UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS JOINVILLE
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE SISTEMAS ELETRÔNICOS

Felipe de Pontes Adachi

Aplicação de Técnicas de Aprendizado de Máquina para Detecção de Falhas em Veículos Subaquáticos Não-tripulados

DISSERTAÇÃO DE MESTRADO - DM-PPGESE-005

Joinville
2019

Felipe de Pontes Adachi

Aplicação de Técnicas de Aprendizado de Máquina para Detecção de Falhas em Veículos Subaquáticos Não-tripulados

Dissertação submetida ao Programa de Pós-Graduação em Engenharia de Sistemas Eletrônicos da Universidade Federal de Santa Catarina para a obtenção do título de mestre em Engenharia de Sistemas Eletrônicos.
Orientador: Prof. Antônio Augusto Fröhlich, Dr.
Coorientador: Prof. Pablo Andretta Jaskowiak, Dr.

Joinville
2019

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Adachi, Felipe de Pontes
Aplicação de técnicas de aprendizado de máquina para detecção de falhas em veículos subaquáticos não-tripulados / Felipe de Pontes Adachi ; orientador, Antônio Augusto Fröhlich, coorientador, Pablo Andretta Jaskowiak, 2019.
132 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Campus Joinville, Programa de Pós-Graduação em Engenharia de Sistemas Eletrônicos, Joinville, 2019.

Inclui referências.

1. Engenharia de Sistemas Eletrônicos. 2. Detecção de falhas. 3. Redes neurais artificiais. 4. Veículos subaquáticos não-tripulados. I. Fröhlich, Antônio Augusto. II. Jaskowiak, Pablo Andretta. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Sistemas Eletrônicos. IV. Título.

Felipe de Pontes Adachi

Aplicação de Técnicas de Aprendizado de Máquina para Detecção de Falhas em Veículos Subaquáticos Não-tripulados

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Mauro Roisenberg, Dr.
Universidade Federal de Santa Catarina

Prof. Thiago Ferreira Covões, Dr.
Universidade Federal do ABC

Prof. Giovani Gracioli , Dr.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Engenharia de Sistemas Eletrônicos.

_____

Prof. Alexandro Garro Brito, Dr.
Coordenador do Programa

_____

Prof. Antônio Augusto Fröhlich, Dr.
Orientador

Joinville, 8 de Outubro de 2019.

AGRADECIMENTOS

À minha família, por todo o apoio e carinho, e em especial à minha tia Eloisa, meus irmãos Rodrigo, Camila e Murilo e meus pais, Eliana e Edson. Ao apoio, amor e valores dados pelos meus pais, devo tudo o que tenho e sou.

À minha esposa, Débora, por ter me ajudado, compreendido e acompanhado, em todos os aspectos da vida. Não imagino o caminho, já trilhado e o por vir, sem a sua companhia.

Aos meus amigos, de Marechal Rondon, São Carlos, Joinville e espalhados mundo afora: meus momentos de repouso, alegria e descontração.

Aos meus orientadores, Prof. Dr. Antônio Augusto Fröhlich e Prof. Dr. Pablo Andretta Jaskowiak. A eles atribuo a motivação para meu crescimento pessoal, espiritual e profissional durante este período.

A todos os professores, mestrandos e graduandos do LISHA que de alguma forma participaram desta minha caminhada, pelas conversas, risadas e sugestões.

Aos membros da minha banca do exame de qualificação, Prof. Dr. Giovani Gracioli e Prof. Dr. Gustavo Medeiros de Araújo, pelas valiosas sugestões e motivação para realizar um trabalho cada vez melhor.

Ao Eros e à Hope, pelo carinho e companhia.

À CAPES, pelo apoio financeiro que permitiu a realização deste trabalho.

“Fabiano dava-se bem com a ignorância.
Tinha o direito de saber? Tinha? Não tinha.
— Está aí.
Se aprendesse qualquer coisa, necessitaria aprender mais, e nunca ficaria satisfeito.”
(Vidas Secas - Graciliano Ramos)

RESUMO

A demanda por robustez e segurança na operação de veículos não tripulados aumenta à medida em que estes veículos são utilizados em operações cada vez mais críticas. Em ambientes subaquáticos, caracterizados frequentemente por sua natureza hostil e de difícil acesso, a preocupação pelo desenvolvimento de sistemas tolerantes a falhas torna-se evidente. Este trabalho propõe uma estratégia para detecção de falhas por meio da obtenção de um modelo dinâmico do comportamento normal do veículo e subsequente comparação do valor previsto com a informação real fornecida pelo veículo. O modelo preditivo é obtido treinando-se uma rede neural artificial para a representação de um modelo Não-linear Autoregressivo com Entradas Exógenas (*Nonlinear Autoregressive with Exogenous Inputs*-NARX). A estratégia proposta é aplicada a um estudo de caso real, utilizando um veículo remotamente operado de baixo custo da empresa OpenROV. Em uma etapa de pré-processamento, relações dinâmicas entre os atributos disponíveis são avaliadas por meio de análises de autocorrelação e Informação Mútua. Em seguida, diferentes métodos de seleção de atributos são aplicados - RReliefF, Seleção de Atributos Baseado em Correlação (*Correlation-based Feature Selection*-CFS), *Stepwise Regression* e sem seleção de atributos. Resultados do desempenho para cada modelo resultante são comparados e avaliados, com relação ao erro quadrático médio dos modelos de regressão. A capacidade de detecção da estratégia também é avaliada através de medidas de precisão, recall e latência de detecção, mediante a inserção de falhas intermitentes de sensor de diversos tipos e magnitudes. Os resultados mostram que, de maneira geral, o método RReliefF para a seleção de atributos obteve desempenho para a presente aplicação. A estratégia proposta apresentou uma alta taxa de detecção para diversos tipos de falha de sensor, apesar de se mostrar insensível a determinadas falhas de baixa magnitude e curta duração. Além disto, após etapas de treinamento e validação, o detector na sua versão final foi integrado em um *framework* baseado em Internet das Coisas e computação em nuvem, contribuindo para a flexibilidade e escalabilidade da solução proposta.

Palavras-chave: Detecção e diagnóstico de falhas. Veículos subaquáticos não tripulados. Redes neurais artificiais. Seleção de atributos.

ABSTRACT

The need for reliability and safety on the operation of unmanned vehicles increases in conjunction with the growing criticality of operations undertaken by these vehicles. In underwater environments, distinguished by its hostile environment, the importance of fault-tolerant systems design is straightforward. This paper proposes a strategy for fault detection by obtaining the dynamic model of the vehicle's normal behavior and comparing the predicted output to the actual information provided by the vehicle. The predictive model is obtained by training a Nonlinear Autoregressive with Exogenous Inputs (NARX) neural network. The proposed strategy is applied to a real case study, by gathering data of the OpenROV, a low-cost remotely operated vehicle (ROV). In a preprocessing phase, dynamic relationships between available features are assessed through auto and cross-correlation analysis. Thereafter, different feature selection procedures are applied- ReliefF, Correlation-Based Feature Selection, Stepwise Regression and no feature selection. Performance results for each resulting model are compared and assessed, regarding mean squared error of regression models. Detection performance results are also compared through precision, recall and latency measures, when the strategy is applied to several different intermittent sensor faults, from incipient to abrupt nature. Results have shown that, in overall, the ReliefF method for feature selection yielded better performance. The proposed strategy proved to be able to detect several types of faults, albeit being insensitive to certain faults of low magnitude and short duration. In addition, the proposed strategy was succesfully integrated into a framework based on Internet of Things and Big Data. Latency measurements denoted the feasibility of the implemented scheme, allowing a prompt response to the ROV pilot, contributing to the flexibility and scalability of the proposed strategy.

Keywords: Fault detection and diagnostics. Underwater unmanned vehicles. Artificial Neural networks. Feature Selection.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
AUV	*Autonomous Underwater Vehicle*
BNP	*Bayesian Nonparametric*
CFS	*Correlation Based Feature Selection*
DDF	Detecção e Diagnóstico de Falhas
EKF	*Extended Kalman Filter*
ESC	*Electronic Speed Control*
FDP	Função de Densidade de Probabilidade
FN	Fator de Novidade
FTA	*Fault Tree Analysis*
GDL	*Grau De Liberdade*
IMU	*Inertial Measurement Unit*
I2C	*Inter-Integrated Circuit*
IoT	*Internet of Things*
JSON	*JavaScript Object Notation*
KDE	*Kernel Density Estimator*
KF	*Kalman Filter*
LDA	*Latent Dirichlet Allocation*
LISHA	Laboratório de Integração Software/Hardware
MISO	*Multiple Input-Single Output*
MLP	*Multilayer Perceptron*
MSE	*Mean Square Error*
NARX	*Nonlinear Autoregressive with Exogenous Inputs*
NLPCA	*Nonlinear Principal Component Analysis*
OCROV	*Observation Class ROV*

PCA	*Principal Component Analysis*
PWM	*Pulse Width Modulation*
RBF	*Radial Basis Function*
ReLU	*Rectified Linear Unit*
RFE	*Recursive Feature Elimination*
RNA	*Rede Neural Artificial*
RNN	*Recurrent Neural Network*
ROV	*Remotely Operated Vehicle*
SPI	*Serial Peripheral Interface*
SVDD	*Support Vector Domain Description*
SVM	*Support Vector Machine*
UIO	*Unknown Input Oberser*
UUV	*Unmanned Underwater Vehicle*
VIF	*Variance Inflation Factor*

SUMÁRIO

1 INTRODUÇÃO

Considerando que o meio subaquático é caracterizado por ser um ambiente altamente hostil e de difícil acesso ao ser humano, a utilização de veículos subaquáticos não-tripulados (*Unmanned Underwater Vehicle*-UUV) para a realização de atividades submarinas tem ganhado grande atenção (XIANG; YU; ZHANG, 2017). Neste sentido, a exploração subaquática vem se beneficiando do uso de veículos não-tripulados em uma diversidade de aplicações (ANTONELLI, 2018). No setor militar, veículos subaquáticos são utilizados, por exemplo, em missões de vigilância, varredura de minas e de busca e resgate (CHEN; ZHU, 2018). A extração de recursos naturais marinhos também utiliza veículos não-tripulados na localização, obtenção, avaliação e gerenciamento de combustíveis fósseis, minérios, peixes e frutos-do-mar, por exemplo (MACREADIE *et al.*, 2018). Outras importantes áreas de aplicação que podem ser mencionadas são a pesquisa oceanográfica (CHOYEKH *et al.*, 2015), monitoramento ambiental (BAYAT *et al.*, 2017), inspeção de tubulações e cabos subaquáticos (XIANG; YU; NIU *et al.*, 2016), intervenção subaquática (ZHANG, F. *et al.*, 2015) e manutenção industrial (MAI *et al.*, 2017).

Embora o uso de UUVs seja motivado pela prevenção de riscos à vida humana inerentes a missões tripuladas, a ocorrência de falhas durante a operação deste veículo pode resultar na interrupção da missão ou perda permanente do equipamento. Como exemplo, pode-se citar a perda do veículo subaquático Nereus durante uma expedição de pesquisa de organismos e sedimentos do fundo do mar em águas profundas (SHOWSTACK, 2014). Portanto, a detecção apropriada da ocorrência de falhas é uma etapa importante na implementação de uma estratégia tolerante a falhas, possibilitando o término da missão ou, então, a recuperação do veículo (ANTONELLI, 2003). No caso de veículos subaquáticos operados remotamente (*Remotely Operated Vehicle* - ROVs), em que um operador humano especializado é responsável pelo comando do veículo, uma estratégia de detecção de falhas é importante para auxiliar o processo de tomada de decisão por parte do piloto. Neste contexto, a importância de projetos de sistemas tolerantes a falhas em veículos subaquáticos não-tripulados é clara.

De acordo com Dai e Gao (2013), na automação industrial, o termo Detecção e Diagnóstico de Falhas (DDF) refere-se ao ato de detectar a ocorrência de uma falha da maneira mais rápida possível e identificar a localização e tipo da falha da maneira mais precisa possível. Em métodos de DDF, a fase inicial de detecção consiste na emissão de um alarme quando uma falha é percebida. Já a fase de diagnósticos de falhas trata da determinação da natureza e localização da falha detectada na fase anterior. Os sistemas de DDF, de uma maneira mais ampla, podem ser categorizados em métodos baseados em redundância física (ou de *hardware*) e analítica (DAI; GAO, 2013). Em abordagens baseadas em redundância física, utiliza-se a replicação de componentes

do sistema, como sensores e atuadores, que recebem ambos a mesma entrada. Desta maneira, falhas podem ser detectadas através do desvio das saídas dos componentes duplicados (GOLOMBEK, 2014). A principal desvantagem desta abordagem é o custo decorrente da instalação de componentes adicionais. Além disto, esta solução pode não ser interessante em aplicações em que o espaço físico é um fator limitante (TINÓS, 1999). Por estes motivos, a redundância física é geralmente utilizada em um número limitado de componentes cuja criticidade é considerada alta (GOLOMBEK, 2014).

A tarefa de detecção e diagnóstico de falhas também pode ser abordada com base na redundância analítica, ou seja, nas relações funcionais entre as diversas variáveis do processo monitorado (CHEN; PATTON, 2012). Em outras palavras, é utilizado um modelo matemático do sistema como um todo ou de partes do mesmo (PATTON; FRANK; CLARK, 2013). Assumindo que falhas influenciarão a saída ou estado do sistema, pode-se comparar o valor da saída estimado pelo modelo matemático com o valor observado. Através desta comparação gera-se o chamado resíduo, que pode então ser avaliado por um método subsequente para a classificação do estado do sistema (DING, 2008).

Para a obtenção do modelo a ser utilizado em estratégias de DDF, pode-se utilizar técnicas de modelagem de sistemas, partindo de um conhecimento prévio da dinâmica do processo (VENKATASUBRAMANIAN, V. *et al.*, 2003). No entanto, existem casos em que a complexidade do processo pode dificultar a obtenção de um modelo matemático (ALZGHOUL *et al.*, 2014). Isto é, de fato, verdadeiro para veículos subaquáticos, considerando que estes estão sujeitos a diversas forças de natureza complexa, multivariável e não-linear (GOMES, 2011).

É razoável, portanto, abordar a detecção e diagnóstico de falhas por meio de métodos capazes de extrair o conhecimento implícito presente nos dados históricos, obtidos a partir de sessões passadas de operação.

Ao utilizar métodos de aprendizado de máquina para a função de DDF em veículos subaquáticos, uma característica a ser levada em conta é a dificuldade de obtenção de dados referentes a estados de falha do veículo (ZHANG; WU; CHU, 2014), além da incapacidade de previsão de todos os tipos possíveis de falhas. Neste contexto, a detecção de falhas, etapa inicial de estratégias de DDF, pode ser abordada por métodos baseados em reconstrução, para os quais um modelo de normalidade é construído a partir de dados nominais (sem falhas) do sistema (PIMENTEL *et al.*, 2014). A saída do modelo pode então ser comparada às medições reais disponíveis do processo monitorado, obtendo desta maneira o erro de reconstrução, ou resíduo, que pode ser relacionado a condições de falhas (HU; PALMÉ; FINK, 2017). Espera-se que, desta maneira, comportamentos anômalos do sistema monitorado levarão a altos erros de reconstrução.

Particularmente, o uso de Redes Neurais Artificiais (RNAs) em abordagens

baseadas em reconstrução tem aumentado. RNAs tradicionais do tipo *feedforward* são conhecidas pela sua habilidade de capturar comportamento não-linear. No entanto, sem modificações, este tipo de rede fornece um mapeamento estático, não sendo adequado para a representação de sistemas dinâmicos. De acordo com Haykin *et al.* (2009), existem duas formas de incorporar o componente de tempo na operação de uma RNA:

1. A adoção de uma estrutura de memória na entrada de uma RNA estática.
2. Por meio do uso de realimentação.

Um exemplo de arquitetura que utiliza estruturas de memória são as TDNNs (*Time-delay neural networks*), em que linhas de atraso são associadas a RNAs do tipo *feedforward*. Este tipo de rede é utilizado, por exemplo, no trabalho de Jäger *et al.* (2014), em que falhas de sensores em sistemas industriais são detectadas, e no trabalho de Christensen *et al.* (2008), em que é abordada a detecção de falhas de sensores e atuadores em sistemas robóticos autônomos.

Uma segunda classe de RNA muito utilizada para a representação de sistemas dinâmicos é a rede NARX (*Nonlinear Autoregressive with Exogenous Inputs*). Nesta estrutura, linhas de atraso também são utilizadas, podendo também contar com conexões de realimentação entre a saída e a entrada da rede. Os trabalhos de Capriglione *et al.* (2018) e Nascimento e Valdenegro-Toro (2018), por exemplo, empregam redes NARX para a detecção de falhas em motocicletas e em propulsores subaquáticos, respectivamente.

Uma segunda questão acerca do uso de técnicas de aprendizado de máquina consiste nos atributos que serão utilizados para a análise. Em abordagens de aprendizado de máquina aplicado a veículos não-tripulados, frequentemente diversos tipos de dados estão disponíveis, como informações de sensores internos e externos, bem como de atuadores e outros componentes do sistema. Neste contexto, em uma fase de pré-processamento, o processo de seleção de atributos constitui uma questão importante. A inclusão de atributos irrelevantes pode afetar o desempenho do sistema de detecção, além de aumentar o tempo de aprendizado e complexidade do modelo preditivo (KOPRINSKA; RANA; AGELIDIS, 2015). Neste sentido, justifica-se a utilização de métodos de seleção para a determinação de um subconjunto apropriado de atributos. Além disto, para determinados processos dinâmicos, uma seleção apropriada deve também considerar aspectos temporais de relacionamentos entre os atributos; i.e., atributos podem afetar uns aos outros em momentos futuros (KANEKO; FUNATSU, 2012). Nestes casos, diversas técnicas podem ser utilizadas para a obtenção de medidas de similaridade entre variáveis, como nos trabalhos de Sánchez-Fernández *et al.* (2018) e Wunsch, Liesch e Broda (2018).

1.1 OBJETIVOS

O principal objetivo deste trabalho é o desenvolvimento de uma estratégia de detecção de falhas em UUVs com base em técnicas de aprendizado de máquina, por meio da análise do desvio do comportamento do veículo com relação a um modelo de normalidade, quando sujeito a falhas oriundas de defeitos de sensores. Para tal, os seguintes objetivos específicos são definidos:

- Obtenção de um modelo preditivo capaz de representar o comportamento dinâmico e não-linear de UUVs
- Desenvolvimento de um processo de tomada de decisão, a partir dos resultados obtidos pelo modelo preditivo
- Aplicação e validação da estratégia em um estudo de caso real
- Avaliação do desempenho da estratégia aplicada ao estudo de caso, por meio de métricas de capacidade e latência de detecção de falhas da estratégia
- Implementação da estratégia em sua etapa de operação, após obtenção dos modelos e definição dos parâmetros

1.2 CONTRIBUIÇÕES

A contribuição obtida com este trabalho consiste na implementação e avaliação de uma estratégia de detecção de falhas aplicada a veículos subaquáticos não-tripulados, cujo resultado possa auxiliar no processo de tomada de decisão, tanto de sistemas inteligentes tolerantes a falha ou por parte de pilotos humanos, em caso de veículos operados remotamente. A estratégia foi desenvolvida considerando limitações comumente encontradas em casos práticos, como a dificuldade de modelagem de sistemas dinâmicos complexos, a dificuldade de obtenção de dados referentes a estados de falha e a dificuldade em prever as possíveis condições de falhas.

1.3 METODOLOGIA

O modelo preditivo utilizado consiste em uma RNA para a representação de um modelo NARX. Em uma etapa de pré-processamento, relações dinâmicas entre atributos são avaliadas utilizando análises de autocorrelação e Informação Mútua e, posteriormente, diferentes métodos de seleção de atributos são aplicados e avaliados: RReliefF (RF) (KONONENKO, 1994), método baseado em correlação (*Correlation-based feature selection* - CFS) (HALL, 1999), Stepwise regression (SW) (KLEINBAUM *et al.*, 1988) e sem seleção de atributos.

A estratégia proposta é aplicada em um estudo de caso real, utilizando um veículo subaquático da plataforma OpenROV V2.8 (STACKPOLE; LANG, 2013), um veículo operado remotamente de baixo custo. A base de dados utilizada para o treinamento do modelo foi obtida a partir da coleta de leituras de sensores realizada ao longo de diversos experimentos em campo. A eficácia da estratégia proposta é avaliada utilizando um conjunto de dados de teste, mediante injeção de falhas de sensores de diferentes tipos, magnitudes e durações. Os resultados obtidos demonstram que a estratégia proposta foi capaz de detectar determinados tipos de falhas, mantendo uma baixa taxa de falsos positivos. No entanto, a estratégia não foi capaz de detectar determinados tipos de falhas, geralmente de baixa magnitude e curta duração. Os valores de latência encontrados também demonstram que, em média, as falhas são detectadas enquanto ainda estão ativas. Com relação ao método de seleção de atributos, o método RReliefF, de maneira geral, levou a melhores resultados, tanto para a capacidade de regressão do modelo quanto para o desempenho de detecção de estratégia como um todo.

