previsões com o modelo treinado, empregando o conjunto de teste. Diversas métricas de desempenho foram calculadas, tais como a acurácia, *recall*, *F1 score* e taxa de erro. Além disso, foi gerada uma matriz de confusão para comparar os resultados verdadeiros com os previstos pelo modelo (GÉRON, 2019).

A partir dos resultados da matriz de confusão, o desempenho do modelo foi avaliado tanto na detecção quanto no diagnóstico das falhas, de maneira individual. Essa abordagem permitiu verificar a capacidade do modelo em detectar as falhas e, após a detecção, sua eficácia em identificar corretamente o tipo específico de falha.

O esquema simplificado da estratégia de detecção e diagnóstico de falhas implementada neste trabalho é apresentado na Figura 8, ilustrando o funcionamento do modelo desenvolvido.

Figura 8 – Esquema simplificado da estratégia de detecção e diagnóstico de falhas.



Fonte: Autor (2023).

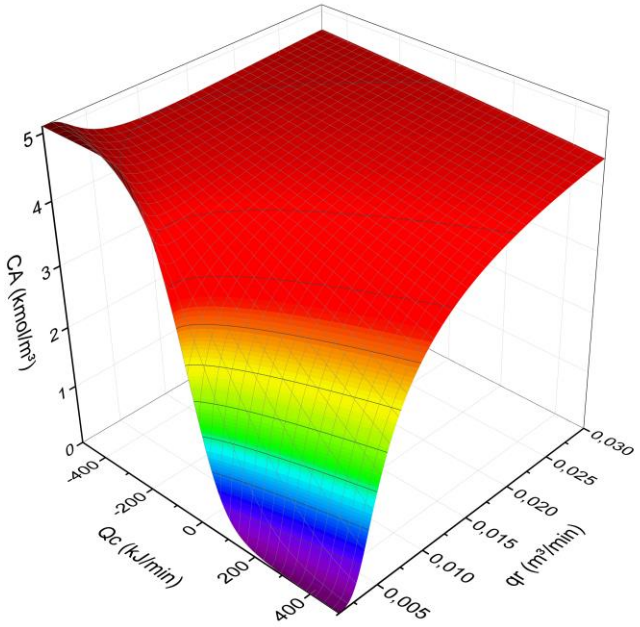
5 RESULTADOS E DISCUSSÃO

Tendo em vista que o objetivo do trabalho é a simulação computacional do sistema reativo complexo e o emprego de RNA e SVM no desenvolvimento de sensores virtuais e modelos de detecção e diagnóstico de falhas, os resultados foram separados em: determinação do ponto de operação, geração de dados, avaliação e discussão dos modelos de sensores virtuais e modelos de detecção e diagnóstico de falhas. Por fim, os desempenhos de ambas as técnicas de aprendizado de máquina foram discutidos e comparados.

5.1 Ponto de Operação

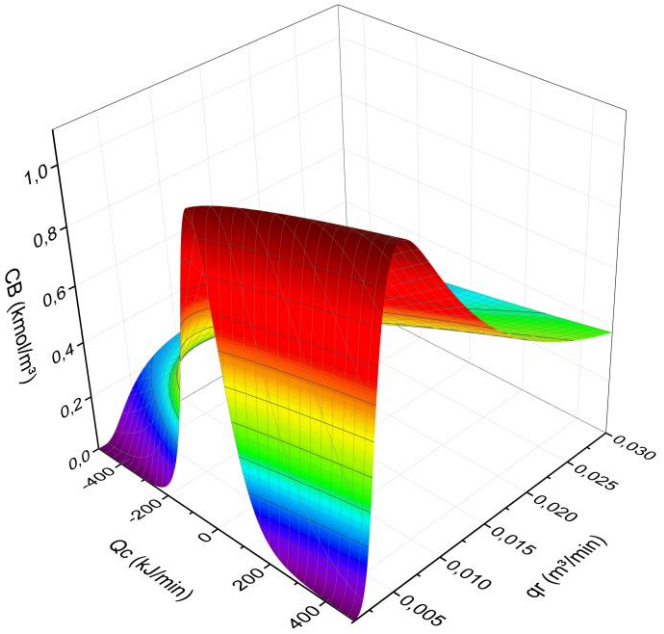
Com base nas simulações de estado estacionário, foram construídas superfícies de resposta para a concentração de A (C_A), concentração de B (C_B), temperatura do meio reacional (T_r) e a temperatura de saída do fluido refrigerante (T_c) usando o *software OriginPro*. As Figuras 9-12 apresentam as superfícies construídas para os cenários analisados. A análise permite visualizar o comportamento do reator em várias condições operacionais de Q_c e q_r, e identificar as regiões que maximizam a produção do composto B.

Figura 9 – Superfície da concentração de A.



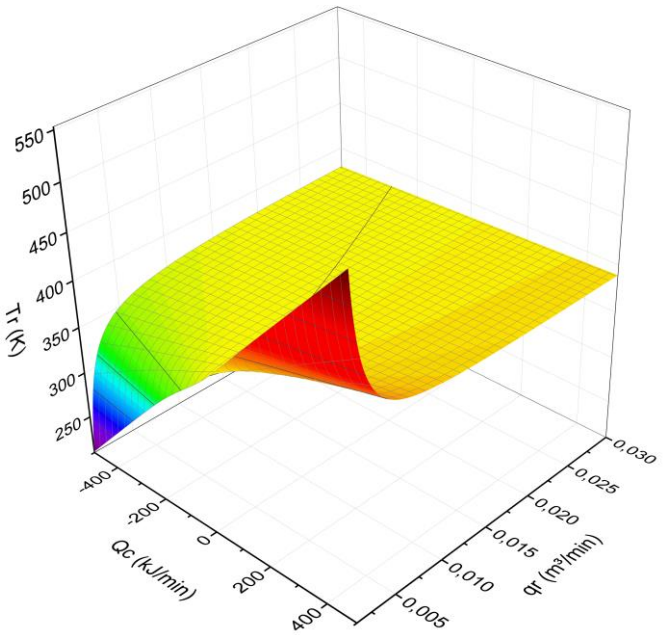
Fonte: Autor (2023).

Figura 10 – Superfície da concentração de B.



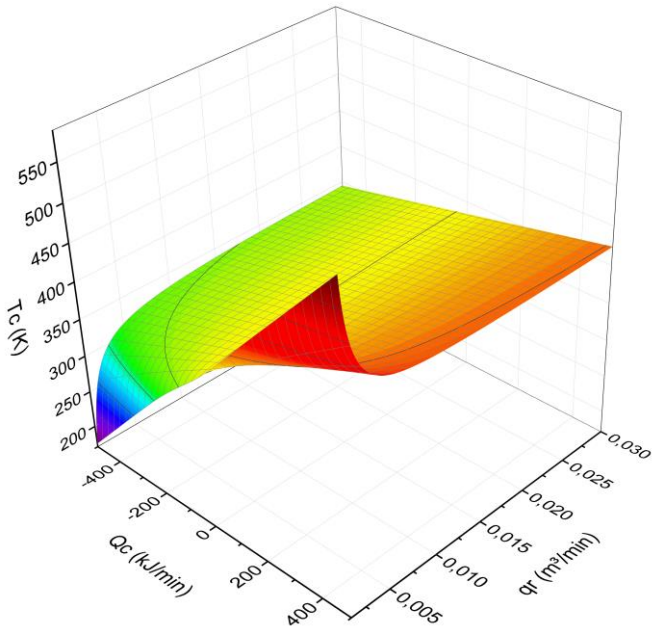
Fonte: Autor (2023).