Finalmente, concluída a fase de treinamento e validação, a estratégia proposta é integrada em um *framework* baseado em computação em nuvem. Desta maneira, as informações sensoriais do ROV são enviadas ao servidor na nuvem, onde o processo de detecção ocorre. O diagnóstico é então armazenado em uma base de dados, que está disponível para visualização ao piloto do ROV. Para tal, foi escolhida a plataforma de sensoriamento distribuído do LISHA (Laboratório de Integração Software/Hardware - UFSC).

1.4 ORGANIZAÇÃO DO TEXTO

O restante deste trabalho é organizado da seguinte maneira.

No Capítulo 2 serão descritos os ROVs, que são a aplicação-alvo do sistema de detecção proposto, seguido de uma breve explicação acerca de sua dinâmica e cinemática. Por fim, um detalhamento maior será dado ao modelo de ROV escolhido para a realização deste projeto - o OpenROV V2.8, com relação ao controle e manobrabilidade, hardware, software, sistema de coleta de dados e módulos adicionais.

No Capítulo 3 será abordado o tópico da detecção e diagnóstico de falhas. Serão introduzidos conceitos necessários ao desenvolvimento e entendimento do projeto, como a definição de falhas e seus tipos, etapas que compõem um sistema DDF e diferentes categorias de sistemas de DDF. Ao longo do capítulo, serão apresentados e discutidos estudos relacionados à tarefa de DDF aplicada em diferentes áreas, com maior foco a estudos aplicados a UUVs.

No Capítulo 4 são apresentados conceitos relacionados às principais técnicas estatísticas e de aprendizado de máquina utilizadas ao longo de diversas etapas deste trabalho, como exploração dos dados, pré-processamento e extração de conhecimento.

No Capítulo 5 a estrutura da estratégia de detecção de falhas proposta será explicada, compreendendo todas as etapas necessárias para a sua implementação e integração com o ROV.

No Capítulo 6 serão apresentados os procedimentos e resultados obtidos ao aplicar a estratégia proposta ao veículo OpenROV, iniciando pelas etapas preliminares de modificação do ROV, processo de coleta de dados e etapas de pré-processamento. A seguir, os resultados das etapas intermediárias serão discutidos, como as análises de correlação, aplicação dos métodos de seleção de atributos e validação dos modelos preditivos. Por fim, a eficácia da estratégia, com relação à capacidade de detecção de falhas, será discutida.

No Capítulo 7, tendo concluído as etapas de avaliação da estratégia, a integração do detector com a plataforma IoT será contextualizada, explicada e demonstrada.

No Capítulo 8 as considerações finais são apresentadas, juntamente com propostas para trabalhos futuros.

2 VEÍCULOS OPERADOS REMOTAMENTE

De acordo com Christ e Wernli Sr (2013), veículos subaquáticos podem ser classificados entre veículos tripulados e não-tripulados, sendo ainda subdivididos em categorias subsequentes, como pode ser visto na Figura 1.

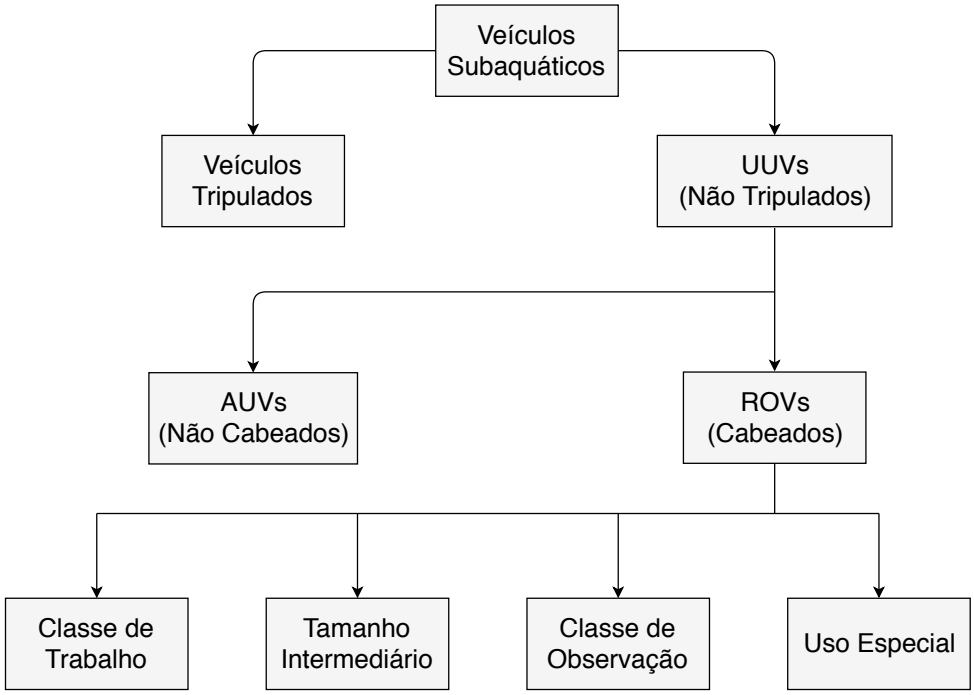


Figura 1 – Classes de veículos subaquáticos.

De acordo com a marinha americana (NAVY, 2004), um veículo subaquático não-tripulado (UUV - *Unmanned underwater vehicle*) é definido como:

> "Submersível autopropelido cuja operação seja completamente autônoma (pré-programada ou controle de missão adaptativa em tempo real) ou sob controle supervisório mínimo e não-cabeado exceto, possivelmente, por enlace de dados como um cabo de fibra óptica."

Já a distinção entre veículos subaquáticos autônomos (AUVs - *Autonomous Underwater Vehicles*) e veículos operados remotamente (ROVs - *Remotely Operated Vehicles*), de acordo com Christ e Wernli Sr (2013), se dá principalmente pela presença ou não de um cabo, denominado de cabo umbilical, utilizado para transmissão de sinais e/ou alimentação. Um AUV não o possui e opera de forma autônoma, sem a necessidade de comandos de entrada de um operador, embora possa ter habilidades de comunicação com a superfície por meio de um modem acústico ou, quando em superfície, via radiofrequência ou comunicação óptica.

Já o ROV (Figura 2) é conectado via um cabo umbilical a um console localizado na superfície, e pode ter um comportamento autônomo ou controlado remotamente,

dependendo da capacidade do veículo e quantidade de comandos enviados pelo operador (CHRIST; WERNLI SR, 2013). Utilizando o console, o operador pode inserir comandos para movimentar o veículo e também receber feedback dos sensores embarcados, como câmera digital, profundidade, orientação, dentre outros.

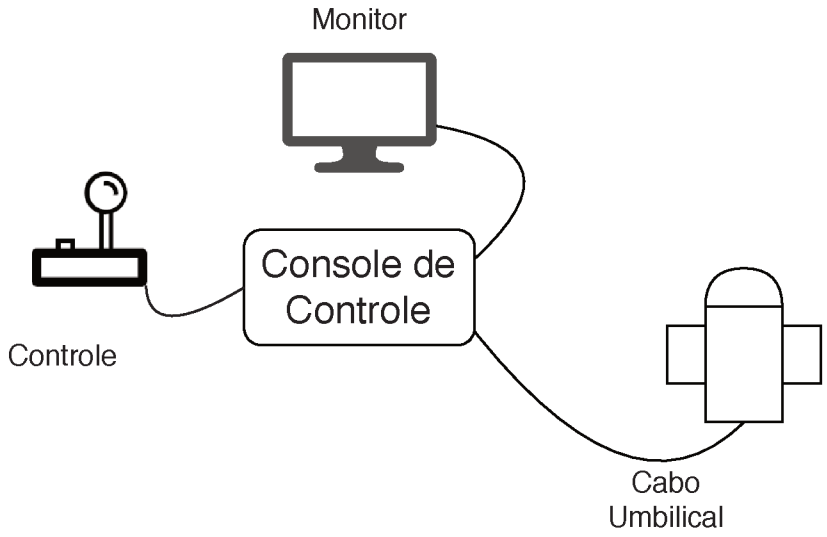


Figura 2 – Componentes básicos de um ROV.

Ainda de acordo com Christ e Wernli Sr (2013), os ROVs existentes podem se situar em diferentes classes, já que apresentam especificações e funcionalidades diversas conforme a aplicação para a qual foram projetados. Os veículos integrantes da classe mais simples, denominada de *Observation Class ROV* (OCROV), são geralmente utilizados para fins de inspeção em águas rasas de até 300 metros de profundidade, apresentam capacidade de intervenção limitada e usualmente apresentam um baixo custo. À medida que a profundidade de operação aumenta e a natureza da missão se torna mais complexa, ROVs podem aumentar muito de tamanho e peso, assumir configurações mais robustas, e contar com funcionalidades e ferramentas adicionais, como braços manipuladores. As maiores diferenças entre classes de ROVs se dá pela especificação de profundidade e transmissão de energia. ROVs mais simples podem ser alimentados com tensão contínua, por baterias embarcadas no próprio veículo, deixando o cabo apenas para transmissão de sinais. Já ROVs maiores e mais sofisticados geralmente são alimentados através do cabo, em corrente alternada.

Considerando que existem veículos não-cabeados, é importante ressaltar o motivo da presença do cabo umbilical em ROVs. Diferente do meio aéreo, ondas eletromagnéticas viajam a curtas distâncias na água, devido a altos efeitos de atenuação e absorção do ambiente subaquático. Sabe-se que a absorção de energia eletromagnética no ambiente marítimo é de cerca de $45 \times f$ dB por quilômetro, onde f é a frequência em Hertz (JIANG, 2008). Sistemas ópticos também sofrem de limitações de distância,

devido a efeitos de retroespalhamento e absorção do meio subaquático (PARTAN; KUROSE; LEVINE, 2007), necessitando de ambientes com água muito límpida. De acordo com Christ e Wernli Sr (2013), a maneira mais utilizada de comunicação subaquática é a acústica, cujo limite de taxa de transmissão (em 2013) é de cerca de 100kbps. Além disto, a velocidade de transmissão de sinais acústicos em água salgada é de cerca de 1500 m/s (URICK, 1967). Esta baixa velocidade do sinal no meio subaquático também introduz limitações na latência da comunicação. Por sua vez, a necessidade de um feedback de vídeo na maioria das tarefas realizadas por um ROV impõe a necessidade de uma alta taxa de transmissão de dados e baixa latência, que não podem ser fornecidos via comunicação acústica. Portanto, com a tecnologia atual, faz-se necessário, em muitos casos, um meio de comunicação cabeado. Este é o caso do veículo empregado no presente trabalho.

2.1 DINÂMICA E CINEMÁTICA

Um ROV, por estar sujeito a diversas forças de natureza complexa, multivariável e não-linear, impõe grande dificuldade no equacionamento e definição de parâmetros envolvidos na obtenção de um modelo dinâmico correspondente (GOMES, 2011). Por exemplo, podemos citar as forças exercidas pelo arrasto hidrodinâmico, propulsores e força de sustentação (VERVOORT, 2008). A força de tração causada pelo cabo umbilical que liga o ROV ao console também é um fator relevante, sendo maior a sua influência de acordo com a incidência da corrente oceânica (GOMES, 2011). Ademais, a modificação física do ROV, como inclusão, remoção ou substituição de componentes do veículo exigiriam um esforço de reparametrização e remodelagem, visto que a dinâmica do veículo é altamente dependente de forma e massa. Neste sentido, justifica-se a utilização de um método de obtenção de conhecimento a partir de dados históricos do veículo que seja capaz de refletir o seu comportamento, levando em consideração todas as suas particularidades.

Em razão do discorrido, a modelagem da dinâmica de um veículo subaquático não está no escopo deste trabalho. Esta seção abordará a cinemática do ROV, utilizando as convenções adotadas por Vervoort (2008), com o propósito de definir e explicar os termos utilizados posteriormente.

2.1.1 Cinemática

Para o controle e posicionamento de um ROV, parâmetros como posição, velocidade e orientação relativos a um referencial inercial são muito importantes. No entanto, toda a dinâmica do veículo é modelada com base em um referencial fixo ao veículo. Desta maneira, realiza-se a adoção de dois referenciais, um fixo ao planeta Terra, e outro fixo no próprio veículo, como visto na Figura 3. Convenciona-se que o eixo X e

Y do referencial inercial apontam para o norte e leste, respectivamente, enquanto que o eixo Z aponta para o centro da Terra. Já o referencial do corpo tem os eixos X_0 e Y_0 apontando para a parte frontal e lateral do ROV, respectivamente, enquanto que o eixo Z_0 aponta para a parte de baixo do veículo.

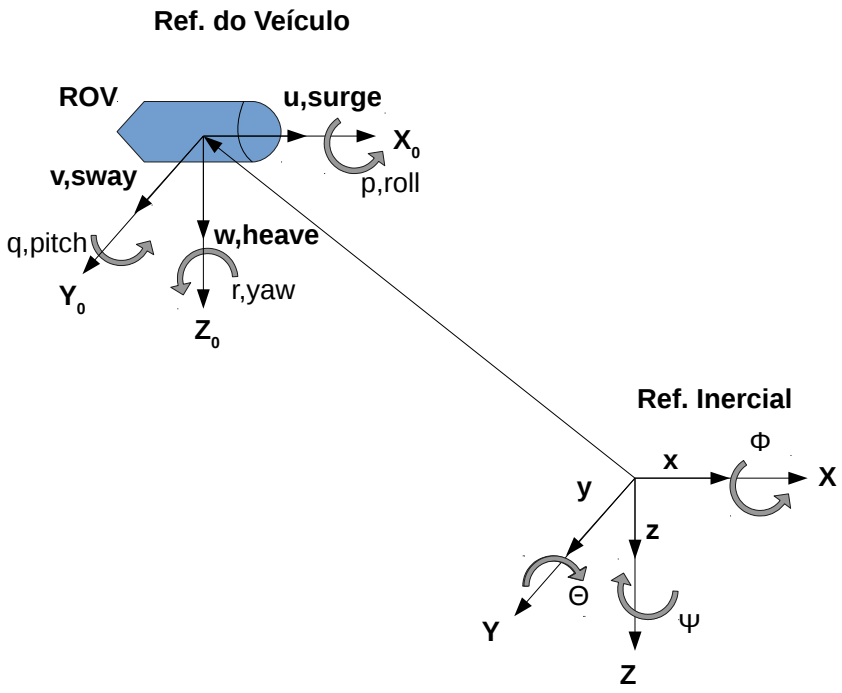


Figura 3 – Referenciais: Inerciais e do Corpo.

A movimentação do ROV no ambiente subaquático não é restrita em nenhum grau de liberdade (GDL), portanto todos os 6 GDLs do referencial do corpo são de relevância: os 3 graus de translação e 3 graus de rotação nos eixos X_0, Y_0, Z_0. Os movimentos lineares nos eixos longitudinal, transversal e normal X_0, Y_0, Z_0 são denominados de movimentos de avanço (*surge*), balanço (*sway*) e oscilação (*heave*), respectivamente. Já os movimentos de rotação nos eixos X_0, Y_0, Z_0 são denominados de movimentos de rolamento (*roll*), arfagem (*pitch*) e de guinada (*yaw*). Já com relação à velocidade, os termos *u, v, w* e *p, q, r* são utilizados para definir as velocidades lineares e de rotação dos eixos X_0, Y_0 e Z_0, respectivamente.

2.1.1.1 Ângulos de Euler

Para se determinar a orientação do ROV em relação a um referencial inercial, é necessário relacionar um sistema de coordenadas com relação a outro, expressando a orientação do referencial móvel de acordo com o referencial fixo. Estes ângulos, denominados de ângulos de Euler (DIEBEL, 2006), são obtidos através de uma sequência

de três rotações em torno dos 3 eixos, convencionado na ordem z-y-x. Desta maneira, o referencial móvel é primeiramente rotacionado com relação ao eixo z, seguido por rotações em torno do eixo y e, então, do eixo x, correspondendo aos ângulos de rotação de yaw, pitch e roll do ROV.

2.2 A PLATAFORMA OPENROV

O OpenROV V2.8 (Figura 4) é um submarino telerobótico desenvolvido pela empresa OpenROV (STACKPOLE; LANG, 2013), com o objetivo de tornar mais acessível a exploração subaquática. A plataforma foi escolhida para este projeto devido ao seu baixo custo, quando comparado com outros ROVs comerciais existentes. Além disto, o fato de ser uma plataforma aberta facilita eventuais desenvolvimentos realizados ao veículo.

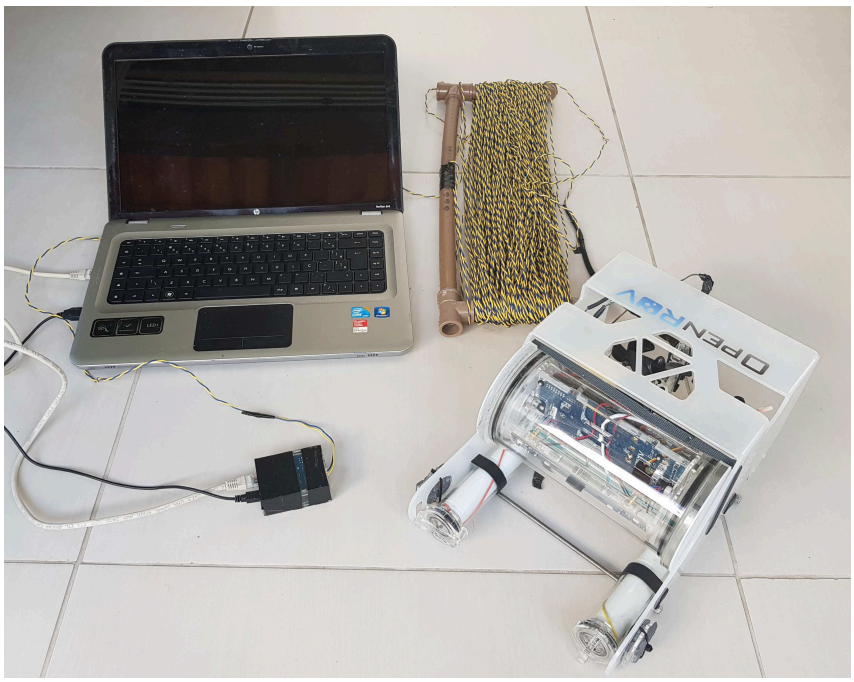


Figura 4 – O OpenROV.

Tendo em mente eventuais modificações futuras pela comunidade OpenROV, o veículo dispõe de um barramento externo para comunicação com dispositivos adicionais utilizando protocolo de comunicação I2C (SEMICONDUCTORS, 2000). O funcionamento interno do ROV é baseado no microcontrolador Arduino ATMega (ARDUINO, 2018), responsável pelo interfaceamento dos sensores e atuadores utilizados no veículo, e na plataforma BeagleBone Black (COLEY, 2013), que implementa a interface de comando e processamento de imagem do ROV. A interface com o usuário é denominada de cockpit e é implementada para uso em um notebook padrão, através de navegadores como o Google Chrome. Através do teclado do notebook ou um gamepad, o usuário pode inserir comandos para movimentar o veículo em níveis diferentes de

Tabela 1 – Lista de componentes de hardware do OpenROV V2.8

Componente	**Quantidade**	**Empresa**	**Informações Adicionais**
Placa de Controle OpenROV V2.8	1	OpenROV	Atmel AtMega2560 embarcado
BeagleBone Black	1	BeagleBoard.org	
Motor DC Brushless	3	Turnigy DST-700	Modelo DST-700
Controlador de Velocidade - ESC	3	Afro	12 A
Propulsor Bombordo	1	Graupner	Modelo 2308.60L
Propulsor Boreste	1	Graupner	Modelo 2308.60
Propulsor Vertical	1	Graupner	Modelo 2303.57
Baterias Recarregáveis	6	AAPC	Li-FePO4, 3.2v, 3300mAh
Adaptador PowerLine (Par)	1	Tenda	200 Mbps
Placa de Interface Topside	1	OpenROV	

Tabela 2 – Especificações principais do OpenROV V2.8

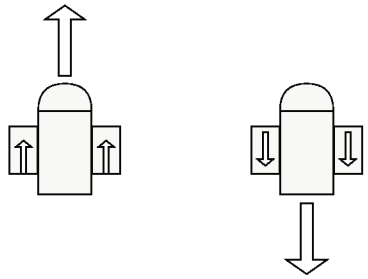
Dimensões	30 cm x 20 cm x 15 cm
Massa	2.6 kg
Profundidade Máxima	100 m
Velocidade Máxima	3,7 km/h (2 nós)

velocidade, ligar ou desligar os LEDs embarcados, alterar a inclinação da câmera e ligar os lasers de medição. O feedback dos sensores embarcados é realizado visualmente através do monitor do notebook. As listas dos componentes de hardware e especificações principais podem ser vistas nas Tabelas 1 e 2, respectivamente.

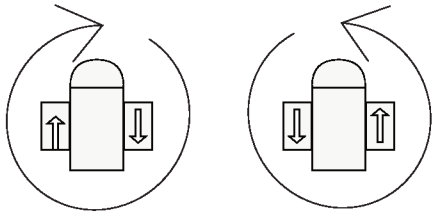
2.2.1 Controle e Manobrabilidade

A propulsão do OpenROV é realizada por um conjunto de 3 hélices acopladas por um eixo a motores elétricos *brushless* de tensão contínua, que por sua vez são acionados por controladores digitais, ou ESCs (*Electronic Speed Control*). Desta maneira, a velocidade de rotação de cada uma das hélices é controlada por meio de um sinal de PWM (*Pulse Width Modulation*) enviado pelo microcontrolador. A velocidade do veículo pode ser estabelecida pelo piloto através do ajuste da variável denominada fator de propulsão (*thrust factor*) em 6 níveis distintos, sendo o fator 1 equivalente à velocidade mais baixa e 6 a velocidade mais alta.

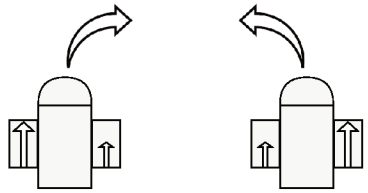
O controle de profundidade é realizado por meio de um propulsor disposto ao longo do eixo Z_0, enquanto que dois propulsores são dispostos na laterais do veículo para a movimentação no plano horizontal X_0-Y_0. Apesar da simples configuração, a combinação dos dois propulsores laterais permite uma variedade de movimentos ao veículo. Na Figura 5, as setas internas representam a força de propulsão causada pelos propulsores ao ROV, enquanto que as setas externas representam o movimento efetivo do veículo. Como visto, de acordo com a intensidade e direção das forças propulsivas, o veículo pode se movimentar à vante e a ré, bem como giros à esquerda e direita, quando as forças em sentidos opostos geram um momento ao redor do centro do ROV. Uma combinação dos dois movimentos também é possível, fazendo com que o ROV



(a) Movimentos de vante e ré do ROV.



(b) Movimentos de giro à direita e esquerda do ROV.



(c) Movimentos de curvas à direita e esquerda do ROV.

Figura 5 – Movimentos possíveis no plano horizontal do ROV com 2 propulsores laterais.

faça curvas à esquerda e à direita, com ângulos que podem variar, de acordo com o valor do sinal PWM enviado a cada um dos motores.

Os 3 propulsores em conjunto, portanto, habilitam o controle do ROV em 3 graus de liberdade: movimentos lineares de oscilação (heave) e de avanço (surge) e movimento de rotação de guinada (yaw). Visto que o número de graus controlados é inferior ao número de graus de liberdade, o OpenROV é definido como um veículo sub-atuado (SPONG, 1998).

2.2.2 Estrutura dos Dados Coletados

O cockpit disponibiliza ao piloto o recurso de gravação de dados embarcados a uma frequência predeterminada. Após o encerramento da sessão de gravação, um arquivo no formato JSON (JavaScript Object Notation) é obtido a partir do navegador utilizado, com dados separados em duas classes: dados de navegação e dados de telemetria. Os dados de navegação compreendem atributos relacionados à orientação do OpenROV no espaço, como roll, pitch, yaw e profundidade. Já os dados de

telemetria estão relacionados ao estado interno do ROV, como corrente dos grupos de bateria 1 e 2, temperatura, tensão e corrente da placa de controle, corrente dos ESCs 1,2 e 3, porcentagem de uso de CPU, comandos enviados aos motores, estados de inicialização de periféricos, dentre vários outros. Um exemplo de dados coletados, tanto de navegação quanto de telemetria, pode ser visto no Anexo A na página 129.

Tanto os dados de navegação quanto de telemetria possuem a mesma estrutura: são vetores de objetos, sendo que cada objeto é uma coleção de atributos, cujos valores correspondem a uma coordenada temporal específica. O tempo da coleta da amostra é definido pelo timestamp contido como um atributo em cada objeto do vetor. No Anexo A, apenas um objeto de cada classe de dados é ilustrado. O timestamp é representado no sistema Unix Epoch, com resolução de um milissegundo. A frequência de amostragem varia de acordo com a categoria dos dados, sendo que para os dados de navegação o período entre duas amostragens é de 100 ms (10 Hz) e para os dados de telemetria o período é de 1s (1Hz). Durante a realização deste trabalho, as frequências originais de amostragem foram alteradas. O procedimento de alteração será explicado em maiores detalhes em seções futuras.

2.2.3 O Módulo de Medição Inercial/Profundidade

Os sensores responsáveis por fornecer os dados de navegação fazem parte de um módulo de IMU (*Inertial Measurement Unit*) / Profundidade, e não fazem parte do hardware básico do ROV. Para que estas leituras sejam possíveis, deve-se comprar o módulo à parte ou, como se trada de uma tecnologia de código aberto, fabricá-lo a partir dos esquemáticos disponíveis e componentes avulsos. Com o módulo construído, sua integração com o ROV é realizada através do barramento I2C disponível para a comunicação com dispositivos externos. O módulo é composto por 2 componentes: O System in Package (SiP) BNO055 (SENSORTEC, 2014), e o sensor de pressão MS5837-30BA (CONNECTIVITY, 2012). O BNO055 implementa a funcionalidade de medição inercial, integrando em um único encapsulamento um acelerômetro triaxial de 14 bits, um giroscópio triaxial de 16 bits, um sensor geomagnético triaxial e um microcontrolador de 32 bits cortex M0+. O BNO055 é utilizado pelo OpenROV para obter os valores de roll, pitch e yaw apresentados no cockpit, para auxiliar a navegação do veículo.

Já o MS5837-30BA é um sensor de pressão de 24 bits de silício piezoresistivo desenvolvido para sistemas de medição de profundidade, com resolução de profundidade de água de 2mm. O sensor pode operar em pressões no intervalo de 0 a 30 bar e conta com um sensor de temperatura para compensação, cujo valor é apresentado no cockpit, assim como o valor de profundidade.

Espera-se que a inclusão de atributos relativos à navegação do ROV auxilie na tarefa de detecção de falhas e, por este motivo, o módulo mencionado foi fabricado e

integrado ao ROV. Mais detalhes acerca da integração serão fornecidos em seções futuras deste trabalho.

3 DETECÇÃO E DIAGNÓSTICO DE FALHAS

Os trabalhos que compõem a literatura de DDF utilizam definições e conceitos que podem variar em nomenclatura dependendo da área ou comunidade de pesquisa. Na área de engenharia e sistemas de controle, utiliza-se, usualmente, os termos *falha* e *defeito*. Já na área de ciências da computação, o termo *erro* também é introduzido e definido e, embora os termos *falha* e *defeito* também sejam utilizados, os significados atribuídos a eles variam. Com relação à aplicação, os trabalhos de engenharia geralmente associam tarefas de DDF a sistemas industriais (SARTORI *et al.*, 2012), enquanto que trabalhos na área de ciências da computação associam a sistemas de computação e comunicação (AVIZIENIS *et al.*, 2004). Desta maneira, a literatura de DDF em sistemas robóticos, por conta da sua multidisciplinaridade, utiliza frequentemente os dois conjuntos de definições mencionados. Neste capítulo, será feita uma revisão acerca da tarefa de DDF, com enfoque na aplicação em ROVs e AUVs, tema deste trabalho.

No domínio da engenharia, de acordo com Himmelblau (1978) e Witczak (2007), uma falha pode ser definida como qualquer desvio não aceitável de uma variável observada ou parâmetro calculado associado a um processo. Sob esta perspectiva, a falha é definida como um sintoma ou anormalidade do processo. Já o termo defeito é definido como a causa subjacente da referida anormalidade (VENKATASUBRAMANIAN, V. *et al.*, 2003). Como exemplo, a presença de água indicada por algum sensor em um compartimento estanque (impermeável a líquido) pode ser vista como uma falha, enquanto que o vazamento causado por um furo pode ser visto como o defeito causador da falha.

Ainda de acordo com V. Venkatasubramanian *et al.* (2003), de um modo geral, defeitos podem ser definidos em 3 classes:

1. Mudanças de parâmetros em um modelo: surgem devido a um distúrbio no processo causado por uma ou mais variáveis exógenas. Como exemplo, pode-se citar fenômenos de multipath, em que o sinal viaja do emissor ao receptor por mais de um caminho, podendo causar interferências no sistema de ecolocalização de um AUV.

2. Mudanças Estruturais: Referem-se a mudanças no processo em si. Um exemplo seria uma rachadura na estrutura de um ROV, podendo causar a entrada de água no compartimento estanque.

3. Defeitos de sensores e atuadores: efeitos de histerese ou curto-circuito em sensores e desgaste do rolamento de motores são exemplos desta classe de defeitos.

Já no contexto de computação tolerante a falhas, segundo Johnson (1996) e Avizienis *et al.* (2004), o termo *fault* está associado ao universo físico, carregando

um significado similar ao termo *failure*, como definido em V. Venkatasubramanian *et al.* (2003). Por este motivo, neste trabalho ambos os termos serão traduzidos como *defeito*, como definido anteriormente.

Segundo Avizienis *et al.* (2004), defeitos podem ser categorizados quanto a:

1. Fase: Podem ocorrer na fase de desenvolvimento do sistema ou na fase de uso.
2. Dimensão: Falhas originadas em hardware ou software.
3. Fronteira: Falhas originadas dentro da fronteira do sistema ou externamente.

Com relação à tarefa de detecção e diagnóstico de falha (DDF), três subetapas principais estão envolvidas (CHEN; PATTON, 2012):

1. Detecção: Decisão do estado do sistema entre condição normal / com falha.
2. Isolamento: Determinação do local da falha, e.g., qual sensor ou atuador.
3. Identificação: Determinação do tipo, magnitude ou natureza da falha.

Neste trabalho, a etapa inicial de detecção de falhas será considerada, limitando-se à tarefa de classificação do estado atual do sistema entre duas opções possíveis: normal / com falha. No entanto, como a detecção de falhas é uma etapa integrante da tarefa de DDF e muitos trabalhos da literatura envolvem todas as etapas do processo, a discussão neste capítulo será feita no contexto mais amplo de detecção e diagnóstico de falhas, seguida por considerações sobre a utilização de métodos de aprendizado de máquina para a etapa específica de detecção de falhas.

Para fins de comparação e avaliação das estratégias existentes na literatura, é importante estabelecer um conjunto de características desejáveis que um sistema de DDF deve possuir. A seguir, são apresentadas algumas características pertinentes à etapa de detecção de falhas, como apresentado em V. Venkatasubramanian *et al.* (2003):

- Rapidez da Detecção: Embora a rapidez da detecção de falhas em um sistema seja desejável, verifica-se que é uma característica conflitante com o desempenho durante a operação normal. Um sistema projetado para detectar falhas rapidamente também será sensível a ruídos e influências de alta frequência, levando a uma possível ocorrência frequente de falsos positivos. Portanto, deve-se sempre procurar um balanço entre as duas características para a aplicação em específico.
- Estimativa do erro de classificação: É importante que o sistema de DDF forneça uma estimativa a priori do erro de classificação, para que o usuário possa realizar uma projeção dos níveis de confiança dos diagnósticos obtidos, fornecendo uma base mais sólida para a tomada de decisões futuras.

- Robustez: Um sistema é dito robusto se este mantém o seu desempenho satisfatoriamente na presença de ruídos e incertezas. A adoção de limiares para a classificação pode ter que ser realizada de maneira conservadora na presença de ruídos, implicando em um balanço entre a robustez e a acurácia de um sistema (TIDRIRI *et al.*, 2016).

- Adaptabilidade: Os processos de um modo geral estão sujeitos a mudanças nas condições de operação, ambientais e mudanças estruturais. Um atributo desejável de um sistema de DDF é que este se adapte a tais mudanças, evoluindo à medida que novas informações estejam disponíveis.

- Requisitos de Modelagem: A quantidade de esforço envolvida para a obtenção e disposição do método de DDF também deve ser considerada, não somente para a etapa inicial de desenvolvimento, mas também nas etapas de manutenção. Por exemplo, um sistema de DDF aplicado a uma plataforma robótica deve apresentar facilidade na remodelagem em ocasiões de inclusão de módulos, funcionalidades ou alteração de firmware.

- Requisitos computacionais e de armazenamento: Sistemas de DDF sensíveis a requisitos de latência podem optar por soluções que exijam baixo processamento computacional à custa de uma alta necessidade de armazenamento. É importante que o sistema proposto apresente um balanço entre os dois parâmetros que seja adequado à aplicação específica.

3.1 CATEGORIAS DE SISTEMAS DDF

Dentre as abordagens de DDF existentes, as mais simples consistem em sistemas de alarme, em que sinais são monitorados e comparados a limites pré-definidos para a determinação de falhas (CHEN; PATTON, 2012). Mais de um limite pode ser utilizado, representando um valor máximo e/ou mínimo, podendo apresentar um ou dois níveis (pré-alarme e alarme total) (GERTLER, 2013). Apesar de ser uma estratégia simples, é uma solução que prejudica o isolamento de falhas, uma vez que uma variável que exceda determinado limite é um sintoma que pode ser causado por diversos defeitos. Além disto, os limites adotados devem ser consideravelmente altos, já que o intervalo de um estado “normal” de um processo complexo é dependente das entradas do sistema, o que não é contemplado nesta abordagem. Desta forma, a capacidade de detecção de falhas do sistema é prejudicada. Não obstante, tais sistemas são amplamente utilizados na indústria devido a sua simplicidade e facilidade de implementação (GERTLER, 2013).

Com o aumento da complexidade dos sistemas, faz-se necessário o uso de estratégias de DDF mais rebuscadas. Um ponto em comum de estratégias de DDF é que

todas se valem da redundância de informações do sistema para a inferência de falhas, seja esta redundância na forma física ou analítica (CHOW; WILLSKY, 1984; DAI; GAO, 2013). Na redundância física, componentes de hardware do sistema, como sensores e controladores, são duplicados, e a falha é detectada se a saída do componente em questão é diferente de sua contraparte (um ou mais componentes) redundante. Por meio da redundância física, é possível realizar a detecção de uma maneira rápida e precisa, além de permitir o isolamento de falhas, uma vez que a indicação de falha obtida é relativa a componentes individuais do sistema (GOLOMBEK, 2014). Este tipo de abordagem é comumente utilizado em sistemas críticos, como veículos espaciais e reatores nucleares (VENKATASUBRAMANIAN, V. *et al.*, 2003). Apesar de fornecer um isolamento direto da falha, é uma solução que implica em custos financeiros, manutenção adicional e pode não ser atrativa em aplicações com restrições de espaço físico (CHEN; PATTON, 2012).

A redundância analítica, por sua vez, realiza uma verificação cruzada das medições obtidas, fazendo uso das relações analíticas (ou funcionais) entre as variáveis do processo monitorado. É uma abordagem que utiliza, explícita ou implicitamente, um modelo matemático do sistema e, comparando o modelo obtido com as medições disponíveis do sistema, realiza a detecção de falhas. Os conceitos de redundância física e analítica são ilustrados na Figura 6, em que os componentes replicados são representados por um conjunto de sensores.

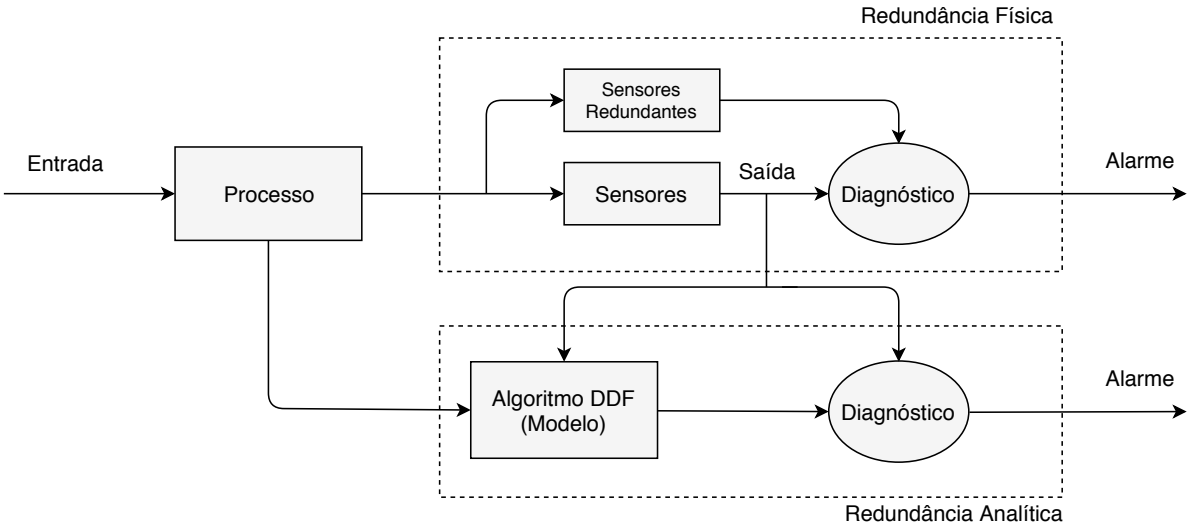


Figura 6 – A redundância física vs redundância analítica. Adaptado de Chen e Patton (2012).

Com relação à categorização de métodos que utilizam redundância analítica, autores apresentam perspectivas diversas. Frank (1990) e Isermann (2006), por exemplo, dão maior enfoque no conceito de modelo, em que os processos são representados, quantitativamente ou qualitativamente, por modelos matemáticos. A partir deste modelo e das variáveis medidas, atributos especiais são extraídos, como parâmetros,

variáveis de estado ou resíduos, que são então utilizados para a geração de sintomas analíticos. Estes sintomas, são, por sua vez, a base para o diagnóstico de falhas.

Outra perspectiva, como no trabalho realizado por Dai e Gao (2013), toma como foco o conceito de dado, sob a perspectiva de que todo sistema de DDF é um sistema de processamento de dados. A classificação das estratégias de DDF, portanto, é realizada com base no modo com que cada método processa os dados disponíveis para a realização da DDF. Nesta perspectiva, um esquema de DDF possui 3 elementos principais: a) uma forma de redundância de informação, que pode ser expressa na forma de um modelo, uma base de conhecimentos derivada de dados ou padrões conhecidos de sinais; b) dados coletados durante a operação do sistema, que serão comparados na próxima etapa com a representação obtida no item anterior e; c) um comparador-classificador, fornecendo um resultado final de classificação. Desta maneira, um método de DDF pode ser definido com base na forma de redundância de informação e na quantidade de dados utilizada, situando-o em uma das seguintes categorias: baseados em modelo, em sinais ou em conhecimento. Este conceito é ilustrado na Figura 7. Os métodos baseados em modelo partem de uma descrição quantitativa ou qualitativa da dinâmica do processo, obtida através de técnicas de modelagem de sistemas. Este modelo é então utilizado para verificação de consistência dos dados obtidos durante a operação do sistema.

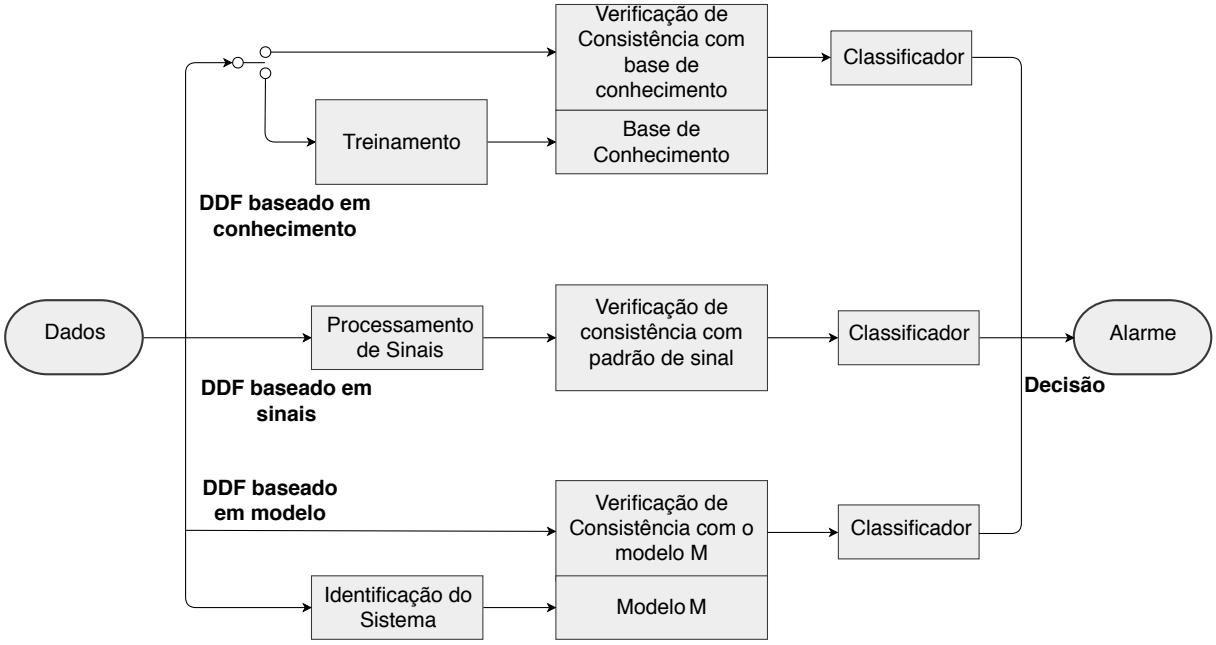


Figura 7 – Fluxo de dados nas diferentes categorias de DDF. Adaptado de Dai e Gao (2013).