Figura 11 – Superfície da temperatura do meio reacional.



Fonte: Autor (2023).

Figura 12 – Superfície da temperatura do fluido refrigerante.



Fonte: Autor (2023).

De acordo com os dados obtidos nas simulações de estado estacionário, o melhor ponto de operação do reator é atingido nas seguintes condições operacionais: taxa de troca térmica de 144,5 kJ/min, vazão volumétrica de alimentação da corrente de reagentes de 0,0032 m³/min, concentração de reagente na corrente de alimentação de 5,1 kmol/m³ e temperatura da corrente de alimentação de 387,05 K.

Dessa forma, os parâmetros operacionais são definidos (Tabela 7) para o estado estacionário que servirá de referência para o modelo de geração de dados. A identificação dessa condição ideal é essencial para implementação das estratégias de controle e monitoramento, contribuindo para a otimização do desempenho do reator e garantindo a produção eficiente do composto de interesse.

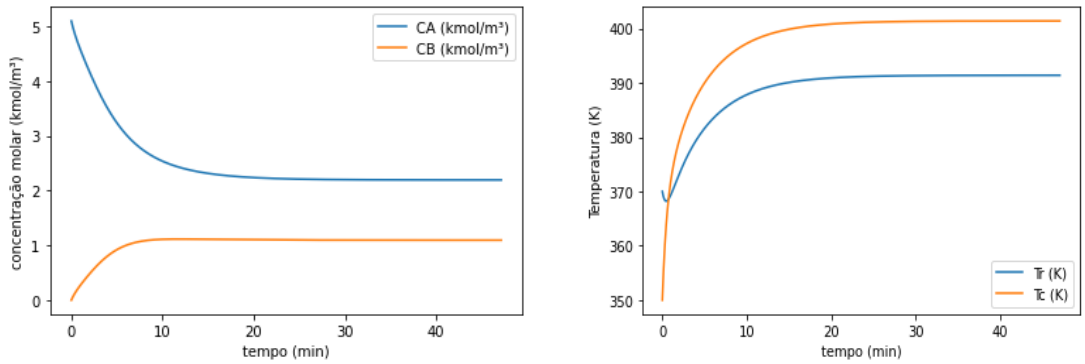
Tabela 7 – Condição ótima de operação do reator.

Variável	Valor de Estado Estacionário Ideal
Q_c	144,5 kJ/min
q_r	0,0032 m³/min
C_{A0}	5,1 kmol/m³
T_{r0}	387,05 K

Fonte: Autor (2023).

Após a determinação do ponto ótimo de operação, realizou-se uma simulação com o objetivo de identificar os valores das variáveis do sistema após alcançar o estado estacionário. A simulação considerou uma temperatura inicial de 370 K para o meio reacional, 350 K para a temperatura inicial do fluido na jaqueta térmica e uma concentração inicial de 5,1 kmol/m³ para o reagente A no meio reacional, simulando a partida do reator. A Figura 13 apresenta os perfis das concentrações e temperaturas do sistema até atingir o estado estacionário utilizado como referência para geração de dados de operação do reator.

Figura 13 – Perfil dinâmico do modelo com os parâmetros de trabalho.



Fonte: Autor (2023).

Os resultados obtidos durante as simulações demonstram que o estado estacionário final do sistema é determinado pelas variáveis Q_c, q_r, C_{A0} e T_{r0}, sendo independente das condições iniciais da temperatura do meio reacional, da condição inicial da temperatura do fluido da jaqueta térmica e das concentrações iniciais do reagente A e do produto B no meio. Sendo assim, as condições iniciais foram selecionadas de forma aleatória, pois, independentemente do estado inicial escolhido para essas variáveis o sistema converge para o mesmo estado estacionário final. Nesse contexto, o ponto de trabalho ideal, determinado por q_r = 0,0032 m³/min, Q_c = 144,5 kJ/min, C_{A0} = 5,1 kmol/m³ e T_{r0} = 387,05 K, é apresentado na Tabela 8.

Tabela 8 – Valores dos parâmetros no ponto ótimo de operação.

Variável	Valor de Estado Estacionário Ideal
C_A^S	$2{,}194\ kmol/m^3$
C_B^S	$1{,}095\ kmol/m^3$
T_r^S	$391{,}38\ K$
T_c^S	$401{,}39\ K$

Fonte: Autor (2023).

5.2 Geração de Dados

A análise para determinação do tempo de amostragem do sistema foi conduzida levando em consideração o período necessário para o sistema alcançar um novo estado estacionário, após experimentar perturbações em suas variáveis em torno do ponto de operação utilizado como referência (SILVA, 2022). Os resultados dessa avaliação estão apresentados na Tabela 9.

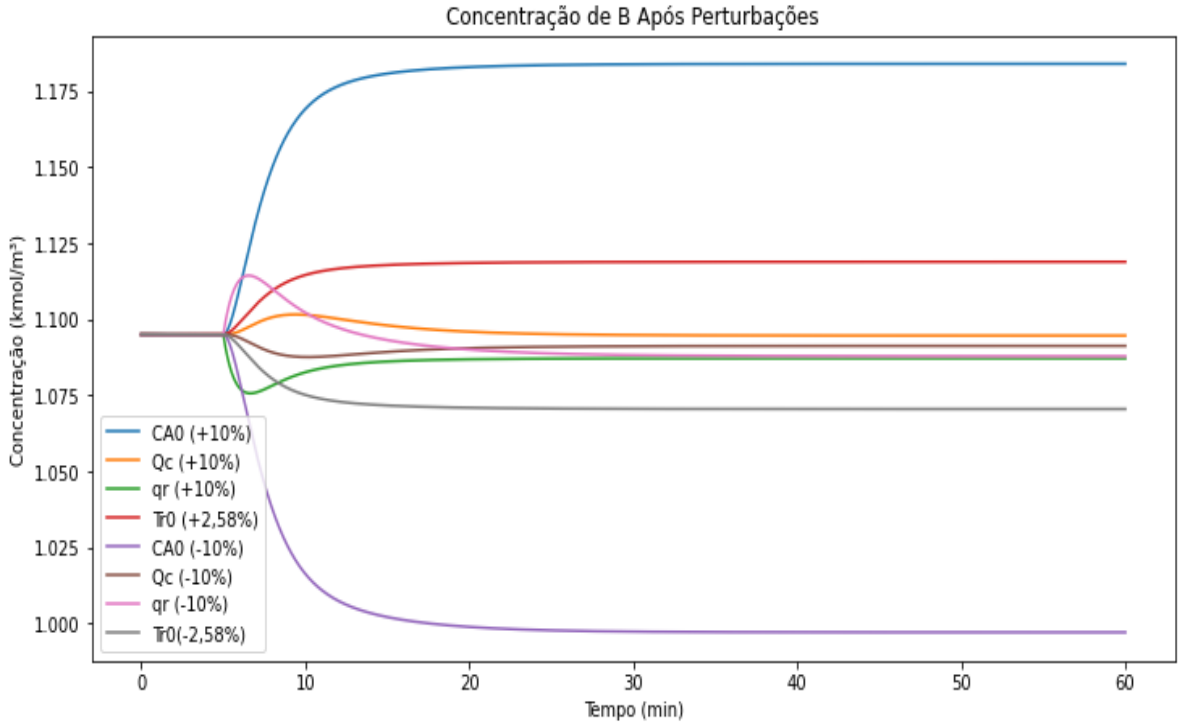
Tabela 9 – Tempos para atingir o novo estado estacionário.