Já nos métodos baseados em sinais, a redundância de informação é expressa através da relação entre as falhas e o padrão observado dos sinais (DAI; GAO, 2013). Estes métodos procuram analisar os sinais fornecidos por sensores para a determinação de anomalias, sem o uso de um modelo explícito de entrada e saída (HABRICH;

WAGNER; HELLINGRATH, 2018). Por considerar apenas os valores de saída, técnicas puramente baseadas em sinais apresentam as suas limitações em aplicações que apresentam mapeamentos de entrada e saída ambíguos (GOLOMBEK, 2014) ou cujas anomalias não são definidas apenas por um sinal individual (HABRICH; WAGNER; HELLINGRATH, 2018). Como exemplo, a indicação correta de falha de um veículo não responsivo depende não apenas de sua velocidade linear, mas também de um contexto definido por outros parâmetros, como o comando enviado ao veículo.

Existem casos em que os processos em questão podem ser muito complexos para serem modelados analiticamente, e que os padrões de sinais não são conhecidos a priori. Se, somado a isto, dados históricos do processo estiverem disponíveis, estratégias baseadas em conhecimento podem ser utilizadas. Estas abordagens podem ser vistas como uma extensão do paradigma baseado em modelo (GOLOMBEK, 2014), em que um módulo adicional extrai conhecimento a partir dos dados históricos para gerar uma base de conhecimento. Na Figura 7, o processo de extração de conhecimento consiste na etapa de treinamento. A etapa de treinamento ocorre durante o desenvolvimento do sistema de DDF, mas também pode ser realizada em períodos de manutenção durante a fase de uso, decorrentes de alterações ou ajustes do sistema que se deseja diagnosticar. Por disporem, geralmente, de uma grande quantidade de dados e pouco conhecimento prévio do modelo analítico do processo, os DDFs baseados em conhecimento são frequentemente chamados de métodos *data-driven* (DAI; GAO, 2013).

Nas próximas seções, as diferentes estratégias comumente utilizadas para a detecção e diagnóstico de falhas em UUVs serão apresentadas em maiores detalhes, juntamente com discussões de estudos relacionados às respectivas categorias. O presente trabalho propõe uma abordagem baseada em conhecimento para a detecção de falhas e, portanto, maior atenção será dada às estratégias propostas pertencentes a esta categoria.

3.1.1 Estratégias Baseadas em Modelo

Esta categoria de abordagens procura utilizar um modelo que descreve o comportamento estático ou dinâmico do sistema que se deseja monitorar, o qual é, geralmente, desenvolvido com base em princípios fundamentais da física do sistema (GOLOMBEK, 2014). Durante a operação do sistema, o modelo é executado paralelamente, compartilhando as entradas recebidas pelo sistema que se deseja diagnosticar. Desta maneira, assume-se que a saída fornecida pelo sistema real e pelo modelo sejam similares em condições de operação normal. A diferença entre a saída do sistema e a predição fornecida pelo modelo é chamada de resíduo, que é utilizado para a detecção e, possivelmente, para a etapa posterior de diagnóstico de falhas. Nesta abordagem, apenas uma pequena quantidade de dados é utilizada, que são as medidas obtidas

durante a operação do sistema, no instante em que se deseja detectar a falha.

De acordo com V. Venkatasubramanian *et al.* (2003), estratégias baseadas em modelo podem ainda ser categorizadas com base na natureza do modelo utilizado. Em modelos quantitativos, a compreensão do sistema é expressa com base em relações funcionais matemáticas entre as entradas e saídas do sistema. Os modelos qualitativos, por sua vez, utilizam um conhecimento declarativo (heurístico) do sistema, como informações de variáveis relativas ao sinal, tendência, ordem ou magnitude, por exemplo (CHEN; PATTON, 2012). Modelos qualitativos podem ser utilizados nos casos em que a obtenção de um modelo quantitativo pode ser excessivamente difícil ou demorada, e podem oferecer robustez com relação a incertezas inerentes do sistema (CHEN; PATTON, 2012).

3.1.1.1 Abordagens baseadas em modelos quantitativos

Abordagens baseadas em modelos quantitativos consistem no primeiro tipo de estratégia de detecção de falha baseada em modelo, sendo que os primeiros trabalhos surgiram no início da década de 70 (JONES, 1973). Considerando um modelo $F(\cdot)$ representado no espaço de estados de um sistema com m entradas e p saídas, de acordo com a Equação (1) (DAI; GAO, 2013):

$$y_k = F(u_k, x_k, d_k, f_k, \theta) \tag{1}$$

Em que k denota o índice no tempo, $y_k \in \mathbb{R}^p$ o vetor de saídas, $u_k \in \mathbb{R}^m$ o vetor de entradas, $x_k \in \mathbb{R}^n$ é um vetor de estados n-dimensional, d_k é um vetor de entradas desconhecidas, representando erros de modelagem, distúrbios externos e ruídos de medições, f_k as possíveis falhas a serem detectadas e θ representa os parâmetros do sistema.

Esta classe de abordagens parte do pressuposto de que falhas presentes no sistema causarão alterações na variável de estado x, nos parâmetros do modelo θ ou então na saída y, desviando-se do valor esperado.

Em esquemas baseados na estimação de parâmetros, a supervisão do processo é realizada através do monitoramento da evolução dos parâmetros estruturais do modelo (ISERMANN, 1984; VENKATASUBRAMANIAN, V. *et al.*, 2003). Apesar de os parâmetros do processo serem imensuráveis, pode-se estimá-los com base nas medições de entrada e saída (TIDRIRI *et al.*, 2016). A estimativa pode ser realizada, por exemplo, utilizando Filtro de Kalman (*Kalman Filter - KF*) (KALMAN, 1960) ou, para o caso de sistemas não-lineares, Filtro de Kalman Estendido (*Extended Kalman Filter - EKF*) (PAN *et al.*, 2016). Desta maneira, a detecção de falhas pode ser realizada com base na comparação entre os parâmetros nominais do sistema, computados durante a operação sem falhas, e parâmetros estimados durante a operação *online* (TIDRIRI *et al.*, 2016). A detecção de falhas por meio desta abordagem pode ser facilitada caso

os parâmetros do sistema tenham um mapeamento direto com coeficientes físicos, como amortecimentos, cargas e resistências, por exemplo (DAI; GAO, 2013).

A estimação dos estados x através de observadores, utilizando KF ou observador de Luenberger (LUENBERGER, 1966), por exemplo, também é uma abordagem utilizada para a detecção de falhas. Através da reconstrução dos estados internos do sistema, os valores de saída podem ser estimados, obtendo assim os resíduos, que por sua vez serão utilizados para inferir condições de falha no processo. Em uma terceira classe de estudos, o modelo no espaço de estados pode ser transformado de maneira que sejam obtidas equações que dependam somente de variáveis conhecidas e mensuráveis, i.e., as entradas e saídas u e y, gerando assim as relações de paridade (TIDRIRI *et al.*, 2016). Em um estudo realizado por Gertler (1991), foi demonstrado que esquemas baseados em observadores e equações de paridade levam a geradores de resíduos equivalentes.

De acordo com Patton e Chen (1997), um modelo matemático completo e preciso de um processo, na prática, nunca está disponível. Portanto, incertezas presentes no modelo podem influenciar nos resíduos obtidos, que podem resultar, erroneamente, em um diagnóstico de falha (TIDRIRI *et al.*, 2016). Além disto, entradas desconhecidas (distúrbios) podem atuar no sistema, influenciando assim o resultado final da estratégia de DDF. Por este motivo, a robustez é abordada em diversos estudos, como em Liu, Gao e Han (2016), que utiliza Observadores de Entrada Desconhecida (*Unknown Input Observers*-UIO) para o desacoplamento de efeitos de falha e distúrbios. Pode-se citar, também, o estudo de He, Liu e Hua (2015), em que uma estratégia baseada em EKF adaptativo é utilizada para a detecção de falhas em sensores de corrente e tensão em bancos de bateria de íon-lítio.

Com relação a estratégias de DDF em UUVs, em uma pesquisa sobre estratégias de detecção/tolerância a falhas para AUVs e ROVs realizado por Antonelli (2003), a maior parte dos estudos relatados eram baseados em modelos quantitativos, como os apresentados por Alekseev, Kostenko e Shumsky (1994) e Alessandri, Caccia e Veruggio (1998). Além disto, uma boa parte dos trabalhos se concentrava apenas em subsistemas do veículo, como o sistema de propulsão, por exemplo. No trabalho de Alessandri, Caccia e Veruggio (1999), são consideradas possíveis falhas de atuador (parcial e total) a bordo do UUV Roby 2 (BONO; CACCIA; VERUGGIO, 1995). A detecção de falhas é realizada a partir de um banco de estimadores baseados em EKF para a geração de resíduos, utilizando modelos analíticos do UUV e de possíveis falhas. Esta abordagem apresentou um isolamento eficaz de falhas à custa de um processamento computacional elevado. Outras abordagens baseadas em modelos analíticos focados em propulsores podem ser mencionadas, como a apresenta em Alessandri, Hawkinson *et al.* (1999), que utiliza observadores por modos deslizantes para o banco de estimadores, e Shumsky, Zhirabok e Hajiyev (2010), que utiliza um método de

estimação de estados.

As abordagens baseadas em modelos quantitativos fornecem bons resultados, caso o modelo utilizado reproduza o comportamento do sistema de maneira satisfatória. No entanto, este tipo de estratégia apresenta suas limitações. De acordo com Golombek (2014), a dificuldade de aplicação deste tipo de abordagem se acentua com o aumento da complexidade do sistema a ser monitorado, com relação ao número de componentes e interações entre eles (GOLOMBEK, 2014). Além disto, componentes cujas funcionalidades são representadas em níveis mais altos de abstração, como componentes de sofware, não são adequadamente representados.

3.1.1.2 Abordagens baseadas em modelos qualitativos

Nesta classe de abordagens, a detecção de falhas é realizada através do uso de modelos qualitativos para a comparação do comportamento previsto do sistema com o real. Desta maneira, componentes individuais do processo, tanto no estado nominal como em estados diversos de falha, são simulados, de maneira a descrever os possíveis comportamentos do sistema geral durante a operação, sem a utilização de modelos numéricos (ANGELI; CHATZINIKOLAOU, 2004). A modelagem é realizada através de equações qualitativas derivadas de descrições acerca de relações entre as variáveis de processo, ou então derivadas de equações quantitativas. Os modelos obtidos, por sua vez, contêm conhecimento acerca da estrutura, função e comportamento do processo a ser representado (ANGELI; CHATZINIKOLAOU, 2004). Desta maneira, é possível realizar a modelagem de sistemas cuja complexidade tornaria a modelagem quantitativa inviável, além de ser uma alternativa mais apropriada para a representação de funcionalidades de componentes de software (GOLOMBEK, 2014).

Um método muito utilizado para análise de risco consiste na Análise por Árvore de Falhas (*Fault Tree Analysis - FTA*) (ULERICH; POWERS, 1988; XIANG; YU; ZHANG, 2017). Uma árvore de falhas é um método gráfico que modela o modo pelo qual falhas se propagam dentro de um sistema, i.e., como falhas de componentes levam a falhas do sistema. Em uma árvore de falhas, os nós folha modelam falhas de componentes, enquanto que as portas lógicas, como operações E ou OU, propagam as falhas para o nível superior (RUIJTERS; STOELINGA, 2015). De acordo com Venkatsubramanian (2003), uma desvantagem considerável desta abordagem é que a árvore de falhas construída é apenas tão boa quanto o modelo mental concebido pelo seu desenvolvedor. As árvores utilizadas devem ser capazes de prever e explicar todos os cenários de falhas para que um diagnóstico consistente seja possível.

Outra abordagem que pode ser mencionada é o diagnóstico baseado em consistência (DE KLEER; WILLIAMS, 1987; WILLIAMS; NAYAK, 1996; NARASIMHAN; BROWNSTON, 2007). Sendo $S(f)$ o modelo que representa o sistema na condição de falha f (ou condição sem falhas), o diagnóstico baseado em consistência resume-

se na verificação do par entrada/saída medido com o comportamento β do modelo, sendo β o conjunto de todos os pares de entrada/saída que são consistentes com o modelo (BLANKE *et al.*, 2006). O sistema de DDF Livingstone 2 (KURIEN; NAYAK, 2000) é um exemplo de uma estratégia qualitativa discreta baseada em consistência, em que o modelo do sistema é composto por máquinas de estados finitos que representam modelos de componentes individuais (DEARDEN; ERNITS, 2013). No trabalho de Bajwa, Sweet e Korsmeyer (2003), o Livingstone 2 foi utilizado para o diagnóstico de falha do sistema de propulsão principal de uma espaçonave, e foi também aplicado ao AUV Autosub 6000 (MCPHAIL, 2009) no trabalho realizado por Dearden e Ernits (2013). Através da integração de um modelo de diagnóstico de software com um modelo de hardware, o sistema proposto provou ser capaz de detectar e identificar, ao longo de diversos componentes e subsistemas do veículo, vários tipos de falhas que poderiam ameaçar a segurança do AUV. Além disto, a estratégia provou ter baixo custo computacional, de maneira que as limitações de consumo de energia inerentes a AUVs sejam respeitadas.

3.1.2 Estratégias Baseados em Conhecimento

Em métodos baseados em modelo, um conhecimento a priori é pressuposto. Os métodos baseados em conhecimento, por sua vez, partem do pressuposto de que apenas uma grande quantidade de dados históricos está disponível (VENKATASUBRAMANIAN, V. *et al.*, 2003). Esta abordagem procura extrair o conhecimento implícito presente nos dados históricos através de técnicas de aprendizado de máquina. Este conhecimento será então utilizado para formar a base de conhecimento contra a qual os dados coletados durante a operação serão comparados para a detecção de falhas (GAO, Z.; CECATI, C.; DING, S. X., 2015). O processo de diagnóstico de falhas baseado em conhecimento pode ser visto na Figura 8.

De acordo com Venkatsubramanian (2003), a utilização de sistemas especialistas foi a primeira iniciativa, em uma metodologia formal, de capturar conhecimento a fim de se obter conclusões acerca do processo. Estes métodos se baseiam em um conjunto de regras, no formato *if/then/else*, derivadas de especialistas humanos, a partir de experiências passadas. Este conjunto de regras é então aplicado para a avaliação de dados online para a realização do diagnóstico (GAO, Zhiwei; CECATI, Carlo; DING, Steven X, 2015; VENKATASUBRAMANIAN, Venkat *et al.*, 2003). Dentre as vantagens do uso de sistemas especialistas, pode-se citar a facilidade do desenvolvimento, transparência do modo de raciocínio, e a habilidade de lidar com incertezas (GAO, Z.; CECATI, C.; DING, S. X., 2015). No entanto, este tipo de abordagem é limitado pela qualidade do conhecimento fornecido pelo especialista. À medida que as interações entre os subsistemas crescem em complexidade, a dificuldade de obtenção de conhecimento especialista também aumenta, além de dificultar o gerenciamento de um

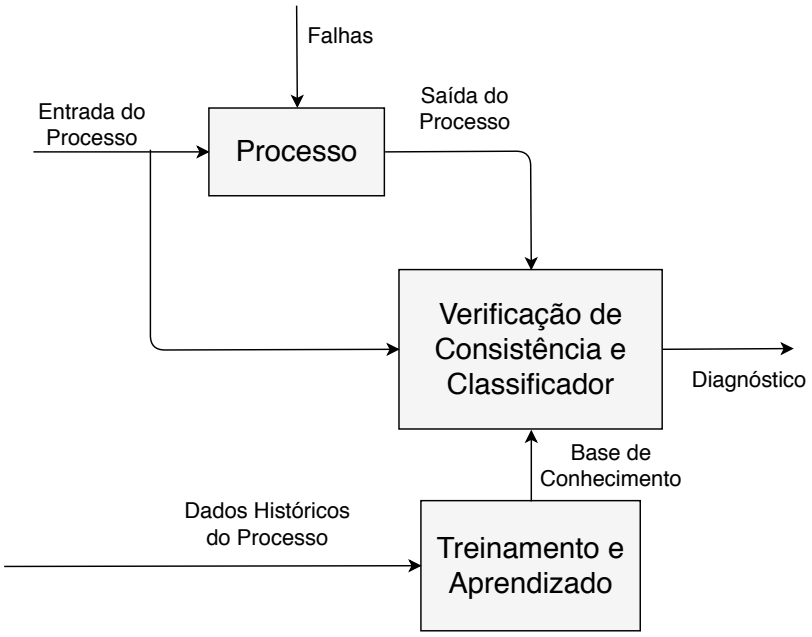


Figura 8 – Esquemático do diagnóstico de falhas baseado em conhecimento. Adaptado de (GAO, Z.; CECATI, C.; DING, S. X., 2015).

conjunto de regras cada vez maior e mais complexo (DEARDEN; ERNITS, 2013).

Dentre as abordagens baseadas em conhecimento, uma possível distinção pode ser feita com base na estratégia de aprendizado, dividindo-as em métodos supervisionados ou não supervisionados (TIDRIRI *et al.*, 2016). Em estratégias supervisionadas, a base de conhecimento é construída a partir de dados de treinamento rotulados, i.e., dados referentes a estados bem-conhecidos do processo: normal ou estados diversos de falha. Desta maneira, abordagens supervisionadas são tipicamente usadas na construção de modelos preditivos (CHANDOLA; BANERJEE; KUMAR, 2009), que procuram encontrar uma descrição generalizada que mapeia todos os possíveis pontos de amostra a classes apropriadas (HABRICH; WAGNER; HELLINGRATH, 2018).

Em muitos casos, a obtenção de dados rotulados é difícil ou até mesmo inviável (MARSLAND, 2011). Assim, técnicas de aprendizado de máquina não supervisionadas são mais indicadas, onde o aprendizado ocorre utilizando dados não rotulados. De acordo com Hodge e Austin (2004), falhas podem ser detectadas em uma abordagem análoga ao agrupamento não supervisionado, assumindo que falhas se apresentarão como pontos distantes (de acordo com alguma medida) daqueles que compõem o comportamento normal do sistema.

Ainda pode ser definido uma terceira categoria, que se situa entre as duas abor-

dadas anteriormente: métodos semi-supervisionados, em que dados de treinamento têm rótulos disponíveis apenas para a classe normal (CHANDOLA; BANERJEE; KUMAR, 2009). Neste caso, o comportamento normal do sistema deve ser isolado de todas as outras possibilidades (PIMENTEL *et al.*, 2014), não havendo distinção entre os diversos estados de falha. Portanto, esta categoria está mais associada à etapa inicial de sistemas de DDF - a etapa de detecção de falhas. Esta classe será discutida em detalhes posteriormente, no contexto de detecção de falhas sob o paradigma de detecção de novidades.

3.1.2.1 Abordagens baseadas em métodos supervisionados

De acordo com Chandola, Banerjee e Kumar (2009), em estratégias supervisionadas, pontos coletados durante a operação do sistema são comparados com o modelo preditivo construído para a determinação da classe à qual cada ponto pertence. Classes podem ser definidas tanto para o estado normal quanto para estados distintos de falhas (CHANDOLA; BANERJEE; KUMAR, 2009). Um ponto a ser considerado neste tipo de abordagem é o desequilíbrio na distribuição de pontos entre classes, uma vez que na maioria das aplicações, dados do comportamento sem falhas são mais abundantes do que os referentes a estados de falha. Implicações advindas deste desequilíbrio são abordadas na literatura de aprendizado de máquina, como em Sun e Chawla (2004) e Phua, Alahakoon e Lee (2004). Outra limitação deste tipo de abordagem é a dificuldade, ou inclusive a impossibilidade, de obtenção de exemplos negativos (pertencentes a classes de falha) (PIMENTEL *et al.*, 2014). Nestes casos, a falta de dados representativos dificulta o processo de aprendizagem, independentemente do desequilíbrio entre classes (WEISS, 2004).

Dentro do contexto supervisionado, as Redes Neurais Artificiais (RNAs) têm recebido considerável atenção, sendo utilizadas principalmente para fins de classificação e aproximação de funções e podem ser classificadas de acordo com a sua topologia: redes de base radial, recorrentes e redes de retropropagação, por exemplo (GAO, Z.; CECATI, C.; DING, S. X., 2015). Devido a sua eficiência na modelagem de processos não-lineares e capacidade adaptativa, as RNAs vêm sendo muito utilizadas para o diagnóstico de falhas (DAI; GAO, 2013). RNAs também podem ser utilizadas de modo não-supervisionado. Um exemplo são as redes neurais auto organizáveis, pelo fato de que sua estrutura é determinada de modo adaptativos baseado na entrada fornecida à rede (VENKATASUBRAMANIAN, V. *et al.*, 2003).

Nos casos em que o conhecimento acerca do sistema é impreciso, a lógica fuzzy pode ser utilizada em conjunto com métodos supervisionados. De maneira similar a sistemas especialistas e árvores de falha, a lógica fuzzy usa um conjunto de regras no formato *se-então*, criadas por especialistas. De acordo com Dai e Gao (2013), sistemas fuzzy são facilmente interpretáveis por se utilizarem de regras de inferência que

usam variáveis linguísticas e termos simbólicos. No entanto, sistemas fuzzy geralmente dependem dos especialistas que os projetam, não apresentando boa capacidade de generalização. Uma abordagem recorrente é a utilização em conjunto de dois ou mais métodos, com a intenção de unir as qualidades dos métodos individuais e suprimir ou compensar as desvantagens. Por exemplo, métodos neuro-fuzzy podem ser utilizados com a intenção de unir a capacidade de generalização de redes neurais com a capacidade de interpretação e expressividade característica de sistemas fuzzy.