Descrição	Valor	Tempo para o novo E. E.
q_r (+10%)	0,00352	491,64 s
q_r (–10%)	0,00288	620,81 s
Q_c (+10%)	158,95	579,30 s
Q_c (–10%)	130,05	571,80 s
C_{A0} (+10%)	5,61	700,20 s
C_{A0} (–10%)	4,59	687,60 s
T_{r0} (+2,58%)	397,05	1126 s
T_{r0} (–2,58%)	377,05	1031,4 s

Fonte: Autor (2023).

Dessa forma, 491,64 segundos representa o menor tempo que o sistema leva para alcançar o novo estado estacionário, enquanto o tempo de 1126 segundos corresponde ao maior tempo. A Figura 14 ilustra o comportamento do sistema após sofrer as perturbações correspondentes aos limites inferiores e superiores das variáveis manipuladas e de distúrbio, individualmente. A partir da Figura 14, é possível observar a influência de cada variável no comportamento do sistema.

Figura 14 – Comportamento da concentração de B após perturbações no sistema.



Fonte: Autor (2023).

Conforme a Figura 14, nota-se que a concentração de B é mais sensível a variações em C_{A0}. Ao analisar a curva correspondente a T_{r0}, observa-se que, após um aumento de 2,58%, ocorre um crescimento na concentração de B até atingir um pico. O mesmo acontece após uma redução de 2,58%, com a concentração diminuindo até um valor mínimo. O pico de concentração é alcançado após 65 segundos. Logo, levando em consideração o tempo de 65 segundos, os tempos de amostragem de 3, 6, 9 e 12 segundos foram selecionados convenientemente, de modo a preservar as informações sobre essa dinâmica do processo, uma vez que são tempos menores que o tempo para atingir o pico de concentração.

A Tabela 10 exibe as informações sobre a base de dados obtida na etapa de geração de dados para o desenvolvimento dos sensores virtuais.

Tabela 10 – Informações sobre a base de dados do processo.

Descrição	Quantidade
Número de perturbações em T_{r0}	24.995
Número de perturbações em C_{A0}	24.988
Número de perturbações em Q_c	24.788
Número de perturbações em q_r	25.020
Dados com tempo de 3s	11.962.180
Dados com tempo de 6s	5.981.090
Dados com tempo de 9s	3.987.393
Dados com tempo de 12s	2.990.545

Fonte: Autor (2023).

A base de dados com tempo de amostragem de 3 segundos resultou em um arquivo do tipo CSV com tamanho de 1,594 GB. As bases de dados de 6, 9 e 12 segundos, por sua vez, resultaram em arquivos do tipo CSV com 0,717 GB, 0,477 GB e 0,358 GB, respectivamente. A partir das informações sobre o tamanho dos arquivos é possível concluir que, quanto menor o tempo de amostragem, maior será o uso de recursos computacionais. O grande volume de dados gerados na simulação justifica o uso o banco de dados relacional, uma vez que há uma limitação computacional na memória RAM, sendo capaz de armazenar temporariamente apenas 16 MB. A grande quantidade de dados visa garantir que o modelo de sensor virtual seja capaz de aprender uma grande variedade de condições operacionais, aumentando a sua capacidade de generalizar para conjuntos de dados novos.

5.3 Sensor Virtual

5.3.1 Rede Neural Artificial

Para cada conjunto de dados analisado, as métricas de desempenho consideradas foram o MSE, MAE e o R^2. Os valores das métricas estão apresentados nas Tabelas 11-14.

Tabela 11 – Métricas do modelo de RNA para o tempo de amostragem de 3 segundos.

Dados	R^2	MAE	MSE
3 segundos e 1 atraso	0,9999	$1,36x10^{-4}$	$4,49x10^{-8}$
3 segundos e 2 atrasos	0,9999	$2,10x10^{-4}$	$4,77x10^{-8}$
3 segundos e 3 atrasos	0,9999	$1,11x10^{-4}$	$1,29x10^{-8}$

Fonte: Autor (2023).

Tabela 12 - Métricas do modelo de RNA para o tempo de amostragem de 6 segundos.

Dados	R^2	MAE	MSE
6 segundos e 1 atraso	0,9999	$2,56x10^{-4}$	$1,57x10^{-7}$
6 segundos e 2 atrasos	0,9999	$2,58x10^{-4}$	$8,36x10^{-8}$
6 segundos e 3 atrasos	0,9998	$4,87x10^{-4}$	$2,43x10^{-7}$

Fonte: Autor (2023).

Tabela 13 - Métricas do modelo de RNA para o tempo de amostragem de 9 segundos.

Dados	R^2	MAE	MSE
9 segundos e 1 atraso	0,9998	$2,73x10^{-4}$	$2,89x10^{-7}$
9 segundos e 2 atrasos	0,9998	$4,39x10^{-4}$	$2,31x10^{-7}$
9 segundos e 3 atrasos	0,9999	$1,38x10^{-4}$	$2,10x10^{-8}$

Fonte: Autor (2023).

Tabela 14 - Métricas do modelo de RNA para o tempo de amostragem de 12 segundos.

Dados	R^2	MAE	MSE
12 segundos e 1 atraso	0,9995	$6,86x10^{-4}$	$8,61x10^{-7}$
12 segundos e 2 atrasos	0,9999	$2,00x10^{-4}$	$1,25x10^{-7}$
12 segundos e 3 atrasos	0,9999	$1,76x10^{-4}$	$5,57x10^{-8}$

Fonte: Autor (2023).

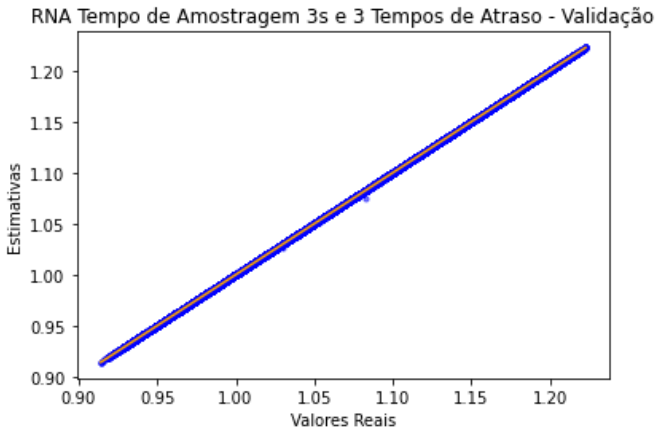
A análise das métricas de desempenho revela que os tempos de amostragem de 3 segundos e 9 segundos, ambos com 3 tempos de atraso nas variáveis de entrada, exibiram os melhores resultados. Ambos os modelos alcançaram um R^2 superior a 0,9999 e um MAE inferior a $1,38x10^{-4}$. Apesar de os dois modelos apresentarem o melhor desempenho, todos os

modelos analisados podem ser aplicados na implementação do sensor virtual, uma vez que o modelo com pior desempenho obteve R^2 de 0,9995 e MAE de $6{,}86 \times 10^{-4}$.

Os modelos de sensores virtuais apresentados foram desenvolvidos com base no modelo de RNA que demonstrou a melhor capacidade de generalização para o conjunto de dados novos. Dessa forma, o modelo foi construído com três camadas de neurônios: uma camada de entrada, uma camada oculta e uma camada de saída. A camada de entrada continha 16 neurônios e utilizou a função de ativação ‘relu’; a camada oculta empregou 8 neurônios e função de ativação ‘relu’; e, na camada de saída, foi usado 1 neurônio com função de ativação ‘linear’. Durante o treinamento dos modelos, o otimizador ‘adam’ e a função de perda ‘MSE’ foram empregados.