Como exemplo de uma abordagem baseada em métodos supervisionados, podemos citar o trabalho de Xiang, Yu e Zhang (2017). Nele, uma estratégia híbrida baseada em conhecimento e modelo qualitativo é proposta para um sistema de tratamento de falhas para um veículo robótico subaquático, abordando as camadas de análise de risco e de tomada de decisão. A abordagem parte de um modelo qualitativo hierárquico de árvore de falhas obtido a partir da integração de informações de sensores, atuadores e estado de operação do veículo. O subsistema de análise de falha é realizado através do treinamento supervisionado de uma rede neuro-fuzzy, enquanto que a tomada de decisão é realizada com base nos princípios de pertinência máxima e limiarização. A escolha pelo uso de uma rede neuro-fuzzy para a análise de falha é motivada pelo fato de que a utilização direta de lógica fuzzy exigiria, para o modelo de árvore de falhas proposto, um número excessivo de regras a serem definidas, impactando na implementação e desempenho de tempo real do sistema.

Outro método popular utilizado para o diagnóstico de falha é o chamado SVM (*Support Vector Machine*) (VAPNIK, 1995). O SVM é um método baseado na teoria de aprendizado estatístico, apresentando algumas qualidades como a capacidade de lidar com um número baixo de amostras para treinamento e alta dimensionalidade (GAO, Z.; CECATI, C.; DING, S. X., 2015).

Um exemplo de estratégia de DDF aplicado a UUVs baseado em SVMs é o trabalho realizado por Zhang, Wu e Chu (2014). Neste estudo, o problema da distribuição desigual e escassez de amostras referente a estados de falha é abordado, utilizando uma estratégia supervisionada para a classificação de múltiplas classes, referentes a falhas diversas de sensor e atuador. A tarefa de classificação múltipla é feita através da decomposição em uma série de tarefas de classificação de duas classes, com amostras pertencentes a classes negativas (não-alvo) ou positivas (alvo). As tarefas de classificação de duas classes, por sua vez, são realizadas utilizando uma estratégia baseada no método SVDD (Support Vector Domain Description) (TAX; DUIN, 1999), que por sua vez utiliza os princípios do SVM. Para lidar com áreas de classificação desconhecidas, como a sobreposição de duas regiões ou então áreas que não pertencem a classe alguma, a lógica fuzzy é utilizada para a determinação de graus de pertinência a classes. O treinamento e avaliação da estratégia é realizada utilizando o AUV Beaver (WANG; ZHANG, 2006) em experimentos em piscina, utilizando falhas

simuladas por software.

3.1.2.2 Abordagens baseadas em métodos não supervisionados

Considerando que em muitos casos reais, conjuntos de dados referentes às falhas do veículo não estão disponíveis e tem-se pouca informação a priori dos dados, uma alternativa é utilização de métodos não supervisionados. A estratégia de DDF proposta por Raanan *et al.* (2018), por exemplo, utiliza métodos de aprendizado não-supervisionados para a descoberta de padrões em coleções de dados não-estruturados para aplicação em um AUV. Para tal, utiliza-se a técnica de modelagem Bayesiana não-paramétrica (BNP) baseada em Alocação Latente de Dirichlet (*Latent Dirichlet Allocation*-LDA), frequentemente utilizada para análise semântica de documentos de texto. Através do modelo BNP, o método é capaz de inferir o número de grupos no conjunto de dados, permitindo que o modelo cresça automaticamente para se ajustar ao tamanho e complexidade dos dados. Após a criação e treinamento de um modelo, é realizada a atribuição de significado semântico a cada componente latente descoberto, para que seja possível uma classificação posterior. O modelo obtido é então utilizado para a detecção e diagnóstico dos dados em tempo real, com base em um classificador baseado em vizinho mais próximo (*Nearest Neighbour*) (COVER; HART, 1967).

3.1.2.3 Detecção de Novidades para a Detecção de Falhas

O processo de DDF, como um todo, consiste em um problema de classificação, uma vez que dados são separados em estados normais e de falha, incluindo a distinção entre o tipo e localização da falha (VENKATASUBRAMANIAN, 2005). Já a etapa inicial de detecção de falha pode ser vista como uma classificação de uma classe (*one-class classification*). Nesta estrutura, o comportamento normal precisa ser diferenciado de todas as possibilidades restantes (PIMENTEL *et al.*, 2014). Neste contexto, a tarefa de detecção de falhas pode ser realizada através da abordagem de detecção de novidades, em que uma classe representa o estado do sistema sem falhas e as possibilidades restantes representam as diversas falhas possíveis.

De acordo com Pimentel *et al.* (2014), no paradigma de detecção de novidades, geralmente se assume que a classe positiva é bem-amostrada, cujas medições são relativamente fáceis de se obter, enquanto que as classes restantes apresentam escassez de amostras. O motivo desta escassez geralmente se dá pelo alto custo e dificuldade de obtenção das referidas amostras. De fato, para o caso específico deste projeto, medições com relação à operação normal do veículo subaquático são mais diretas de se obter, enquanto que a obtenção de uma quantidade significativa de amostras de estados de falha seria problemática, por dois motivos: a necessidade de

causar danos intencionais ao veículo, ou a veículos similares, e a necessidade de se contemplar todos os tipos de falhas possíveis durante a operação do veículo.

Desta forma, um modelo é inferido a partir de dados normais (sem falhas), e então utilizado para a atribuição de índices de novidade $z(x)$, em que x representa dados até então não vistos. A classificação é então realizada através da definição de uma fronteira de decisão delimitada por $z(x) = k$, em que x é tido como normal se $z(x) \leq k$, ou anormal caso contrário (PIMENTEL *et al.*, 2014). A detecção de novidade, portanto, consiste em uma técnica semi-supervisionada, em que apenas a normalidade é modelada (HODGE; AUSTIN, 2004).

De acordo com o definido em Pimentel *et al.* (2014), uma das categorias de métodos de detecção de novidades é aquela baseada em reconstrução. Esta abordagem realiza o treinamento de um modelo de regressão. Desta maneira, quando dados anormais são mapeados usando o modelo treinado, o erro de reconstrução entre o alvo da regressão e o valor observado dá origem a altos índices de novidade. No contexto de DDF, este erro de reconstrução consiste no chamado resíduo. RNAs podem ser usadas desta maneira e oferecem as mesmas vantagens, quando comparado à sua utilização em problemas de classificação tradicionais. Um exemplo de topologia de RNA utilizada para a detecção de novidades são as *Replicator Neural Networks*, também conhecido como autoencoders (DAU; CIESIELSKI; SONG, 2014; HAWKINS *et al.*, 2002). O autoencoder é uma RNA auto-associativa, i.e., uma RNA cujo número de neurônios de saída equivale ao número de neurônios de entrada, e que tem o objetivo de reproduzir os pontos de entrada na camada de saída. Desta maneira, determinado ponto pode ser considerado uma novidade caso o erro de reconstrução obtido seja alto, significando que o modelo de normalidade não foi capaz de reproduzir na saída o vetor de entrada de maneira satisfatória.

Outra técnica possível de ser utilizada em uma abordagem baseada em reconstrução é a Análise de Componentes Principais (*Principal Component Analysis*-PCA). A PCA é uma abordagem estatística multivariada utilizada para redução de dimensionalidade através do cálculo de componentes principais, que são conjuntos de direções ortogonais de máxima variância dos dados (JOLLIFFE, 2002). Desta maneira, as principais características de um processo são obtidas a partir da combinação linear dos dados originais. Além da sua aplicação em métodos de DDF, a redução de dimensionalidade fornecida pelo PCA também é muito utilizada para auxiliar na visualização de dados com muitas dimensões (RINGNÉR, 2008).

Dentro deste contexto, é possível citar como exemplo o trabalho de Zhu e Gu (2007), em que o diagnóstico de falha de sensores em AUVs é abordado. Os autores utilizam redes neurais RBF (*Radial Basis Function*) (KIM; LEE; SIM, 2005) para a realização da análise de componente principal não-linear (NLPCA – *Nonlinear Principal Component Analysis*). A NLPCA, por sua vez, é utilizada para modelar o comporta-

mento normal do sistema. A identificação de eventuais falhas de sensor é obtida a partir do monitoramento dos resíduos, utilizando uma métrica relacionada ao grau de ajuste do modelo aos dados observados, denominada SPE – *Squared Prediction Error*. Para determinado tempo k, uma situação anormal é indicada caso $SPE(k) > \delta$, em que δ é um limiar definido a partir dos dados históricos. A etapa de isolamento das falhas é realizada através da reconstrução das variáveis (DUNIA *et al.*, 1996), e então comparando o índice $SPE(k)$ obtido antes e após as reconstruções. A estratégia proposta foi demonstrada em simulações, considerando dois tipos de sensores: ângulo de yaw e velocidade vertical w.

A função de regressão utilizando redes neurais também é empregada no trabalho de Takai e Ura (1999), em que se utiliza uma rede neural recorrente para a obtenção de um modelo nominal da dinâmica do AUV Twin-Burger (FUJII *et al.*, 1993), a partir de dados coletados de operações sem falha do veículo. Os comandos enviados ao veículo são então inseridos como entrada para o modelo obtido, gerando uma saída estimada para valores de velocidade relativa e de yaw do veículo. Um conjunto de resíduos é gerado a partir da diferença entre a estimativa do modelo e o valor real obtido, gerando uma classificação normal/anormal para cada uma das leituras dos dois sensores. A classificação das duas leituras é então obtida com uma tabela de comparação, permitindo isolar a falha entre falhas de sensor ou atuador. Uma estratégia semelhante foi adotada no trabalho de Sun, Li *et al.* (2016), em que uma rede neural recursiva do tipo Elman (ELMAN, 1990) foi utilizada para a modelagem da dinâmica de um AUV, considerando como entrada os comandos enviados aos 8 propulsores do veículo e como saída as velocidades lineares e de rotação *u,v,w,p,q,r*, como definido na Figura 3. O conjunto de classificações de anormalidade com relação a cada velocidade é então comparada a uma tabela, viabilizando o isolamento da falha entre falhas de sensor e atuador, como no estudo anterior.

De acordo com Pimentel *et al.* (2014), abordagens probabilísticas também podem ser utilizadas para a detecção de novidades. Esta classe de abordagens tem como base a estimativa de Funções de Densidade de Probabilidade (FDP) dos dados utilizados. A distribuição resultante pode então ser limiarizada e utilizada para a definição das fronteiras da região de normalidade. No trabalho de Golombek (2014), a estratégia de detecção de falhas é projetada para aplicação em sistemas robóticos cognitivos cuja estrutura de hardware e software é baseada em componentes. A fonte de informação utilizada para a detecção é baseada no tempo de comunicação entre componentes, partindo da consideração de que o comportamento de um Sistema Robótico Baseado em Componentes (*Component Based Robotic System* - CBRS) resulta da interação e comunicação de seus componentes, e que esta comunicação contém padrões característicos do estado do sistema (normal ou com falha) (GOLOMBEK, 2014). Ao utilizar atributos genéricos de comunicação, o trabalho propõe uma estratégia que pode ser

utilizada em sistemas que sofrem alterações frequentes. O modelo de normalidade é obtido através de técnicas estatísticas, utilizando Estimativa de Densidade Kernel (*Kernel Density Estimator*-KDE) (ROSENBLATT, M., 1956) para estimar a FDP .

4 CONCEITOS DE APRENDIZADO DE MÁQUINA

Neste trabalho, uma série de técnicas estatísticas e de aprendizado de máquina foram utilizadas em diferentes etapas do processo. Neste capítulo, será realizada uma introdução dos conceitos mais importantes das técnicas utilizadas, tanto para a parte de exploração e pré-processamento dos dados quanto para a tarefa de detecção de falhas.

4.1 EXPLORAÇÃO DOS DADOS

A etapa de exploração de dados consiste em uma etapa preliminar de investigação dos dados, auxiliando na compreensão das características dos dados em questão. A exploração de dados pode auxiliar no processo de seleção de técnicas futuras de pré-processamento e análise de dados.

Como uma consideração inicial, é conveniente definirmos alguns conceitos que serão utilizados ao longo deste trabalho. O conjunto de informações coletadas a partir do sistema ou processo monitorado serão denominadas de atributos ou variáveis, de maneira intercambiável. O conjunto de atributos disponíveis, em sua totalidade, será representado pelos n atributos $\mathbf{U} = \{U_0, U_1, ..., U_{n-1}\}$. Cada atributo consiste em um vetor de tamanho m, em que m é o número de observações presente no conjunto de dados. Uma observação corresponde ao valor assumido pelo atributo em determinado instante de tempo, de tal forma que $U_j(k)$ representa a observação do *j*-ésimo atributo no instante k.

4.1.1 Correlação de Pearson

A análise de correlação é um conceito amplamente utilizado na tarefa de detecção e classificação de falhas (DAI; GAO, 2013; SÁNCHEZ-FERNÁNDEZ *et al.*, 2018) e seleção de atributos (CHENG; BAI, 2019). A medida de associação mais comumente usada é o coeficiente de correlação de Pearson (LEE RODGERS; NICEWANDER, 1988), e pode ser entendida como o grau com o qual duas variáveis estão linearmente relacionadas (LIU, 2015). Considerando m medições de dois atributos X e Y $(x_1, x_2, ..., x_m), (y_1, y_2, ..., y_m)$, então o coeficiente de correlação de Pearson r pode ser definido de acordo com a Equação (2) (PUTH; NEUHÄUSER; RUXTON, 2014):

$$r(X,Y) = \frac{\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{m}(y_i - \bar{y})^2}} \tag{2}$$

Em que:

$$\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i \,,$$

$$\bar{y} = \frac{1}{m} \sum_{i=1}^{m} y_i \, .$$

4.1.1.1 Autocorrelação

Como descrito anteriormente, o presente trabalho utiliza um conjunto de atributos cujas observações são realizadas de acordo com uma ordem temporal. Esta sequência de observações ordenadas no tempo pode ser denominada de série temporal. Dentre as técnicas estatísticas existentes para a análise de séries temporais, uma das mais utilizadas é a função de autocorrelação (*Autocorrelation Function* - ACF) (WEI, 2006). A ACF, como o nome sugere, é utilizada quando deseja-se avaliar a correlação de uma variável com relação às suas próprias observações passadas. A ACF pode ser entendida como a aplicação da correlação de Pearson entre valores de uma mesma variável em função do intervalo de tempo (NELSON-WONG *et al.*, 2009). Desta maneira, a autocorrelação para um atributo X para um intervalo de tempo delimitado por k e $k + t$ assumirá o valor obtido por $r(X_k, X_{k+t})$.

Quando a autocorrelação é calculada entre observações adjacentes temporalmente, o valor obtido representa de maneira direta as relações lineares entre as observações. Tomemos, no entanto, um intervalo de tempo maior como, por exemplo, $t = 2$. Neste caso, a ACF não leva em conta efeitos de dependências lineares mútuas relacionadas às observações intermediárias; neste exemplo, as dependências lineares entre X_k e X_{k+1} e entre X_{k+1} e X_{k+2}. Quando as dependências lineares destas variáveis intermediárias são removidas, temos o valor de autocorrelação parcial (PACF - *partial autocorrelation function*) (DAS, 1994). Desta maneira, obtém-se a relação direta entre observações separadas por um intervalo de tempo definido. O PACF é amplamente utilizado para a determinação de tempos de atraso em previsão de séries temporais (ZHANG, R. *et al.*, 2019).

4.1.2 Informação Mútua

Uma consideração importante é de que a correlação de Pearson analisa apenas o grau de relação linear entre variáveis, i.e., efeitos não-lineares não são refletidos pelo coeficiente. Um importante conceito da área de teoria da informação é a Informação Mútua (*Mutual Information* - MI), que pode ser usada como uma medida de interdependência entre duas variáveis (KOPRINSKA; RANA; AGELIDIS, 2015). A Informação Mútua entre duas variáveis X e Y pode ser expressa em termos da distribuição de probabilidade conjunta $p(X, Y)$ e distribuições marginais $p(X)$ e $p(Y)$, dado pela Equação (3):

$$MI(X, Y) = \sum_{x,y} p(x, y) log \frac{p(x, y)}{p(x)p(y)} \tag{3}$$

O conceito de Informação Mútua pode ser compreendido de várias maneiras. Uma maneira possível é interpretá-la como sendo a quantidade com a qual a incerteza acerca de Y decresce quando X é conhecido. Em outras palavras, a quantidade de informação que X contém acerca de Y (PLUIM; MAINTZ; VIERGEVER, 2003). Como X e Y podem ser trocados de posição sem alteração do resultado, $MI(X, Y)$ também é a quantidade de informação que Y contém de X e, disto, a denominação de Informação Mútua.

Uma característica importante da informação mútua é a sua capacidade de capturar dependências não-lineares (YANG *et al.*, 2011). Além disto, é conhecida por ser uma métrica adequada para grandes conjuntos de dados (MEYER, Patrick Emmanual, 2008).

4.1.2.1 A Informação Mútua para Variáveis Contínuas

Para que o cálculo de MI seja possível, deve-se conhecer as distribuições de probabilidade conjunta e marginais de X e Y. Na prática, no entanto, estas distribuições são frequentemente desconhecidas. Para o caso de variáveis discretas, estas probabilidades podem ser estimadas a partir de seus histogramas. Já para o caso de variáveis contínuas, o MI pode ser calculado mediante a aplicação prévia de uma técnica denominada discretização (HU; ZHANG *et al.*, 2011). O processo de discretização realiza a transformação de atributos contínuos em discretos, por meio da associação de cada intervalo de valores contínuos a um determinado valor discreto. Dentre as diversas técnicas existentes, uma maneira possível é realizar a discretização do atributo através da criação de um número pré-determinado de intervalos. Esta categoria de discretizadores é denominada de *binning*, e é amplamente utilizada, por sua simplicidade (GARCIA *et al.*, 2012). Dentro desta categoria, temos ainda duas maneiras de se definir os intervalos: O método *Equal Width* define os intervalos (*bins*) dividindo a amplitude do atributo contínuo em intervalos iguais. Já no método *Equal Frequency*, os intervalos são definidos de maneira que cada intervalo terá o mesmo número m/k de observações, em que m é o número total de observações e k o número de intervalos criados (GUANDALINE, 2016).

4.2 MÉTODOS DE SELEÇÃO DE ATRIBUTOS

A inclusão de atributos irrelevantes e/ou redundantes durante o processo de treinamento de um modelo pode contribuir tanto para o aumento do tempo de aprendizado como também para o sobreajuste aos dados (*overfitting*), afetando a sua capacidade de generalização (FLACH, 2012). Desta maneira, é importante realizar uma etapa prévia de seleção de atributos, de maneira que sejam identificados e selecionados os atributos que mais contribuem para a tarefa de predição do atributo alvo.

De acordo com Karagiannopoulos *et al.* (2007), a etapa de seleção de atributos pode ser realizada por diferentes tipos de métodos. Em métodos do tipo filtro, a seleção é realizada como uma etapa de pré-processamento e é independente do algoritmo indutivo que utilizará os atributos selecionados. Já nos métodos do tipo *wrapper*, a seleção de atributos é realizada englobando um determinado algoritmo de indução, avaliando subconjuntos dos atributos de acordo com o seu poder de predição (GUYON; ELISSEEFF, 2003). Uma vantagem de métodos do tipo *wrapper* é que a avaliação considera a interação entre o algoritmo e o conjunto de treinamento. No entanto, é uma abordagem que demanda esforços computacionais intensos, e pode ser inviável caso o número de variáveis seja muito grande (GUYON; ELISSEEFF, 2003). Os métodos do tipo filtro apresentam requisitos computacionais menores, mas ignoram o efeito do subconjunto selecionado no desempenho do algoritmo que se deseja utilizar (BI *et al.*, 2003). Para ambos os casos, o processo é interrompido de acordo com o critério estatístico utilizado: por exemplo, caso o próximo atributo a ser incluído/removido apresente valor-p abaixo de determinado valor estabelecido, o processo é interrompido.

4.2.1 O Método Stepwise Regression

Dentre os métodos do tipo *wrapper*, o método *stepwise Regression* (EFROYMSON, 1960) é considerado um dos métodos estatísticos mais clássicos para a tarefa de seleção de atributos (DESBOULETS, 2018). Nesta abordagem, uma equação de regressão múltipla é estimada utilizando o método dos Mínimos Quadrados (MMQ) (MENDENHALL; SINCICH; BOUDREAU, 1996), como apresentado na Equação (4):

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x2 + ... + \beta_k x_k + \epsilon \tag{4}$$

Em que y é a variável dependente, $x_1, x_2, ..., x_k$ as variáveis independentes, β_i determina a contribuição da variável independente x_i, β_0 a intersecção com o eixo Y e ϵ é o componente de erro.

A partir do modelo obtido, a seleção das variáveis é feita incrementalmente com base em algum critério estatístico, como por exemplo o valor-p (WESTFALL; YOUNG *et al.*, 1993), o coeficiente de determinação múltipla R^2 ajustado (DRAPER; SMITH, 2014) ou o critério de informação de Akaike (AKAIKE, 1974).

O método *stepwise Regression* pode adotar diferentes estratégias de busca (PETER; BRUCE, 2017). Em métodos do tipo *forward selection*, parte-se do subconjunto vazio, e os atributos são progressivamente incorporados, enquanto que em métodos do tipo *backward elimination*, inicia-se com o conjunto completo de atributos, sendo que os menos promissores são progressivamente eliminados (GUYON; ELISSEEFF, 2003).

4.2.2 Análise de Multicolineridade

No processo de obtenção de um modelo de regressão linear, é possível que haja efeitos de interação entre as variáveis preditoras (ROBINSON; SCHUMACKER, 2009). O efeito de multicolinearidade refere-se à relação linear entre duas ou mais variáveis. Sua presença entre as variáveis independentes pode levar a parâmetros não confiáveis do modelo preditivo, afetando a sua precisão (ALIN, 2010). Portanto, é importante que uma análise de multicolinearidade seja feita.

Uma maneira de verificar indícios de multicolinearidade consiste no uso de matrizes de correlação das variáveis independentes (ALIN, 2010), sendo que os elementos da matriz correspondem aos coeficientes de correlação par-a-par entre as variáveis. Espera-se que, para baixos efeitos de multicolinearidade, os elementos da matriz apresentem baixa correlação.

A Multicolinearidade pode existir entre duas variáveis, ou então entre uma variável e uma combinação linear de outras. Considerando que a análise de correlação leva em conta efeitos apenas entre duas variáveis, é possível que haja efeitos de multicolinearidade sem que níveis altos de correlação sejam verificados (VU; MUTTAQI; AGALGAONKAR, 2015). Uma técnica mais apropriada para o diagnóstico de multicolinearidade consiste na análise do Fator de Inflação da Variância (FIV) (BOLLINGER, 1981), dada pela Equação (5):

$$fiv_i = \frac{1}{1 - R_i^2}, \text{para i} = 1, 2, ..., n \tag{5}$$

Em que fiv_i é o FIV para o atributo U_i e R_i^2 é o coeficiente de determinação múltipla de U_i nas variáveis preditoras remanescentes. O coeficiente R^2 é uma medida estatística utilizada para representar o quão bem o modelo de regressão ajusta o conjunto de dados (MENDENHALL; SINCICH; BOUDREAU, 1996). Assim, a ocorrência de um FIV alto para determinado atributo U_i indica que este está envolvido em ao menos uma dependência linear (ALIN, 2010).

4.2.3 Seleção de Atributos Baseado em Correlação

O método de seleção de atributos baseado em correlação (*Correlation-based Feature Selection*-CFS) (HALL, 1999) é um filtro que avalia subconjuntos de atributos de acordo com uma função de avaliação heurística baseada em correlação. A hipótese que norteia o desenvolvimento do algoritmo é a seguinte:

> "Um bom subconjunto de atributos é aquele que contém atributos altamente correlacionados com a classe alvo, porém não correlacionados entre si."

Apesar de a definição utilizar o termo "classe", o princípio pode ser aplicado também a variáveis dependentes contínuas (TRIANDIS, 1996). Dado um subconjunto

de atributos e conhecendo a correlação entre cada um dos atributos e a variável alvo (ou dependente), além da inter-correlação entre cada par dos atributos do subconjunto, então a correlação entre a soma dos atributos e a variável alvo pode ser definida de acordo com a Equação (6):

$$r_{zc} = \frac{k\overline{r_{zi}}}{\sqrt{k + k(k-1)\overline{r_{ii}}}} \tag{6}$$

Em que r_{zc} é a correlação entre o somatório dos atributos e a variável alvo, k é o número de atributos do subconjunto, $\overline{r_{zi}}$ é a média das correlações entre os atributos e a variável alvo, e $\overline{r_{ii}}$ é a média das inter-correlações entre os atributos (TRIANDIS, 1996).

O termo r_{zc} pode ser, então, utilizado como um grau de "mérito" do subconjunto de atributos em questão. O numerador da Equação (6) pode ser interpretado como uma indicação do quão preditivo da variável alvo é o subconjunto, enquanto que o denominador indica o grau de redundância existente entre os atributos que compõem o subconjunto (HALL, 1999).

Como uma avaliação de todos os possíveis subconjuntos é muitas vezes inviável, deve-se também determinar uma estratégia de busca e critério de parada para a aplicação do algoritmo. Neste sentido, três alternativas são comumente adotadas: *forward selection, backward elimination* e *best first* (HALL *et al.*, 2009). De maneira semelhante ao discorrido na Seção 4.2.1, na estratégia *forward selection*, inicia-se com o subconjunto vazio, e atributos são adicionados um por vez, até que nenhuma possível inclusão resulte em uma avaliação superior ao subconjunto atual. No método *backward elimination*, parte-se do conjunto completo de atributos, e atributos são eliminados um por vez, contanto que a avaliação não diminua. Já para o *best first*, pode-se partir tanto do subconjunto completo quanto vazio. De maneira semelhante às alternativas anteriores, o *best first* percorre o espaço de busca realizando mudanças locais no subconjunto atual. No entanto, caso o caminho sendo explorado se torne pouco promissor, a estratégia pode rever suas decisões e prosseguir a busca a partir de subconjuntos anteriores. Para evitar que todas as possíveis combinações sejam avaliadas, a busca é interrompida se um número determinado de subconjuntos completamente expandidos não resultar em aprimoramentos, comparado ao atual melhor subconjunto.

4.2.4 O método RReliefF

O método RReliefF (ROBNIK-SIKONJA; KONONENKO, 2003) é um método de ordenação de atributos da família Relief de métodos para seleção de atributos, que abrange algoritmos aplicáveis tanto para tarefas de classificação e regressão. O algoritmo principal, Relief (KIRA; RENDELL, 1992), foi proposto para problemas de classificação de duas classes, e se baseia na ideia de que atributos de alta qualidade

devem assumir valores diferentes para instâncias de classes diferentes e valores similares para instâncias de classes iguais. O método original foi estendido para o método ReliefF, capaz de lidar com ruídos e conjuntos de dados com múltiplas classes (KONONENKO, 1994).

Os métodos da família Relief selecionam aleatoriamente uma instância R_i do conjunto de treino, e buscam pelo vizinho mais próximo (usando distância euclidiana) da mesma classe de R_i (*Nearest hit* H) e da classe oposta (*Nearest miss* M). Então, o peso w_f do atributo f é atualizado de acordo com a Equação (7):

$$w_f = w_f - \frac{(\text{diff}_f(R_i, H) - \text{diff}_f(R_i, M))}{m} \tag{7}$$

Sendo que diff_f denota a distância entre as duas instâncias com relação à dimensão do atributo f. O processo é, então, repetido m vezes ($i = 1, 2, ..., m$), para m instâncias escolhidas aleatoriamente. De acordo com a equação acima, percebe-se que o peso do atributo f decresce à medida com que a diferença para f em instâncias da mesma classe é maior do que a diferença para f em instâncias de classes opostas.

Para evitar o comportamento não-determinístico do método decorrente da seleção aleatória de instâncias, o número m pode ser adotado como sendo o tamanho do conjunto completo de treinamento, aumentando também a confiabilidade dos pesos obtidos (KOPRINSKA; RANA; AGELIDIS, 2015). Além de m, o número de vizinhos mais próximos também é um parâmetro que pode ser alterado no cálculo dos pesos. Dentre algumas características do método ReliefF, pode-se citar sua capacidade de lidar com dependências condicionais entre atributos (KONONENKO, 1994), redundância entre atributos (KOPRINSKA; RANA; AGELIDIS, 2015) e capacidade de capturar efeitos lineares e não-lineares (ROBNIK-SIKONJA; KONONENKO, 2003).

4.2.4.1 ReliefF para Regressão - RReliefF

O método RReliefF (RF) é também uma extensão do método original, adaptado para lidar com problemas de regressão, em que o atributo alvo assume um valor contínuo. Neste caso, o conceito de duas instâncias pertencerem a uma mesma classe é substituído pela probabilidade de que os valores do atributo alvo são diferentes. Esta probabilidade pode ser modelada utilizando a distância relativa entre os valores do atributo alvo associados às respectivas instâncias (ROBNIK-SIKONJA; KONONENKO, 2003).

Para explicarmos o método RF, podemos partir da Equação (7), interpretando-a como sendo uma aproximação da seguinte diferença de probabilidades, de acordo com a Equação (8):

$$\begin{aligned} w_f = P(&\text{valor dif. de } f | \text{viz. mais próximo de classes dif.}) \\ -P(&\text{valor dif. de } f | \text{viz. mais próximo de classes iguais}) \end{aligned} \tag{8}$$

Como, para a tarefa de regressão, o conceito de classes não pode ser utilizado, a Equação (8) pode ser reescrita conforme a Equação (9) (ROBNIK-SIKONJA; KONONENKO, 2003):

$$w_f = \frac{P(\text{diff}_C|\text{diff}_f)P(\text{diff}_f)}{P(\text{diff}_C)} - \frac{(1 - P(\text{diff}_C|\text{diff}_f))P(\text{diff}_f)}{1 - P(\text{diff}_C)} \tag{9}$$

Sendo que

$$P(\text{diff}_f) = P(\text{valor dif. de } f|\text{vizinhos mais próximos}),$$

$$P(\text{diff}_C) = P(\text{valor dif. do alvo}|\text{vizinhos mais próximos})$$

e

$$P(\text{diff}_C|\text{diff}_f) = P(\text{valor dif. do alvo}|\text{valor dif. de } f \text{ e vizinhos mais próximos})$$

De maneira análoga à Equação (7), a Equação (9) pode ser aproximada pela Equação (10):

$$w_f = \frac{N_{dC\&df}}{N_{dC}} - \frac{N_{df} - N_{dC\&df}}{m - N_{dc}} \tag{10}$$

Sendo que N_{dC},N_{df} e $N_{dC\&df}$ são ajustados para cada um dos k vizinhos ($I_j, j = \{1, 2, .., k\}$) da instância R_i, e representam os pesos para a diferença entre R_i e I_j para o valor do atributo alvo ($\tau(\cdot)$), para o atributo f e para o atributo alvo e para f, respectivamente, conforme as Equações (11), (12) e (13):

$$N_{dC} = N_{dC} + \text{diff}(\tau(\cdot), R_i, I_j) \cdot d(i, j) \tag{11}$$

$$N_{df} = N_{df} + \text{diff}(f, R_i, I_j) \cdot d(i, j) \tag{12}$$

$$N_{dC\&df} = N_{dC\&df} + \text{diff}(\tau(\cdot), R_i, I_j) \cdot \text{diff}(f, R_i, I_j) \cdot d(i, j) \tag{13}$$

Os pesos são ponderados pela distância entre a instância R_i e o vizinho I_j, representada pelo termo $d(i, j)$ nas equações anteriores, com o pressuposto de que vizinhos mais próximos devem exercer maior influência sobre os pesos (ROBNIK-SIKONJA; KONONENKO, 2003).

4.3 REDES NEURAIS ARTIFICIAIS

As Redes Neurais Artificiais (RNAs) têm sido aplicadas desde a década de 1950 (ROSENBLATT, F., 1957) nas mais diversas áreas do conhecimento. Dentre as

diversas categorias existentes, as RNAs têm em comum o fato de serem modelos de aprendizado de máquina que podem ser usados para tarefas de regressão e classificação, em um contexto supervisionado, e tarefas de aprendizado de representação, em contextos não-supervisionados (HELBING; RITTER, 2018).

4.3.1 O Modelo do Neurônio

A base para a construção de RNAs são unidades de processamento de informações chamadas neurônios. Na Figura 9, pode-se observar os três componentes básicos do modelo de um neurônio k:

1. Um conjunto de sinapses, caracterizadas por um peso. Um sinal x_j na entrada da sinapse j conectada ao neurônio k é multiplicado pelo peso sináptico w_{kj}.
2. Um somador, responsável por somar os sinais de entrada, após a ponderação pelos pesos w_{kj} do neurônio.
3. Uma função de ativação, responsável por limitar a amplitude de saída do neurônio, tipicamente no intervalo [0,1] ou [-1,1].

Além disto, o modelo inclui o $bias(b_k)$, que tem o efeito de aumentar ou diminuir o valor resultante da entrada da função de ativação.

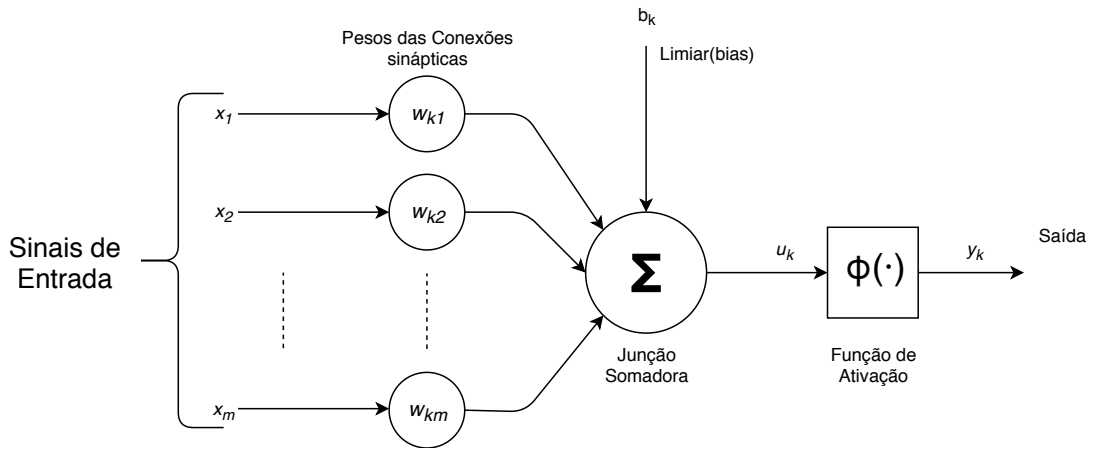


Figura 9 – Modelo de um neurônio (HAYKIN *et al.*, 2009).

Matematicamente, o modelo da Figura 9 pode ser representado pelo conjunto de equações

$$u_k = b_k + \sum_{j=1}^{m} w_{kj} x_j$$

e

$$y_k = \phi(u_k)$$

em que $x_1, x_2, ..., x_m$ representam os sinais de entrada; $w_{k1}, w_{k2}, ..., w_{km}$ os respectivos pesos sinápticos do neurônio k; u_k a saída devido à combinação linear dos sinais de entrada; b_k o valor de $bias$; $\phi()$ a função de ativação; e y_k o sinal de saída do neurônio em questão.

Com relação à função de ativação $\phi()$, a forma mais comum na construção de redes neurais é a função sigmoidal (MENON *et al.*, 1996), uma função estritamente crescente que apresenta um formato de *S*. Um exemplo de função sigmoidal é a função de distribuição logística definida pela Equação (14):

$$\phi(u_k) = \frac{1}{1 + exp(-a \cdot u_k)} \tag{14}$$

Em que a é o chamado parâmetro de inclinação, cuja variação resulta em curvas sigmoidais de diferentes inclinações.

4.3.2 A Rede Perceptron Multicamadas

Partindo da unidade básica, redes de diversas arquiteturas podem ser criadas, por meio de diferentes arranjos e conexões entre neurônios (GOVINDARAJU; RAO, 2013). Dentre as diversas RNAs existentes, as redes do tipo *Multilayer Perceptron* (MLP) estão entre as mais comumente utilizadas (HELBING; RITTER, 2018). As redes do tipo MLP são RNAs de múltiplas camadas: uma camada de entrada, uma camada de saída e uma ou mais camadas intermediárias (ou ocultas). Cada camada é composta por um número de neurônios, sendo que um neurônio de determinada camada recebe os sinais de todos os neurônios que compõem a camada anterior e propaga a sua saída para todos os neurônios da camada posterior (BUENO, 2006). Uma característica importante de MLPs é que o fluxo de informação ocorre sempre para frente, em direção à camada de saída, configurando-a como uma rede do tipo *feedforward* (WUNSCH; LIESCH; BRODA, 2018). Na Figura 10, temos como exemplo uma MLP com três camadas: de entrada, escondida e de saída, com p, m e q neurônios em cada camada, respectivamente. É importante notar que a camada de entrada é responsável por encaminhar os sinais de entrada, sem realizar quaisquer processamentos.

Para a obtenção de um modelo final, deve-se definir os pesos sinápticos de cada conexão que constitui a rede. Isto é feito através do processo de treinamento, em que os pesos da rede são iterativamente reajustados, até que convirjam para determinados valores. Para tal, o algoritmo por retropropagação do erro (*backpropagation algorithm*) é considerado o mais popular para o treinamento de MLPs (HAYKIN *et al.*, 2009). O processo de treinamento consiste em duas fases distintas:

1. Na fase *forward*, os pesos sinápticos da rede se mantêm constantes, e os sinais de entrada são propagados pela rede, camada a camada, até atingir o neurônio de saída.

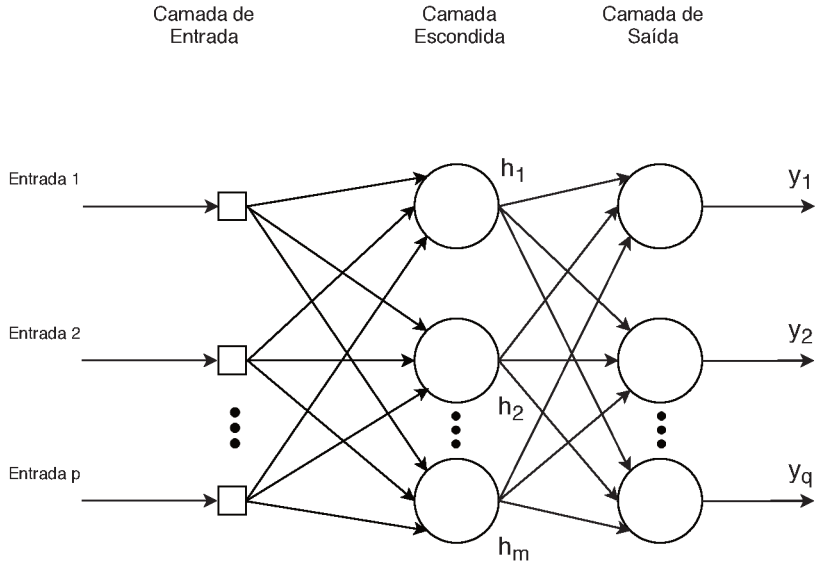


Figura 10 – Rede *Multilayer Perceptron*, com uma camada escondida.

2. Na fase *backward*, um sinal de erro é gerado por meio da comparação entre a saída gerada pela rede e a saída desejada. O sinal de erro obtido é então propagado pela rede no sentido inverso à da fase anterior (para trás). É nesta segunda fase em que ajustes sucessivos nos pesos são realizados.

A seguir, uma breve descrição do algoritmo por retropropagação do erro será feita (HAYKIN *et al.*, 2009). Como exemplo, consideremos uma MLP como ilustrado na Figura 10. Partindo de um conjunto de N amostras com o qual deseja-se realizar o treinamento supervisionado, definido por

$$\mathcal{F} = \{u(n), d(n)\}_{n=1}^{N} \tag{15}$$

Em que $u(n)$ é o n-ésimo estímulo aplicado à entrada, e $d(n)$ é a n-ésima resposta desejada. Desta maneira, o erro instantâneo para o neurônio de saída j é dado por

$$e_j(n) = d_j(n) - y_j(n) \tag{16}$$

Onde $d_j(n)$ é o j-ésimo elemento do vetor $d(n)$ e $y_j(n)$ é o sinal produzido pelo neurônio j na camada de saída, em resposta ao estímulo $x(n)$.

Desta forma, erro quadrático instantâneo nas saídas do MLP é dado por

$$\mathcal{E}(n) = \frac{1}{2}\sum_{j \in C} e_j^2(n) \tag{17}$$

Onde o conjunto C inclui todos os neurônios da camada de saída.

O erro quadrático instantâneo é uma função dos pesos sinápticos da MLP. O método do Gradiente Descendente, ou método Delta (HERTZ, 2018), procura minimizar

a função de erro $\mathcal{E}(n)$, sendo que as conexões entre a camada escondida e a camada de saída são ajustadas por

$$\Delta\omega_{kj}(n) = -\eta(n)\frac{\partial e(n)}{\partial\omega_{kj}(n)} \tag{18}$$

Em que η consiste na taxa de aprendizagem e $\Delta\omega_{kj}(n)$ é a correção aplicada ao peso $\omega_{kj}(n)$. O uso do sinal negativo na Equação 18 considera o gradiente descendente no espaço dos pesos, i.e., procura uma variação nos pesos de forma a reduzir o valor de $\mathcal{E}(n)$. Das duas equações anteriores, temos a lei de ajuste dos pesos

$$\Delta\omega_{kj}(n) = \eta\delta_k(n)h_j(n) \tag{19}$$

Sendo que

$$\delta_k(n) = e_k(n)\dot{\phi}_k(v_k(n)) \tag{20}$$

Em que h_j é a saída do j-ésimo neurônio da camada escondida, e $\phi(v)$ é uma função de ativação diferenciável, como na Equação 14. Da mesma maneira, para as conexões entre a camada de entrada e a camada escondida, temos o ajuste de pesos dado por

$$\Delta\omega_{ji}(n) = \eta\delta_j(n)\xi_i(n) \tag{21}$$

Sendo que

$$\delta_j(n) = \dot{\phi}_j(v_j(n))\sum_k \delta_k(n)\omega_{kj}(n) \tag{22}$$

Em que ξ_i é a saída do i-ésimo neurônio da camada de entrada.