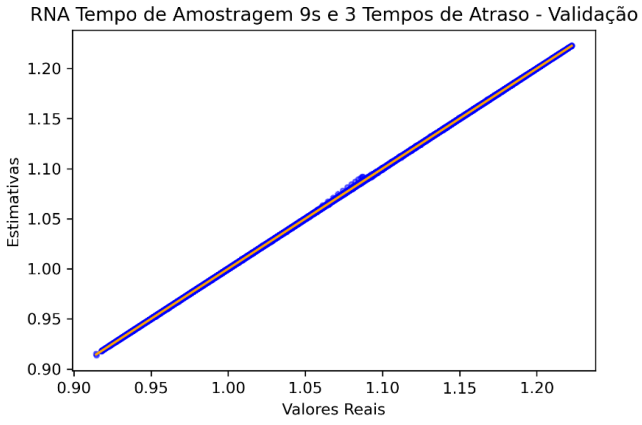
As Figuras 15 e 16 mostram os gráficos de dispersão referentes aos modelos de RNA com melhor desempenho.

Figura 15 – Gráfico de dispersão para a RNA com tempo de amostragem de 3 s e 3 tempos de atraso.



Fonte: Autor (2023).

Figura 16 – Gráfico de dispersão para a RNA com tempo de amostragem de 9 s e 3 tempos de atraso.

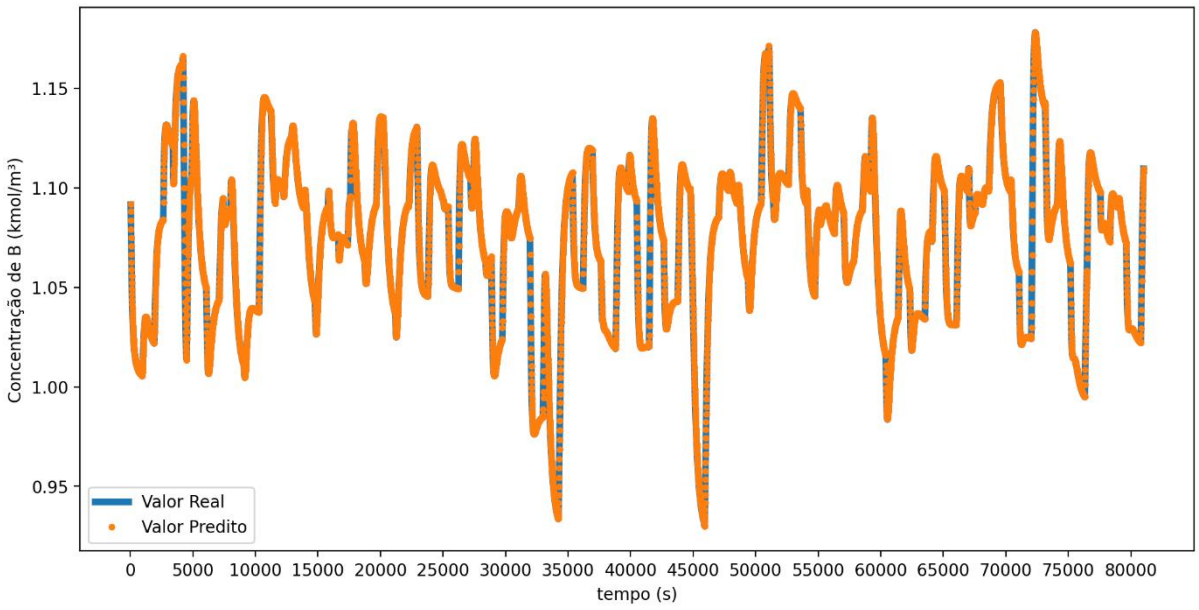


Fonte: Autor (2023).

Nas Figuras 15-16, é possível observar o excelente desempenho de ambos os modelos, uma vez que os dados estão bem ajustados à diagonal principal. Esse resultado evidencia a eficácia das RNAs na estimação da concentração do produto B em diferentes condições operacionais do reator.

A Figura 17 exibe o perfil temporal da concentração de B, comparando os valores previstos pelo sensor virtual, treinado com dados de tempo de amostragem de 9 segundos e 3 tempos de atraso, e os valores reais, utilizando dados desconhecidos pela RNA – isto é, dados que não foram empregados no treinamento do modelo. Os perfis foram construídos a partir de uma simulação do funcionamento do sensor virtual no contexto de uma operação dinâmica do reator.

Figura 17 – Perfil da predição dinâmica da concentração de B com RNA.



Fonte: Autor (2023).

A figura mostra um excelente ajuste entre os perfis de dados reais e preditos, com uma curva praticamente sobreposta à outra, evidenciando a precisão e a eficácia do modelo proposto. As métricas de desempenho calculadas para avaliar a qualidade do modelo na operação dinâmica reforçam essa conclusão. O R^2 obtido foi superior a 0,9999, indicando uma alta correlação entre os valores reais e preditos, e, portanto, uma excelente capacidade do modelo de explicar a variabilidade dos dados. Além disso, o MSE foi inferior a $5{,}9 \times 10^{-8}$ e o MAE foi de $1{,}722 \times 10^{-4}$, ambos os valores demonstram a proximidade dos dados preditos em relação aos dados reais e a baixa magnitude dos erros cometidos pelo sensor virtual.

Os resultados sugerem que o modelo de RNA proposto é altamente eficiente na estimação da concentração do produto B em diferentes condições operacionais. A precisão do modelo pode ser atribuída ao volume de dados empregados no processo de treinamento e à seleção adequada de parâmetros e arquitetura da rede neural.

5.3.2 Máquina de Vetores de Suporte

Os sensores virtuais empregando SVM foram desenvolvidos com base nos hiperparâmetros apresentados no Quadro 6. Esses hiperparâmetros foram selecionados por meio da aplicação da técnica de validação cruzada durante a fase de treinamento do modelo.

Quadro 6 – Hiperparâmetros selecionados para o modelo de SVM.

Hiperparâmetro	**Valor**
C	0,5
Gamma	1
Kernel	RBF

Fonte: Autor (2023).

Para cada tempo de amostragem, foram calculadas as métricas de desempenho correspondentes aos diferentes tempos de atraso. Os modelos foram avaliados com base nos valores de R^2, MAE e MSE. As Tabelas 15-18 apresentam os resultados obtidos ao empregar a técnica de SVM no desenvolvimento dos sensores virtuais para estimação da concentração do composto B.

Tabela 15 - Métricas do modelo de SVM para o tempo de amostragem de 3 segundos.

Dados	**R^2**	**MAE**	**MSE**
3 segundos e 1 tempo de atraso	0,6342	$2{,}08 \times 10^{-2}$	$6{,}66 \times 10^{-4}$
3 segundos e 2 tempos de atraso	0,6495	$1{,}98 \times 10^{-2}$	$6{,}38 \times 10^{-4}$
3 segundos e 3 tempos de atraso	0,6268	$2{,}04 \times 10^{-2}$	$6{,}78 \times 10^{-4}$

Fonte: Autor (2023).

Tabela 16 - Métricas do modelo de SVM para o tempo de amostragem de 6 segundos.

Dados	R^2	MAE	MSE
6 segundos e 1 tempo de atraso	0,6708	$1,96x10^{-2}$	$5,95x10^{-4}$
6 segundos e 2 tempos de atraso	0,6962	$1,80x10^{-2}$	$5,49x10^{-4}$
6 segundos e 3 tempos de atraso	0,6665	$1,87x10^{-2}$	$6,03x10^{-4}$

Fonte: Autor (2023).

Tabela 17 - Métricas do modelo de SVM para o tempo de amostragem de 9 segundos.