4.3.3 Redes Neurais Artificiais para a Representação de Sistemas Dinâmicos

As redes MLP, apesar de fornecerem um mapeamento não-linear de entrada e saída, apresentam um comportamento estático. Isto é, o fator tempo não é levado em conta neste tipo de configuração e, portanto, não são capazes de representar sistemas dinâmicos (CHEN; PATTON, 2012). De acordo com Haykin *et al.* (2009), existem duas formas de incorporar o componente de tempo na operação de uma RNA:

1. A adoção de uma estrutura de memória na entrada de uma RNA estática, como, por exemplo, uma MLP.

2. Por meio do uso de realimentação.

A realimentação em uma RNA, por sua vez, pode ser realizada de duas maneiras: localmente, em que a realimentação é aplicada a um neurônio individual na rede, e globalmente, em que a realimentação engloba uma ou mais camadas da rede, podendo

envolvê-la por completo. RNAs dotadas de realimentação global são comumente denominadas de Redes Neurais Recorrentes (RNN - *Recurrent Neural Networks*) (HAYKIN *et al.*, 2009).

Um exemplo de arquitetura que incorpora a adoção de estruturas de memória na sua operação é a TDNN (*Time-delay Neural Network*) (CLOUSE *et al.*, 1997). TDNNs são RNAs do tipo *feedforward* (sem o uso de conexões recorrentes), cujas entradas são obtidas a partir de linhas de atraso de observações passadas, em pontos igualmente espaçados. TDNNs têm sido utilizadas para a detecção de falhas de sensores e atuadores em sistemas robóticos autônomos, como no trabalho de Christensen *et al.* (2008). Neste trabalho, os autores partem do pressuposto de que falhas de hardware alteram o fluxo de dados sensoriais e a reação do programa de controle, cujas alterações podem ser detectadas pela TDNN. A avaliação é feita aplicando a estratégia em sistemas robóticos reais em diversos cenários experimentais, avaliando taxas de falso positivo e latência de detecção após a injeção de diferentes tipos de falhas, variando-se diferentes parâmetros do detector. Outro exemplo é o trabalho de Jäger *et al.* (2014), que emprega TDNNs para a detecção de falhas de sensores em sistemas industriais, com o objetivo de fornecer uma abordagem geral para substituir conjuntos complexos de métodos individuais de detecção. Em um primeiro momento, uma etapa de seleção dos atributos mais relevantes é feita e a estratégia por fim é aplicada a uma simulação de sistema dinâmico bem conhecida (sistema de tanques de água) e avaliada mediante a injeção determinística de falhas de sensores.

Redes neurais são também utilizadas para a representação de modelos Não-lineares Auto-regressivo de Entradas Exógenas (*Nonlinear Autoregressive with Exogenous Inputs*-NARX). O termo auto-regressivo indica que o modelo relaciona o valor futuro de uma série temporal com valores presentes e passados desta mesma série, enquanto que as entradas exógenas indicam que valores presentes e passados de um ou mais atributos externos também são utilizados pelo modelo. Desta forma, no modelo NARX, a saída no instante futuro $k+1$ é uma função não apenas dos d valores presentes e passados da entrada u, mas também de q valores presentes e passados da própria saída y. Na Figura 11, são apresentadas duas configurações possíveis para a representação deste tipo de modelo. O bloco z^{-1} representa uma unidade de atraso, cuja saída corresponde ao atributo observado no intervalo de tempo anterior. A associação em série dos blocos z^{-1} forma, então, as respectivas linhas de atraso do modelo. Na Figura 11a, a configuração em malha fechada é apresentada, em que valores passados do valor estimado de saída $\hat{y}$ é realimentado à entrada. Nesta arquitetura, o comportamento dinâmico é obtido tanto pelo uso de estruturas de memória - as linhas de atraso da entrada u e saída y - quanto pela realimentação da saída $\hat{y}$ na

entrada da rede. Neste caso, a saída $\hat{y}_k$ é representada pela Equação (23):

$$\hat{y}_k = F(\hat{y}_k, ..., \hat{y}_{k-q}; u_k, ..., u_{k-d}) \tag{23}$$

Já a Figura 11b apresenta a configuração em malha aberta, em que os valores reais da saída y do sistema representado são utilizados na entrada, ao invés dos valores previstos pela rede. A configuração em malha aberta é frequentemente utilizada na etapa de treinamento da rede (HAYKIN *et al.*, 2009), mas também durante a etapa de operação, no caso em que a previsão de apenas um passo adiante (*one-step ahead*) é realizada (RAMESH; ARULMOZHIVARMAN, 2014). Nesta configuração, a conexão de realimentação não é utilizada, e a saída $\hat{y}_k$ é representada pela Equação (24):

$$\hat{y}_k = F(y_k, ..., y_{k-q}; u_k, ..., u_{k-d}) \tag{24}$$

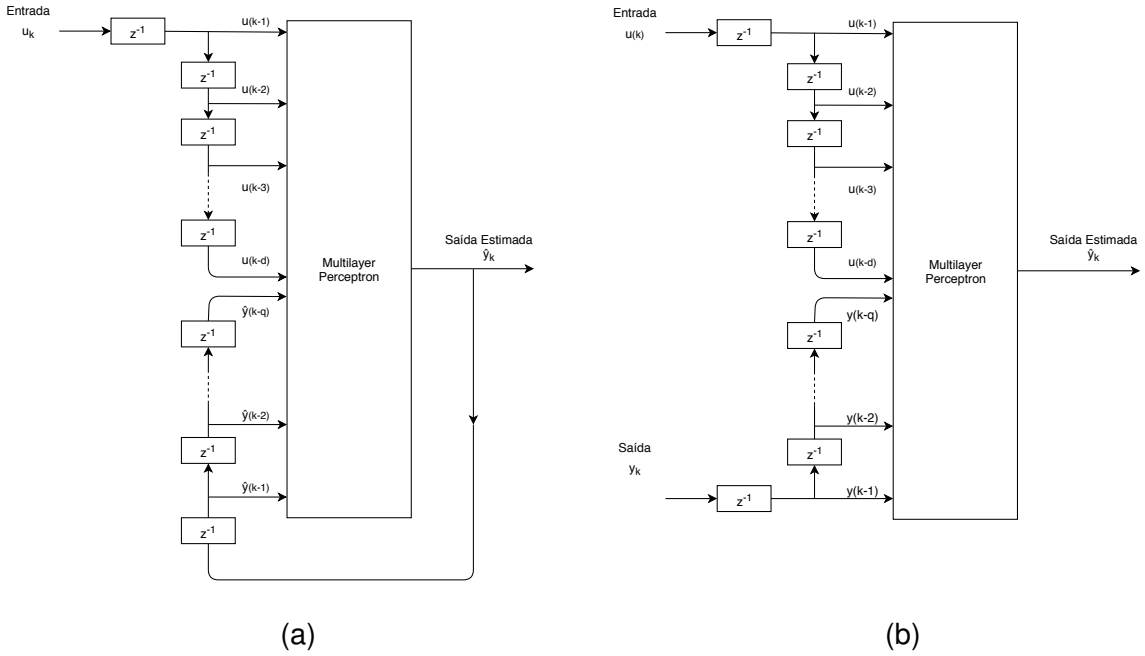
É importante notar que, apesar de na Figura 11 apenas uma entrada estar representada, as redes NARX podem assumir um número maior de entradas, sendo que cada entrada pode assumir um número diferente de unidades de atraso.

Redes NARX, por sua capacidade de modelagem de sistemas dinâmicos não-lineares, são muito aplicadas em sistemas de controle (LEE *et al.*, 2018; AGHADAVO-ODI; SHAHGHOLIAN, 2018), previsão de séries temporais (QIN *et al.*, 2017; IZADY *et al.*, 2013; KUMAR; LOPEZ, 2015) e diagnóstico de falhas (CAPRIGLIONE *et al.*, 2018).

Em um estudo realizado por Nascimento e Valdenegro-Toro (2018), a estimação de modelos dinâmicos de propulsores subaquáticos foi realizada com o uso de uma MLP de comportamento estático, além de três categorias distintas de RNNs - Redes NARX, redes LSTM (*Long Short-term Memory*) (WANG, 2017) e redes GRU (*Gated Recurrent Unit*) (CHUNG *et al.*, 2014). Os resultados do estudo mostraram que a rede NARX foi capaz de identificar de maneira mais precisa as dependências temporais apresentadas pelos propulsores, quando comparado às outras RNAs utilizadas. Em outro trabalho realizado por Capriglione *et al.* (2018), uma rede NARX é utilizada para a detecção de diferentes tipos de falhas injetadas em um sensor de deslocamento linear (potenciômettro) em uma motocicleta. A estratégia baseia-se na geração de resíduos a partir de um modelo NARX do comportamento normal da motocicleta. Uma média móvel é então aplicada aos resíduos obtidos, e o resultado é então comparado a um determinado limiar para realizar a detecção da falha. O desempenho da estratégia é avaliado em termos de falsos alarmes (falsos positivos), detecções perdidas (falsos negativos) e detecções corretas (verdadeiros positivos).

Em um trabalho realizado por Lin *et al.* (1996), resultados experimentais demonstraram que redes NARX foram capazes de preservar informações por um período de tempo de duas a três vezes maior, quando comparado com outros tipos de RNNs. No entanto, redes NARX, juntamente com outras RNNs cujo treinamento é baseado no

algoritmo de gradientes descendentes, não são imunes ao problema de dissipação dos gradientes (*vanishing-gradients problem*) (LIN *et al.*, 1996). Devido à combinação de não-linearidades da rede, a variação de uma entrada temporalmente distante pode surtir pouco ou nenhum efeito durante a etapa de treinamento, dificultando o aprendizado de dependências de longo prazo entre as variáveis (HAYKIN *et al.*, 2009).



(a) (b)

Figura 11 – Modelo não-linear auto-regressivo com entradas exógenas (NARX). (a)- Configuração malha fechada. (b) - Configuração malha aberta

5 ESTRATÉGIA DE DETECÇÃO DE FALHA PARA VEÍCULOS SUBAQUÁTICOS OPERADOS REMOTAMENTE

O presente trabalho pretende implementar uma estratégia de detecção de falhas para ROVs, bem como testá-la e avaliá-la experimentalmente utilizando um veículo subaquático OpenROV V2.8 (STACKPOLE; LANG, 2013). A estratégia de detecção é formulada como um problema de classificação de uma classe (*one-class classification*), em que a classe negativa é definida como a classe normal, e consiste em duas etapas: geração de resíduos e avaliação residual. Para a geração residual, a dinâmica do veículo sem falhas é representada utilizando uma série de modelos de regressão. Construído o modelo, o resíduo é calculado por meio da diferença entre a saída estimada pelo modelo e as leituras reais coletadas. Já a etapa de avaliação residual consiste na classificação final entre os estados normal/falha do veículo, determinada a partir dos resíduos obtidos na etapa anterior.

Os n atributos $\mathbf{U} = \{U_0, U_1, ..., U_{n-1}\}$ coletados e utilizados para a implementação da estratégia proposta são descritos na Tabela 3. Dentre estes, os atributos alvo $\mathbf{Y} = \{Y_0, Y_1, ..., Y_{p-1}\}$ representam os p atributos cujo valor deseja-se prever para que a detecção de falhas seja possível. Baseado no pressuposto de que o comportamento dinâmico do veículo é capaz de capturar condições de falha (TAKAI; URA, 1999; SUN; LI *et al.*, 2016; KHALASTCHI *et al.*, 2011), as variáveis de posição e velocidade apresentadas na Seção 2.1 são candidatas para compor **Y**. No trabalho de Khalastchi *et al.* (2011), é demonstrado que a detecção de anomalias em sistemas robóticos é aprimorada quando se considera não as leituras de posição em si, mas sim as suas taxas de variação. Somado a isto o fato de que, no estudo de caso apresentado neste trabalho os sensores de velocidade linear não estão disponíveis, as velocidades angulares foram adotadas como atributo alvo. Portanto, o conjunto **Y**, neste trabalho, consiste nas velocidades angulares p, q e r ao redor dos eixos X_0, Y_0 e Z_0, denominadas GYROX,GYROY,GYROZ na Tabela 3.

Uma visão geral do sistema de detecção proposto pode ser vista na Figura 12.A estratégia é composta por um conjunto de modelos que fornece, em um instante k, uma estimativa $\hat{\mathbf{Y}}(k) = \{\hat{Y_0}(k), \hat{Y_1}(k), ..., \hat{Y_p}(k)\}$ das leituras reais dos atributos alvo $\mathbf{Y}(k)$. Para tal, será utilizado uma série de p modelos de regressão do tipo NARX, em que a saída do j-ésimo modelo consiste na estimativa $\hat{Y_j}(k)$. Como deseja-se prever o valor de apenas um passo adiante, nenhuma conexão de realimentação foi utilizada na rede NARX, tratando-se, portanto, da configuração de malha aberta, como visto na Seção 4.3.3. Cada modelo recebe como entrada não apenas os valores passados da própria variável alvo, mas também valores passados de um subconjunto de atributos $\mathbf{S}_j \subseteq \mathbf{U}$. O subconjunto $\mathbf{S}_j$ representa os atributos mais relevantes para a obtenção de $\hat{Y_j}$, após a aplicação de determinado processo de seleção de atributos, como os apresentados na Seção 4.2.

Tabela 3 – Atributos coletados do OpenROV.

Atributo	**Informação**	**Unidade**
mtarg1	pulso PWM - motor BB	ms
mtarg2	pulso PWM - motor vertical	ms
mtarg3	pulso PWM - motor BE	ms
roll	Ângulo de roll do ROV	$graus$
yaw	Ângulo de yaw do ROV	$graus$
pitch	Ângulo de pitch do ROV	$graus$
depth	Profundidade do ROV	m
LACCX	Aceleração Linear no eixo X0	m/s^2
LACCY	Aceleração Linear no eixo Y0	m/s^2
LACCZ	Aceleração Linear no eixo Z0	m/s^2
GYROX	Velocidade angular de roll	$rot.por segundo(rps)$
GYROY	Velocidade angular de pitch	$rot.por segundo(rps)$
GYROZ	Velocidade angular de yaw	$rot.por segundo(rps)$
SC1I	Corrente do ESC 1 (BB)	A
SC2I	Corrente do ESC 2 (Vertical)	A
SC3I	Corrente do ESC 3 (BE)	A
BT1I	Corrente do banco de baterias 1 (BB)	A
BT2I	Corrente do banco de baterias 2 (BE)	A
temp	Temperatura do ambiente (lida pelo sensor MS5837)	$^{\circ}C$
vout	Tensão saída BeagleBone	V
iout	Corrente de saída BeagleBone	A
cpuUsage	Uso da CPU - Beaglebone	$\%$
timestamp	Tempo Unix (desde 1 Jan 1970)	ms

É importante notar que, por simplicidade, a Figura 12 apresenta apenas uma Linha de Atraso (LA) para cada subconjunto de atributos $\mathbf{S}_j$. No entanto, na realidade, cada atributo que pertence ao subconjunto é dotado de sua própria LA. Além disto, o número de unidades de atraso que compõe cada LA pode ser diferente de uma variável para outra. De maneira análoga, uma LA também é aplicada à variável autoregressiva Y_j.

Uma vez obtido $\hat{\mathbf{Y}}(k)$, um vetor de resíduos $\phi(k)$ pode ser gerado a partir da diferença entre $\mathbf{Y}(k)$ e $\hat{\mathbf{Y}}(k)$. Os resíduos obtidos podem ser então utilizados como entrada para a próxima etapa, de avaliação residual. Considerando que uma operação sem falhas do ROV deve apresentar valores próximos aos estimados, os resíduos obtidos devem aproximar-se de zero. Da mesma maneira, eventuais falhas no veículo devem refletir no vetor de resíduos obtido. Desta maneira, pode-se empregar um método que analise $\phi(k)$ para finalmente classificar o estado do ROV no instante k entre estados de falha e normal. Este processo é apresentado de maneira simplificada na Figura 12.

O treinamento inicial da MLP é realizado utilizando uma base de dados obtida a partir de diversas sessões de operação normal do ROV. As sessões de operação, por sua vez, são compostas pela execução de diversas sequências de manobras pré-estabelecidas. É importante salientar que, nesta etapa, a definição do estado de normalidade do ROV é realizada de maneira qualitativa, com base no julgamento do

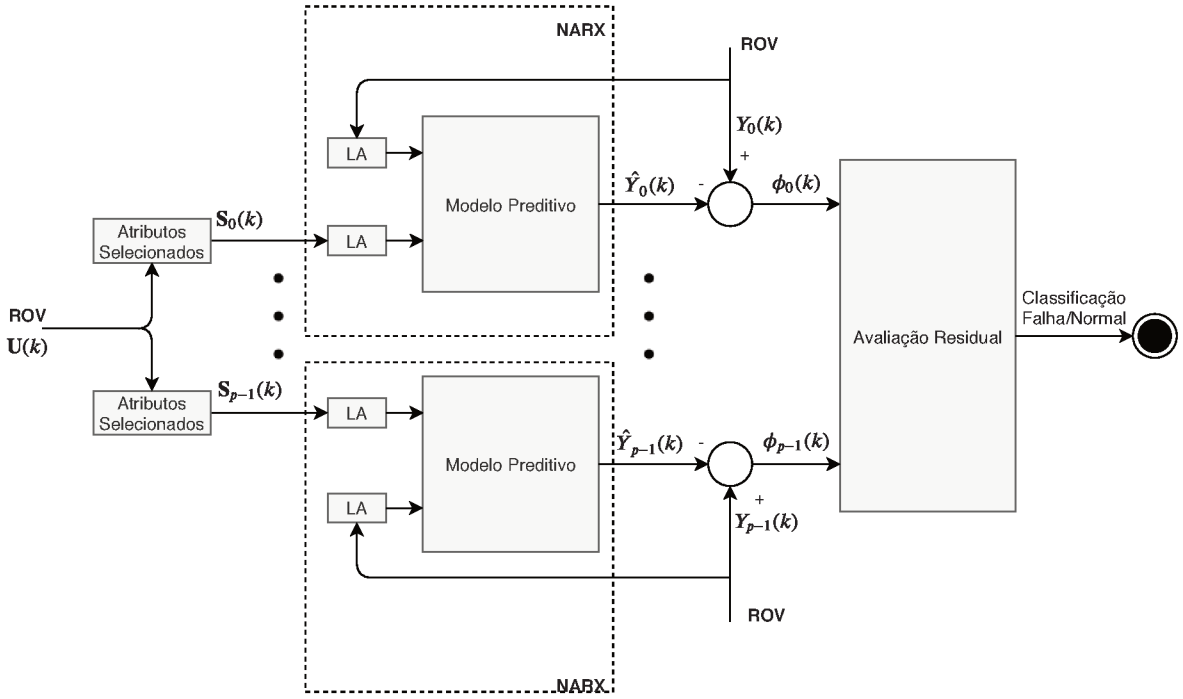


Figura 12 – O Sistema de detecção de falhas.

operador durante a sessão de operação. Caso o veículo não se comporte de maneira satisfatória, de acordo com a opinião do piloto, o trecho em questão da sessão é descartado e não é integrado à base de dados utilizada para a obtenção do modelo de normalidade. Desta maneira, a quantidade de conhecimento de domínio necessária para a obtenção dos dados é reduzida, diminuindo os esforços envolvidos na modelagem. Por outro lado, o critério se torna algo subjetivo, uma vez que diferentes operadores podem apresentar critérios distintos para a definição da normalidade do veículo.

A metodologia proposta para este trabalho será apresentada com o auxílio de exemplos. Os exemplos apresentados ao longo deste trabalho foram elaborados a partir de um conjunto de pontos de amostra coletados com uma frequência de 5Hz, sendo que cada ponto amostrado compreende os 23 atributos apresentados na Tabela 3. Destes, o atributo *timestamp* não é utilizado de maneira explícita na análise, apesar de ter sido utilizado para lidar com as características temporais da aplicação. O processo de coleta de dados será explicado em maiores detalhes na Seção 6.1.2. Os exemplos são elaborados e apresentados com relação a um atributo alvo específico: a velocidade angular GYROZ, sendo que a mesma metodologia será replicada para os dois atributos restantes GYROX e GYROY.