Dados	R^2	MAE	MSE
9 segundos e 1 tempo de atraso	0,7042	$1,83x10^{-2}$	$5,33x10^{-4}$
9 segundos e 2 tempos de atraso	0,7082	$1,76x10^{-2}$	$5,26x10^{-4}$
9 segundos e 3 tempos de atraso	0,6707	$1,86x10^{-2}$	$5,94x10^{-4}$

Fonte: Autor (2023).

Tabela 18 - Métricas do modelo de SVM para o tempo de amostragem de 12 segundos.

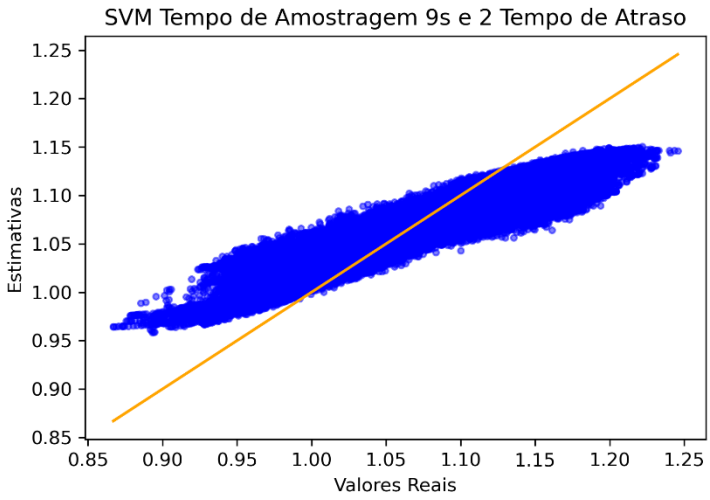
Dados	R^2	MAE	MSE
12 segundos e 1 tempo de atraso	0,7024	$1,82x10^{-2}$	$5,37x10^{-4}$
12 segundos e 2 tempos de atraso	0,7031	$1,76x10^{-2}$	$5,36x10^{-4}$
12 segundos e 3 tempos de atraso	0,6543	$1,93x10^{-2}$	$6,24x10^{-4}$

Fonte: Autor (2023).

A partir das tabelas é possível verificar que os valores da métrica de R^2 variam entre 0,6342 e 0,7082, indicando uma correlação moderada entre os valores reais e preditos, o que sugere uma capacidade limitada dos modelos em explicar a variabilidade dos dados. Além disso, os valores de MSE e MAE são relativamente altos, apontando para uma grande discrepância entre os dados preditos e os valores reais, o que implica em erros consideráveis nas estimativas fornecidas.

A Figura 18 exibe o gráfico de dispersão que ilustra as estimativas fornecidas pelo modelo de SVM com melhor desempenho, permitindo uma visualização clara da precisão alcançada pelo sensor virtual.

Figura 18 – Gráfico de dispersão para a SVM com tempo de amostragem de 9 s e 2 tempos de atraso.



Fonte: Autor (2023).

O gráfico de dispersão também corrobora a baixa qualidade do modelo, mostrando uma distribuição dos dados que se concentra em torno de uma reta diferente da diagonal principal, o que indica um ajuste fraco entre os valores reais e preditos. Dessa forma, a construção do gráfico de predição dinâmica da concentração de B não foi necessário, uma vez que os valores das métricas de desempenho e o gráfico de dispersão já comprovam a baixa qualidade do modelo.

Os resultados sugerem que os modelos de SVM propostos não são adequados para a estimação da concentração do composto B em diferentes condições operacionais. A má performance dos modelos pode ser atribuída a possíveis fatores como: seleção inadequada de parâmetros do SVM, falta de otimização do kernel e complexidade intrínseca do processo químico, que não foi capturada pelo modelo de SVM proposto.

5.3.3 Comparação dos Modelos de Sensores Virtuais

A comparação entre os modelos baseou-se nas métricas de desempenho R^2, MAE e MSE. Os modelos de RNA demonstraram um excelente desempenho, com R^2 superior a 0,9999 e MAE próximo de 0, em comparação aos modelos de SVM, que apresentaram valores de R^2 entre 0,6342 e 0,7082, indicando correlação moderada e uma capacidade limitada em explicar a variabilidade dos dados.

Por outro lado, os modelos de SVM mostraram-se inadequados para a estimação da concentração do composto B, apresentando erros consideráveis nas estimativas fornecidas. Um dos possíveis motivos para o desempenho insatisfatório da SVM pode ser a seleção inadequada de hiperparâmetros. Uma alternativa para solucionar esse problema seria explorar diferentes conjuntos de hiperparâmetros. No entanto, devido ao tempo computacional exigido no treinamento dos modelos, não foi viável realizar testes com outras combinações durante este trabalho, com o treinamento de cada modelo levando em média cerca de 12 horas.

5.4 Detecção e Diagnóstico de Falhas

A base de dados utilizada nesta etapa foi selecionada com base no tempo de amostragem que apresentou o melhor desempenho no desenvolvimento do sensor virtual. Nesse sentido, empregou-se a base de dados com tempo de amostragem de 9 segundos, gerando um total de 3.987.429 dados relacionados ao processo. A Tabela 19 apresenta as quantidades de dados para cada classe presente na base de dados empregada no desenvolvimento dos modelos de detecção e diagnóstico de falhas, utilizando as técnicas de RNA e SVM.

Tabela 19 – Informações sobre a base de dados de falhas do processo.

Classe do Processo	Quantidade de Dados
Normal	1.994.439
Falha 1	332.780
Falha 2	331.598
Falha 3	333.117
Falha 4	331.601
Falha 5	331.647
Falha 6	332.247

Fonte: Autor (2023).

Conforme observado, a base de dados engloba duas classes principais: dados de processo sem falhas e dados de processo com falhas. Na base de dados, contabilizam-se 1.994.439 dados referentes ao processo sem falhas e 1.992.990 dados referentes ao processo com falhas, atenuando assim o problema de desbalanceamento entre as classes. Ademais, ao

analisar os dados do processo com falhas, verifica-se que as diferentes classes de falha estão equilibradas, uma vez que apresentam quantidades de dados similares.

5.4.1 Rede Neural Artificial

A Tabela 20 mostra os resultados obtidos para o modelo de detecção e diagnóstico de falhas utilizando a técnica de RNA.

Tabela 20 – Métricas de desempenho do modelo de RNA.

Métricas de Desempenho	Valores
Acurácia	0,97
Recall	0,97
F1-Score	0,97
Taxa de Erro	0,03

Fonte: Autor (2023).

O modelo apresentou uma acurácia de 97%, indicando um alto grau de precisão das previsões realizadas. Além disso, o recall e o F1-Score, que são medidas de sensibilidade e equilíbrio entre precisão e recall, respectivamente, também apresentaram valores elevados (97% para ambos). Esses resultados sugerem que o modelo é eficiente na identificação das instâncias de cada classe e na minimização de falsos positivos e falsos negativos.

A Tabela 21 apresenta as principais métricas de desempenho de tarefas de classificação, mostrando individualmente o desempenho das diferentes classes.

Tabela 21 – Métricas de desempenho do modelo de RNA por classe.

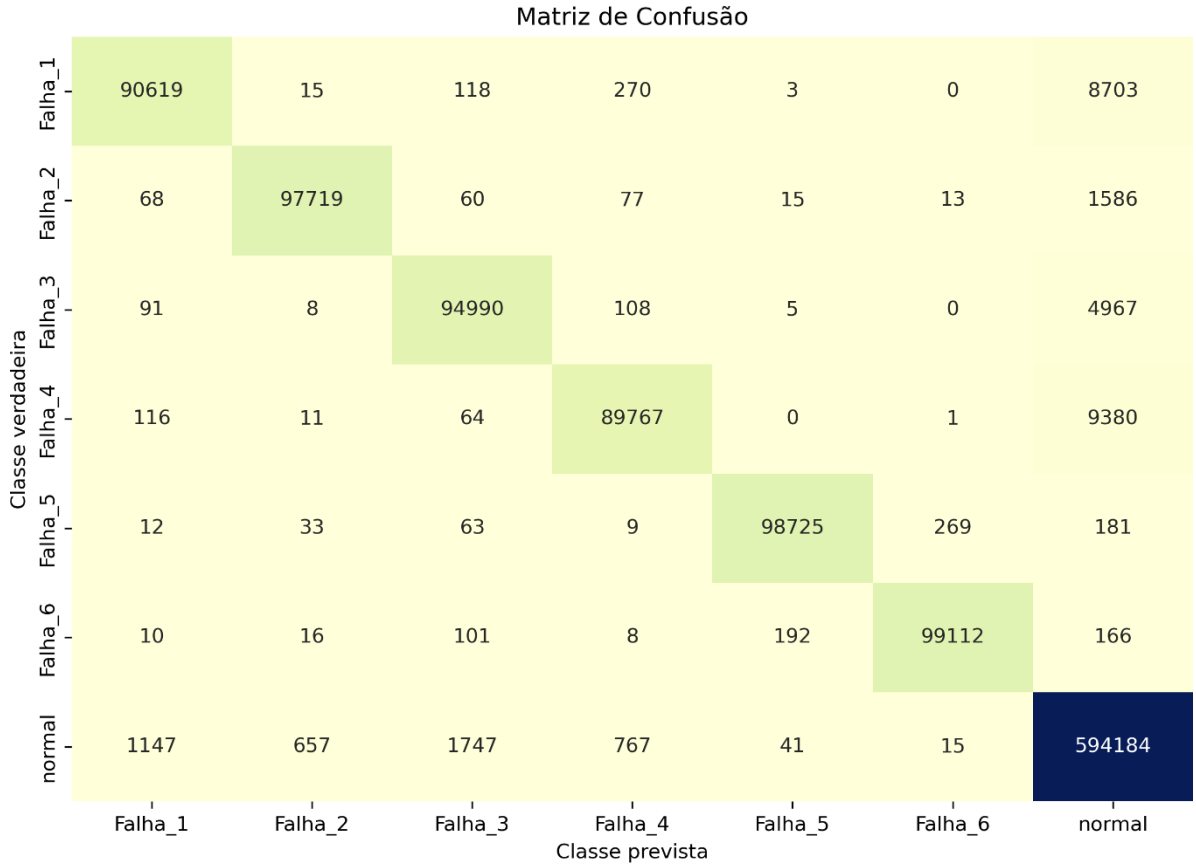
Classe	Precisão	Recall	F1-Score	Support
Falha 1	0,98	0,91	0,94	99.278
Falha 2	0,99	0,98	0,99	99.538
Falha 3	0,98	0,95	0,96	100.169
Falha 4	0,99	0,90	0,94	99.339
Falha 5	1,00	0,99	1,00	99.292
Falha 6	1,00	1,00	1,00	99.605
Normal	0,96	0,99	0,98	598.558

Fonte: Autor (2023).

A análise por classe mostra que a precisão varia de 0,96 (classe Normal) a 1,00 (Falhas 5 e 6), enquanto o recall varia de 0,90 (Falha 4) a 1,00 (Falha 6). O F1-Score, por sua vez, varia de 0,94 (Falhas 1 e 4) a 1,00 (Falhas 5 e 6). Os resultados indicam que o modelo apresenta um bom desempenho em identificar corretamente cada classe, mesmo com variações entre as classes. Além disso, a taxa de erro foi de apenas 3%, demonstrando a capacidade do modelo em fornecer previsões corretas para a maioria das instâncias.

A Figura 19 mostra a matriz de confusão utilizada para avaliar o desempenho do modelo no conjunto de dados de teste. Essa matriz fornece um resumo das previsões corretas e incorretas realizadas pelo modelo de aprendizado de máquina durante a etapa de validação, comparando os valores previstos com os valores reais das classes. Isso permite verificar a capacidade de generalização do modelo para conjuntos de dados não vistos na etapa de treinamento.

Figura 19 – Matriz de confusão para RNA.



Fonte: Autor (2023).

Examinando os resultados, é possível observar que o modelo apresenta bom desempenho na classificação. Entretanto, existem classes em que o modelo demonstra dificuldade, principalmente ao trabalhar com dados da classe “Normal”. Essa dificuldade pode ser atribuída à similaridade entre os padrões de algumas falhas e o comportamento normal do processo. Contudo, apenas 0,73% dos dados da classe “normal” são classificados incorretamente como provenientes de falhas no processo. Em relação às previsões de dados normais realizadas pelo modelo, 4% são classificadas incorretamente como pertencentes a classe “normal”, quando na verdade são oriundos do processo com falhas. De maneira geral, o modelo apresenta uma taxa de erro de 2,60% em suas previsões.

O desempenho do modelo no diagnóstico e detecção de falhas é mostrado na Tabela 22 e Tabela 23, respectivamente.

Tabela 22 – Desempenho da RNA no diagnóstico de falhas.

Desempenho no Diagnóstico de Falhas	
Diagnósticos realizados	577.062
Taxa de erros (%)	1,06

Fonte: Autor (2023).

Tabela 23 - Desempenho da RNA na detecção de falhas.

Desempenho na Detecção de Falhas	
Número de dados com falhas	597.671
Taxa de erros (%)	4,18

Fonte: Autor (2023).

Quanto à detecção da existência de falhas no processo, o modelo exibiu uma taxa de erro de 4,18%, detectando as falhas em 95,82% dos casos. Em relação ao diagnóstico de falhas, o modelo identificou corretamente 98,99% dos tipos específicos de falhas. Logo, o modelo apresenta um ótimo desempenho nas tarefas de detecção e diagnóstico de falhas do processo.

Com base nestes resultados, é possível afirmar que o modelo pode ser aplicado na identificação e classificação de falhas, contribuindo para a melhoria da eficiência e segurança do processo.

5.4.2 Máquinas de Vetores de Suporte

As métricas de desempenho obtidas para o algoritmo proposto são apresentadas na Tabela 24.

Tabela 24 - Métricas de desempenho do modelo de SVM.

Métricas de Desempenho	**Valores**
Acurácia	0,95
Recall	0,95
F1-Score	0,95
Taxa de Erro	0,05

Fonte: Autor (2023).

A Tabela 25 mostra os resultados detalhados para cada classe, incluindo precisão, recall, F1-Score e Support.

Tabela 25 - Métricas de desempenho do modelo de SVM por classe.

Classe	Precisão	Recall	F1-Score	Support
Falha 1	1,00	0,84	0,91	29.856
Falha 2	1,00	0,94	0,97	29.847
Falha 3	1,00	0,85	0,92	29.998
Falha 4	1,00	0,84	0,91	29.543
Falha 5	1,00	0,95	0,97	29.978
Falha 6	1,00	0,95	0,97	29.943
Normal	0,91	1,00	0,95	179.704

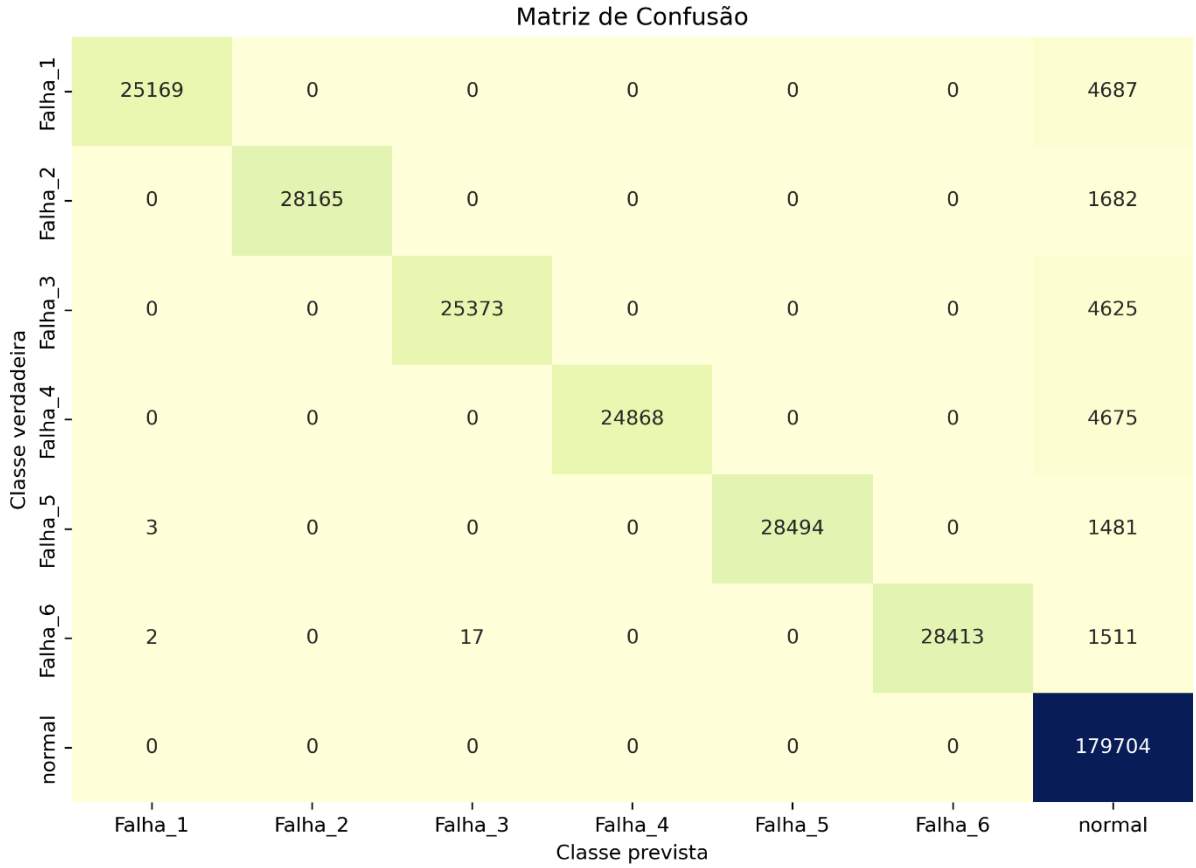
Fonte: Autor (2023).

Analisando os resultados detalhados por classe, é possível observar que o algoritmo tem um desempenho excelente para as falhas, com precisão igual a 1,00 para todas as classes de falhas. Entretanto, o desempenho é inferior para a classe “normal”, com uma precisão de 0,91. Esta diferença pode ser atribuída a vários fatores, como o desbalanceamento entre as classes e similaridade entre uma parte dos dados com falhas e os dados do processo normal. Além disso, é importante ressaltar que a classe “normal” possui um *support* significativamente maior em comparação às classes de falhas, o que pode contribuir para a menor precisão.

O recall e F1-Score também são consistentemente altos para todas as classes, indicando que o modelo é capaz de identificar corretamente a maioria dos casos verdadeiros positivos e minimizar os falsos negativos. A taxa de erro de 0,05 indica que o modelo cometeu erros em torno de 5% das previsões realizadas.

A Figura 20 apresenta a matriz de confusão obtida ao avaliar o desempenho do modelo no conjunto de dados de teste. A matriz de confusão fornece um resumo das previsões corretas e incorretas realizadas pelo modelo durante a etapa de validação, permitindo verificar a capacidade de generalização do modelo para conjuntos de dados não vistos na etapa de treinamento.

Figura 20 – Matriz de confusão para SVM.



Fonte: Autor (2023).

Analisando os resultados, o modelo demonstrou um bom desempenho na classificação das diferentes classes de falhas, bem como na detecção de eventos normais. Em relação a previsão das falhas, o modelo conseguiu classificá-las corretamente, indicando uma alta taxa de precisão na tarefa de diagnóstico. No entanto, visualizando a matriz de confusão, é possível verificar que o modelo tem dificuldade em diferenciar dados da classe “normal” das classes de falha. Essa dificuldade também foi observada no modelo utilizando Redes Neurais Artificias e, de modo análogo, pode ser atribuída a semelhança no padrão de comportamento dos dados de processo com falhas com o padrão de comportamento dos dados de processo normais.

A Tabela 26 apresenta as métricas de desempenho do modelo na tarefa de diagnóstico de falhas.

Tabela 26 - Desempenho da SVM no diagnóstico de falhas.

Desempenho no Diagnóstico de Falhas	
Diagnósticos realizados	160.504
Taxa de erros (%)	0,013

Fonte: Autor (2023).

Os resultados mostram a alta eficiência do algoritmo em identificar o tipo específico de falha existente no processo, acertando em mais de 99,98% dos casos.

Para análise do desempenho do modelo na tarefa de detecção de falhas, a Tabela 27 apresenta as métricas correspondentes.

Tabela 27 - Desempenho da SVM na detecção de falhas.

Desempenho na Detecção de Falhas	
Número de dados com falhas	179.165
Taxa de erros (%)	10,41

Fonte: Autor (2023).

A Tabela 27 mostra que o modelo detectou 10,41% dos dados com falhas, classificando-os como dados do processo em operação com falhas nos sensores. Portanto, apesar do ótimo desempenho no diagnóstico, o modelo apresentou um desempenho razoável na tarefa de detecção, identificando apenas 89,59% dos casos de falhas.

A análise das métricas de desempenho sugere que o algoritmo proposto é eficaz nas tarefas de detecção e diagnóstico de falhas no reator químico em estudo.

5.4.3 Comparação dos Modelos de Detecção e Diagnóstico de Falhas

Comparando os resultados obtidos com o uso de ambas as técnicas, é possível verificar que o modelo de Redes Neurais Artificiais foi superior na etapa de detecção de falha, detectando 95,82% dos dados de casos com falhas. Por outro lado, o modelo de Máquinas de Vetores de Suporte teve melhor desempenho no diagnóstico das falhas, identificando corretamente os tipos específicos de falhas em mais de 99,98% dos casos. Dessa forma, uma possível melhoria do processo de diagnóstico e detecção de falhas seria o uso integrado de ambas as técnicas, empregando as Redes Neurais Artificiais na detecção e as Máquinas de Vetores de Suporte na identificação do tipo de falha.

6 CONCLUSÃO

Neste trabalho, foi abordado o desenvolvimento de sensores virtuais e estratégias de detecção e diagnóstico de falhas em um processo reacional complexo, utilizando RNA e SVM. Através da modelagem matemática de um reator CSTR operando com a reação de Van de Vusse, foram implementados modelos computacionais em Python para simular diferentes condições operacionais e gerar dados para o desenvolvimento dos sensores virtuais e diagnóstico de falhas.

Os principais resultados mostram que as técnicas de RNA apresentaram desempenho superior na criação de sensores virtuais, com alta precisão e eficácia na estimação da concentração do produto B. Por outro lado, os modelos de SVM se mostraram inadequados para essa tarefa, com baixa qualidade e ajuste fraco entre os valores reais e preditos.

No que diz respeito à detecção e diagnóstico de falhas, o modelo de RNA obteve os melhores resultados na detecção de falhas, enquanto a SVM apresentou melhor desempenho no diagnóstico de falhas. Uma possível melhoria no processo de diagnóstico e detecção de falhas seria a integração das técnicas, utilizando RNA na detecção e SVM na identificação do tipo de falha.

O presente trabalho contribui para a otimização do desempenho do reator e o aumento da eficiência da produção do composto de interesse, fornecendo uma base para a implementação de estratégias de controle e monitoramento em processos reacionais complexos. Além disso, os resultados obtidos podem ser aplicados em outras áreas da indústria química, onde a detecção e o diagnóstico de falhas são cruciais para a segurança e a eficiência dos processos. Apesar dos resultados promissores, algumas limitações ainda persistem, como a seleção inadequada de parâmetros e a complexidade intrínseca do processo químico. Futuros trabalhos podem se concentrar na otimização dos parâmetros, no uso de outras técnicas de aprendizado de máquina e na investigação de outras abordagens para melhorar ainda mais o desempenho dos sensores virtuais e dos modelos de detecção e diagnóstico de falhas.

REFERÊNCIAS

ACHSAN, B. M. **Support Vector Machine**: Regression.

ÅSTRÖM, Karl Johan; HÄGGLUND, Tore. **Advanced PID Control**. ISA, 2006.

BRITO, Alexandra Antonia Freitas de. A Quarta Revolução Industrial e as Perspectivas para o Brasil. **Revista Científica Multidisciplinar Núcleo do Conhecimento**, [s.l.], v. 2, n. 7, p. 91-96, out. 2017. Disponível em: https://www.nucleodoconhecimento.com.br/administracao/quarta-revolucaoindustrial#_ftn1. Acesso em: 12 setembro 2022.

BRUCE, Peter; BRUCE, Andrew. **Estatística Prática para Cientista de Dados**: 50 conceitos essenciais. Rio de Janeiro: Alta Books Editora, 2019.

CACCAVALE, F. et al. An integrated approach to fault diagnosis for a class of chemical batch processes. **Journal of Process Control**, v. 19, p. 827-841, 2009.

CNI (Confederação Nacional da Indústria). **Relações trabalhistas no contexto da indústria 4.0. Brasília**: Confederação Nacional da Indústria, 2017. 71 p.

FOGLER, H. S. **Elementos de Engenharia das Reações Químicas**. 4. ed. Rio de Janeiro: LTC, 2009.

FORTUNA, L.; GRAZIANI, S.; RIZZO, A.; XIBILIA, M.G. **Soft sensors for monitoring and control of industrial processes**. Londres: Springer, 2007.

GÉRON, A. **Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow**: conceitos, ferramentas e técnicas para construção de sistemas inteligentes. Rio de Janeiro: Alta Books Editora, 2019.

HARRISON, Matt. **Machine Learning Guia de Referência Rápido**: trabalhando com dados estruturados em python. São Paulo: Novatec Editora Ltda, 2019.

HARRISON, R.; VERA, D.; AHMAD, B. Engineering Methods and Tools for Cyber–Physical Automation Systems. **Proceedings of the IEEE**, v. 104, n. 5, p. 973-985, 2016. DOI: 10.1109/JPROC.2015.2510665.

HAYKIN, S. **Redes Neurais, princípios e prática**. 2. ed. Porto Alegre: Bookman, 2001.

HIMMELBLAU, D. M. **Fault detection and diagnosis in chemical and petrochemical processes**. Amsterdam: Elsevier Press, 1978.

KADLEC, Petr; GABRYS, Bogdan; STRANDT, Sibylle. Data-driven Soft Sensors in the process industry. **Computers & Chemical Engineering**, [S.L.], v. 33, n. 4, p. 795-814, abr. 2009. Elsevier BV. Disponível em: http://dx.doi.org/10.1016/j.compchemeng.2008.12.012. Acesso em: 24 jan. 2023.

MATA, V. S. et al. Indústria 4.0: a Revolução 4.0 e o Impacto na Mão de Obra. **Revista de Ciências Exatas e Tecnologia**, v. 13, n. 13, p. 17-22, 2018. DOI: http://dx.doi.org/10.17921/1890-1793.2018v13n13p17-22. Acesso em: 12 setembro 2022.

MOHAMMADI, M. et al. A comprehensive survey and taxonomy of the SVM-based intrusion detection systems. **Journal of Network and Computer Applications**, v. 178, p. 102983, mar. 2021.

RÜßMANN, Michael et al. **Industry 4.0:** The future of productivity and growth in manufacturing industries. Boston Consulting Group, v. 9, 2015.

SARAIVA, S. V. **Estudo de um controlador preditivo baseado em sistemas inteligentes**. Trabalho de Conclusão de Curso. Faculdade de Engenharia Química, Universidade Federal de Alagoas, 2017.

SCHNITMAN, L. **Controladores preditivos baseados em redes neurais artificiais**. Dissertação de mestrado em Engenharia Elétrica. Salvador: Universidade Federal da Bahia, 1998.

SCHWAB, K. **A quarta revolução industrial**. Tradução: Daniel Moreira Miranda. São Paulo: Edipro, 2016.

SILVA, José Diego da. **Aprendizagem de Máquina Aplicada ao Desenvolvimento de Sensores Virtuais e na Detecção e Diagnóstico de Falhas para Processos Químicos Reativos em Linguagem Python**. 2022. 100 f. TCC (Graduação) - Curso de Engenharia Química, Centro de Tecnologia, Universidade Federal de Alagoas, Maceió, 2022.

SMOLA, A. J.; BARTLETT, P. L.; SCHOLKOPF, B.; SCHUURMANS, D. **Advances in Large Margin Classifiers**. Londres: The Mit Press, 2000.

TAQVI, S. A. A.; ZABIRI, H.; TUFA, L. D.; UDDIN, F.; FATIMA, S. A.; MAULUD, A. S. A Review on Data-Driven Learning Approaches for Fault Detection and Diagnosis in Chemical Processes. **Chembioeng Reviews**, v. 8, n. 3, p. 239-259, 26 fev. 2021.

VAN DE VUSSE, J. G. Plug-flow type reactor versus tank reactors. **Chemical Engineering Science**, v. 19, n. 1, p. 994-996, 1964.

VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; YIN, K.; KAVURI, S. N. A review of process fault detection and diagnosis. **Computers & Chemical Engineering**, [S.l.], v. 27, n. 3, p. 293-311, mar. 2003. Elsevier BV. Disponível em: http://dx.doi.org/10.1016/s0098-1354(02)00160-6.

VOJTĚŠEK, Jiří. **Chemical Reactors**: Modern Control Methods. Tese de Doutorado - Tomas Bata University in Zlín, Faculty of Applied Informatics, Zlín, 2007. 167 p.

VAPNIK, Vladimir N. **Statistical Learning Theory**. New York: Wiley, 1998.

VIAN, C. E. F. **História Econômica Geral**. Materiais de aula. 2015. Disponível em: <https://edisciplinas.usp.br/pluginfile.php/3374875/mod_folder/content/0/HEG%20-%2

0Aula%208-%20revolu%C3%A7%C3%A3o%20industrial%202015.ppt?forcedownload=1>. Acesso em: 20 novembro 2022.