5.1 CARACTERIZAÇÃO DINÂMICA

Como uma etapa preliminar, uma análise exploratória dos atributos é realizada, com o objetivo de se caracterizar as relações dinâmicas entre eles. Os resultados

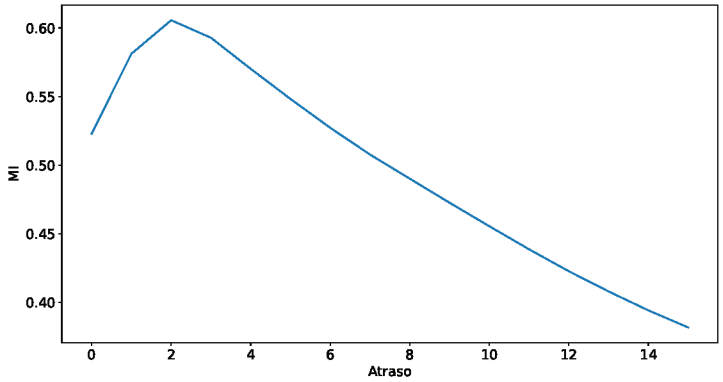


Figura 13 – Medidas de informação mútua (MI) entre *GYROZ* e *mtarg1*, para diferentes atrasos de tempo.

desta etapa serão então utilizados nas etapas subsequentes de seleção de atributos e seleção de unidades de atraso do modelo de regressão.

5.1.1 Dependências Temporais entre Atributos

Uma abordagem direta para a seleção de atributos seria avaliar a relação entre as variáveis considerando um mesmo índice temporal k. No entanto, dependências dinâmicas do sistema representado implicam no fato de que atributos podem influenciar outros atributos em intervalos de tempo futuros. Tomemos como exemplo o cenário de um comando de "giro à direita" enviado ao ROV. Neste caso, o comando de entrada do motor de bombordo, por exemplo, pode influenciar a velocidade angular do veículo apenas no próximo intervalo de tempo, ou ainda em intervalos mais à frente, dependendo das características de resposta do motor e características físicas do ROV. Portanto, se a seleção de atributos é realizada dentro de um mesmo intervalo de tempo, é possível que interações entre os dois atributos citados sejam subestimadas. Por este motivo, medidas de Informação Mútua entre as variáveis são calculadas para diferentes deslocamentos de tempo entre elas.

Como um exemplo, a Figura 13 nos mostra as medidas de MI entre os atributos *mtarg1* e *GYROZ* em função do deslocameanto temporal entre as duas variáveis. Considerando a frequência de amostragem, cada intervalo de tempo corresponde a um intervalo de 0.2s. Assim, valores de MI foram calculados para deslocamentos temporais de 0 a 3s, com o atributo alvo estando sempre avançado com relação ao outro. Assim, o valor relativo ao atraso 3, por exemplo, denota a medida de interdependência entre o comando *mtarg1* amostrado a 3 intervalos de tempo atrás, em relação ao *GYROZ*. Nota-se que o valor de MI máximo de 0.60 ocorre para um deslocamento temporal de 2 (0.4s), ilustrando as considerações feitas.

Considerando isto, o mesmo procedimento pode ser feito entre todas as

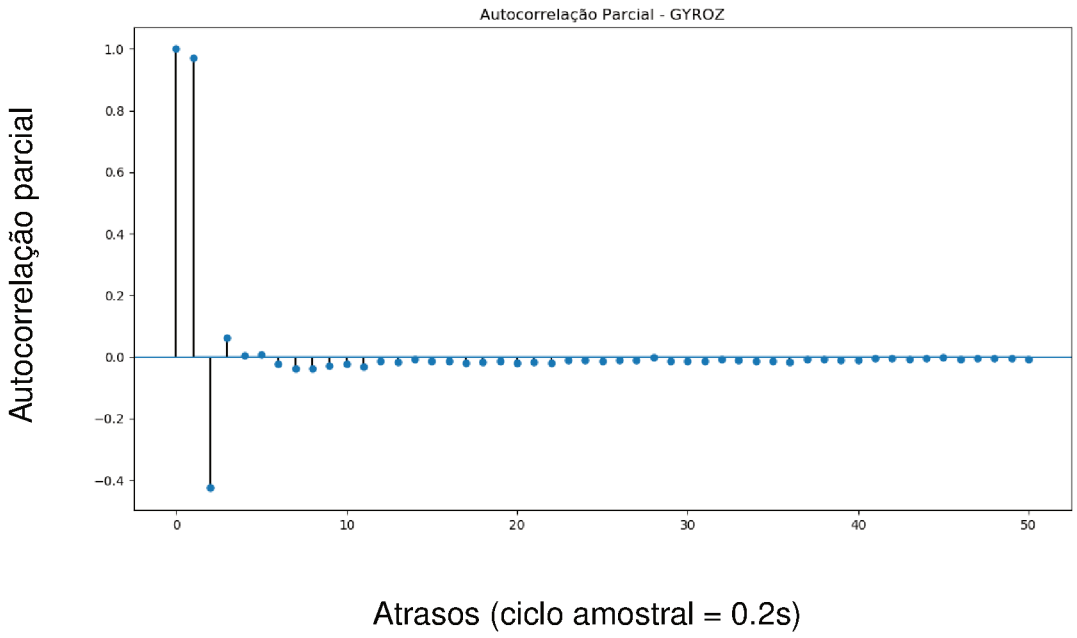


Figura 14 – Autocorrelação parcial de *GYROZ*, em até 50 atrasos de tempo.

variáveis alvo e respectivas variáveis independentes, de maneira a determinar o atraso de tempo para o qual a maior medida de interdependência ocorre (SÁNCHEZ-FERNÁNDEZ *et al.*, 2018).

5.1.2 Análise de Autocorrelação

Além das análises de dependência entre diferentes atributos, é também importante realizar a análise de autocorrelação para as variáveis alvo, como descrito na Seção 4.1. Na Figura 14, é apresentado como exemplo o gráfico com os valores de PACF para os 50 valores passados de *GYROZ* (10 segundos). Através de uma análise visual do gráfico, pode-se perceber que a influência dos valores passados é consideravelmente atenuada após cerca de 7 unidades de atraso.

5.2 SELEÇÃO DE ATRIBUTOS

A etapa de seleção de atributos pode contribuir tanto para a redução no tempo de treinamento quanto para a redução do efeito de sobreajuste, como visto na Seção 4.2. Nesta Seção, será explicado a maneira com que três métodos de seleção de atributos serão aplicados na estratégia proposta: o método Stepwise (SW), RReliefF (RF) e o método baseado em correlação (Correlation-based Feature Selection - CFS). A estratégia proposta será realizada utilizando cada um dos métodos de seleção de atributos mencionados. Os resultados serão, então, avaliados e comparados de acordo com o método de seleção de atributos utilizado.

5.2.1 O método SW

O método SW será utilizado na modalidade *backwards*, partindo do conjunto total de atributos, já levando em consideração os deslocamentos temporais obtidos na etapa anterior de caracterização dinâmica. O processo de seleção foi realizado de acordo com o proposto por Vu, Muttaqi e Agalgaonkar (2015), cuja visão geral pode ser vista na Figura 15.

De maneira prévia, uma análise de multicolinearidade é realizada, com base nos FIV calculados para cada atributo presente no conjunto. A cada iteração, o atributo com maiores efeitos de multicolinearidade é removido, até que todos os atributos restantes apresentem FIV abaixo de determinado valor. Um $FIV = 5$ é comumente utilizado na literatura, ponto para o qual o coeficiente de determinação R^2 para determinado atributo, com relação aos restantes, é superior a 0.8 (ROGERSON, 2014). Neste trabalho, o valor de $FIV = 5$ é adotado como o ponto de parada para a análise de multicolinearidade.

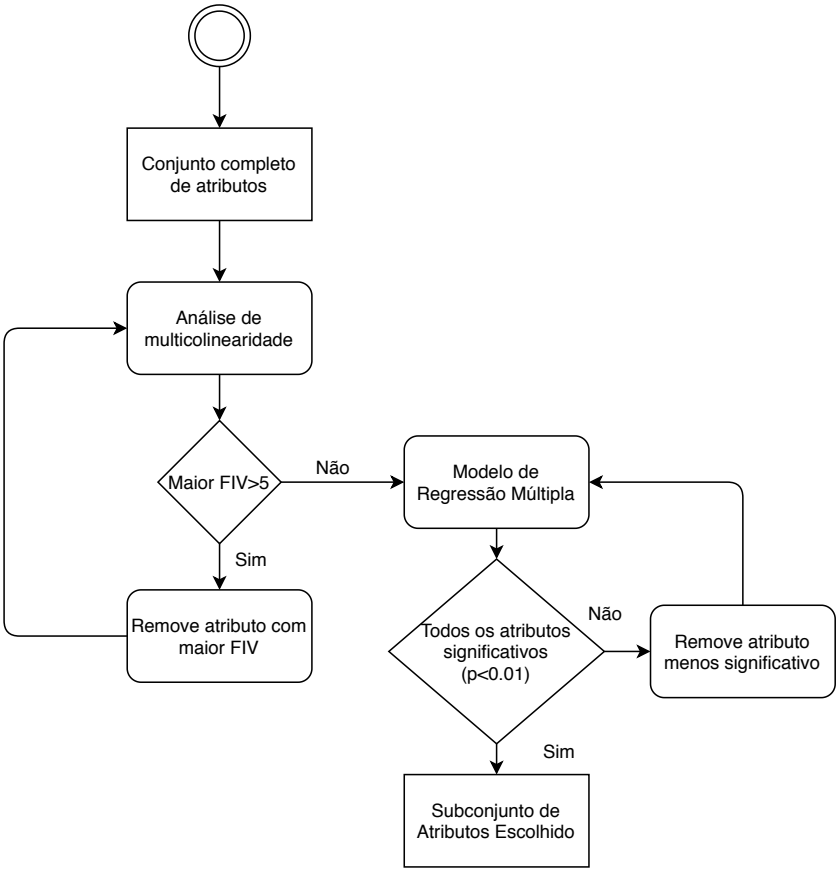


Figura 15 – Procedimento adotado para a seleção de atributos Stepwise.

Concluída a análise de multicolinearidade, o subconjunto resultante é utilizado como ponto de partida para o método Stepwise. A cada passo, um modelo de regressão linear múltipla é construído, e o atributo menos significativo, de acordo com o valor-p, é removido. O processo é interrompido quando todos os atributos resultantes

apresentarem p<0.01, fornecendo assim o subconjunto de atributos final.

5.2.2 O Método CFS

De acordo com o exposto na Seção 4.2, o método de seleção de atributos CFS (HALL, 1999) também será aplicado na metodologia proposta. A estratégia de busca utilizada foi a $bestfirst$, partindo do conjunto vazio de atributos (*forward selection*).

5.2.3 O método RF

Diferentemente do método CFS, a aplicação do método RF, de acordo com o discorrido na Seção 4.2.4, nos fornece uma ordenação dos atributos disponíveis. Resta, portanto, uma maneira de definirmos um subconjunto dos atributos mais relevantes, a partir dos pesos obtidos. Para tal, um limiar é calculado, como proposto por Chen (2012), da seguinte maneira:

1. Os pesos dos f atributos $w_0, w_1, ..., w_{f-1}$ são dispostos em ordem decrescente.
2. A razão $R_i = \frac{w_i}{w_{i+1}}$ é calculada para cada atributo.
3. O ponto de corte é definido no ponto para o qual a diferença máxima entre R_{i+1} e R_i ocorre. O primeiro e último atributos não são usados como ponto de corte (FLOREZ-LOPEZ, 2007).

5.3 SELEÇÃO DE UNIDADES DE ATRASO

Em uma rede NARX, a seleção não somente dos atributos, mas também das unidades de atraso, tanto para os atributos de entrada $\mathbf{U}$ quanto para a saída $\mathbf{Y}$, é crucial (WUNSCH; LIESCH; BRODA, 2018). Idealmente, deve-se incluir as unidades de atraso que sejam significantes, mas ao mesmo tempo mantendo o mínimo de unidades necessárias, de maneira a diminuir o tempo de processamento e efeito de sobreajuste (MAIER *et al.*, 2010).

No presente trabalho, a determinação das unidades de atraso para os atributos de entrada U será realizada utilizando os valores de MI encontrados na etapa de caracterização dinâmica (Seção 5.1). Já a seleção das unidades de atraso de Y é realizada visualmente, de acordo com a análise de PACF, como descrito também na Seção 5.1.

5.4 SELEÇÃO DE MODELO

Uma característica importante em técnicas de aprendizado de máquina é a escolha apropriada de complexidade do modelo: se um modelo é muito complexo,

sua capacidade de generalização pode ser prejudicada, ao passo que um modelo excessivamente simples não irá capturar satisfatoriamente a informação presente nos dados (CLAESEN; DE MOOR, 2015). Este balanço pode ser feito por meio do ajuste de hiperparâmetros, que são usados para configurar vários aspectos do algoritmo de aprendizagem, e influenciam largamente no desempenho do modelo resultante.

No presente trabalho, o ajuste dos hiperparâmetros será realizado utilizando uma estratégia de busca em *grid* (*grid search*) (PEDREGOSA *et al.*, 2011), em que múltiplas combinações de hiperparâmetros são utilizadas para o treinamento do modelo, em busca do conjunto de parâmetros que nos fornece o melhor desempenho.

A métrica de desempenho utilizada foi o erro quadrático médio (MSE -*mean squared error*). Com a intenção de se obter uma estimativa imparcial dos erros obtidos pelo modelo, o processo de validação cruzada será utilizado (TAN, 2018). O processo de validação cruzada particiona o conjunto de dados utilizados para treino em k partes iguais. Neste trabalho, o número de pastas k adotado é de 5. A cada iteração, uma partição, ou pasta, é escolhida para a validação, enquanto que as restantes são utilizadas para o treinamento, de maneira que cada dado é utilizado $k-1$ vezes para treinamento e apenas 1 vez para teste. O erro total é então calculado através da soma dos erros individuais para cada iteração (TAN, 2018).

Todo o processo de seleção e validação do modelo está exemplificado na Figura 16. Inicialmente, o conjunto de dados como um todo é dividido em duas partes: um grupo de treino e um de teste, na proporção de 70%-30%. Desta maneira, todo o processo de seleção do modelo é realizado no conjunto de treino, deixando o conjunto de testes para a avaliação do modelo final.

Utilizando o conjunto de treino, o processo prossegue realizando uma busca em *grid* através de múltiplas combinações de diferentes valores de hiperparâmetros especificados. Neste trabalho, apenas uma camada escondida é utilizada, e os parâmetros variados no processo de busca são: o número de neurônios da camada escondida (15, 30, 50 e 100), o tipo de função de ativação da camada escondida (Rectified Linear Unit - ReLu ou tangente hiperbólica - tanh) e taxa de aprendizagem η (0.01, 0.2 e 0.2). O intervalo de valores foi definido empiricamente, mediante experimentações prévias. Encapsulado no processo, a validação cruzada é realizada após o particionamento do conjunto de treino em 5 pastas de tamanhos iguais, sendo que para cada iteração temos 4 pastas utilizadas para o treinamento e a pasta restante para a validação. Desta maneira, o processo completo envolve o treinamento de $120 = 5 \cdot 24$ modelos, em que 5 é o número de pastas e 24 é o número de combinações de hiperparâmetros a serem analisadas.

É importante notar que os processos de seleção de atributos e de unidades de atraso, envolvendo as análises de caracterização dinâmica, como visto nas Seções 5.2.1,5.2.2 e 5.3, são realizados 5 vezes, uma para cada conjunto de pastas

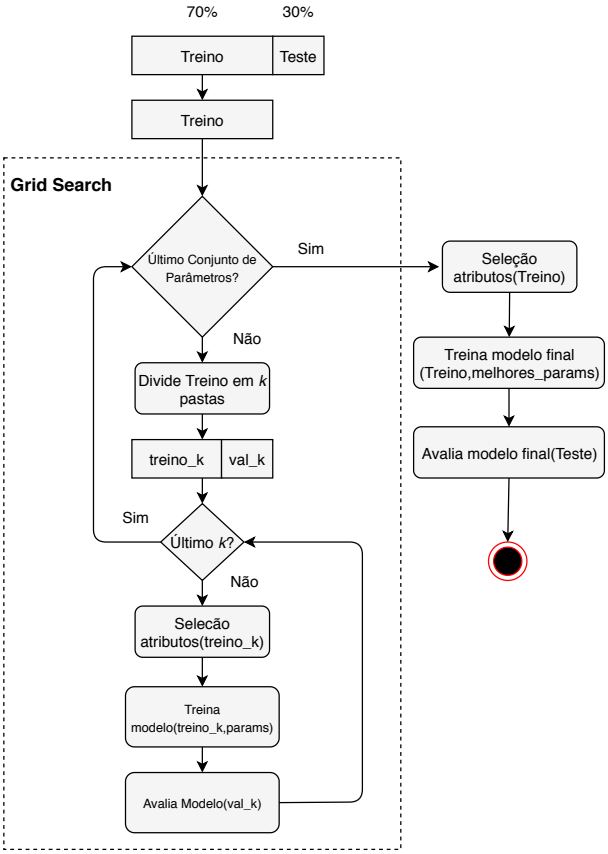


Figura 16 – Processo de validação do modelo preditivo.

de treinamento (REUNANEN, 2003). Desta forma, qualquer informação presente no conjunto de teste é desconsiderada, evitando imparcialidades do processo.

Finalmente, o modelo final será treinado, utilizando os hiperparâmetros que resultaram no melhor desempenho durante a etapa de validação cruzada. Para tal, é realizado novamente os procedimentos para seleção de atributos e unidades de atraso, desta vez para o conjunto completo de treino. O conjunto de teste pode, então, ser usado para a avaliação final do modelo. Por não ser utilizado em nenhum momento no processo de treinamento, o conjunto de testes pode ser utilizado para a comparação do desempenho entre os modelos utilizando métodos de seleção de atributos diferentes, e eventualmente outras diferentes estratégias de detecção.

5.5 AVALIAÇÃO RESIDUAL

Sendo p o número de atributos alvo, ao fim da etapa de geração de resíduos, teremos um vetor de resíduos ϕ com p elementos. Para obter uma classificação do ponto amostrado no instante k, resta a implementação de um bloco funcional capaz de tomar esta decisão com base no vetor ϕ. Neste trabalho, a indicação de falha é

realizada de acordo com a Equação (25):

$$FS = \begin{cases} 0, & \text{se } |\phi_j|(k) < th_j \\ 1, & \text{se } |\phi_j|(k) \geq th_j \end{cases}, \forall j \in (0, 1, ..., p-1) \tag{25}$$

Em que FS representa o estado de falha (0 - sem falhas e 1 - com falhas), ϕ_j o resíduo do *j*-ésimo atributo alvo e th_j o limiar adotado para o atributo em questão. Isto é, uma falha é indicada caso qualquer um dos resíduos calculados para os atributos alvo ultrapasse determinado limiar, que pode ser definido indepentemente para cada alvo. Caso contrário, nenhuma falha é indicada.

5.6 MODELO DE INJEÇÃO DE FALHAS

De maneira a avaliar o desempenho da estratégia proposta, falhas oriundas de defeitos de sensores de diversos tipos serão consideradas. As falhas de sensores foram escolhidas por estarem listadas entre as falhas mais comumente reportadas durante a operação de AUVs e ROVs, de acordo com Antonelli (2003). Não obstante, espera-se que a estratégia proposta seja capaz de capturar outros tipos de falhas. Evidências para isto são exemplos de trabalhos que empregam abordagens similares para a detecção de falhas de diferentes naturezas, como falhas de atuador (CHRISTENSEN *et al.*, 2008; SUN; LI *et al.*, 2016) e falhas de software (GOLOMBEK, 2014). Assim, a avaliação da estratégia frente a falhas de outras naturezas constitui uma importante sugestão para trabalhos futuros.

O modelo de falhas para sensores utilizado neste trabalho segue o adotado nos trabalhos de Zhang, Wu e Chu (2014) e Qi e Han (2007), em que simulações são utilizadas para reproduzir os estados de falha. Considerando y_o como o valor após a injeção de falhas e y a leitura original do sensor, os seguintes tipos de falhas são descritos:

1. Stuck-at : $y_o = \alpha$, em que α é uma constante determinada.
2. Ganho constante: $y_o = \beta y$, em que β é o coeficiente do ganho.
3. Viés constante: $y_o = y + \delta$, em que δ é uma constante.
4. Drift : $y_o = y + tc$, em que t é o índice de tempo e c a inclinação da reta.

Desta maneira, falhas serão injetadas no conjunto de teste, variando os seguintes parâmetros em intervalos pré-estabelecidos: α, β, δ e c.

Um segundo parâmetro importante diz respeito à duração da falha. De acordo com Golombek (2014), falhas podem ser distintas de acordo com sua persistência, com falhas intermitentes (ou transientes) de natureza temporária, enquanto que falhas

persistentes estão continuamente presentes. Por conta de sua natureza temporária, a detecção de falhas intermitentes pode ser ainda mais desafiadora (GOLOMBEK, 2014). Neste trabalho, serão abordadas falhas de natureza intermitente para diferentes períodos de duração. As falhas serão injetadas aleatoriamente no conjunto de teste, comparando posteriormente o resultado do diagnóstico com os períodos de falha registrados.

De maneira a ilustrar cada tipo de falha, a Figura 17 apresenta a comparação entre as curvas de um atributo hipotético original (em azul) e após a inserção de cada tipo testado de falha (em laranja). As Figuras 17a a 17d representam as falhas de stuck-at ($\alpha = 0$), ganho constante ($\beta = 1.1$), viés constante ($\sigma = 0.5$) e drift ($c = 0.05$), respectivamente. Neste exemplo, todas as falhas têm uma duração de 5 segundos, com início e término aos 4 e 9 segundos, respectivamente.

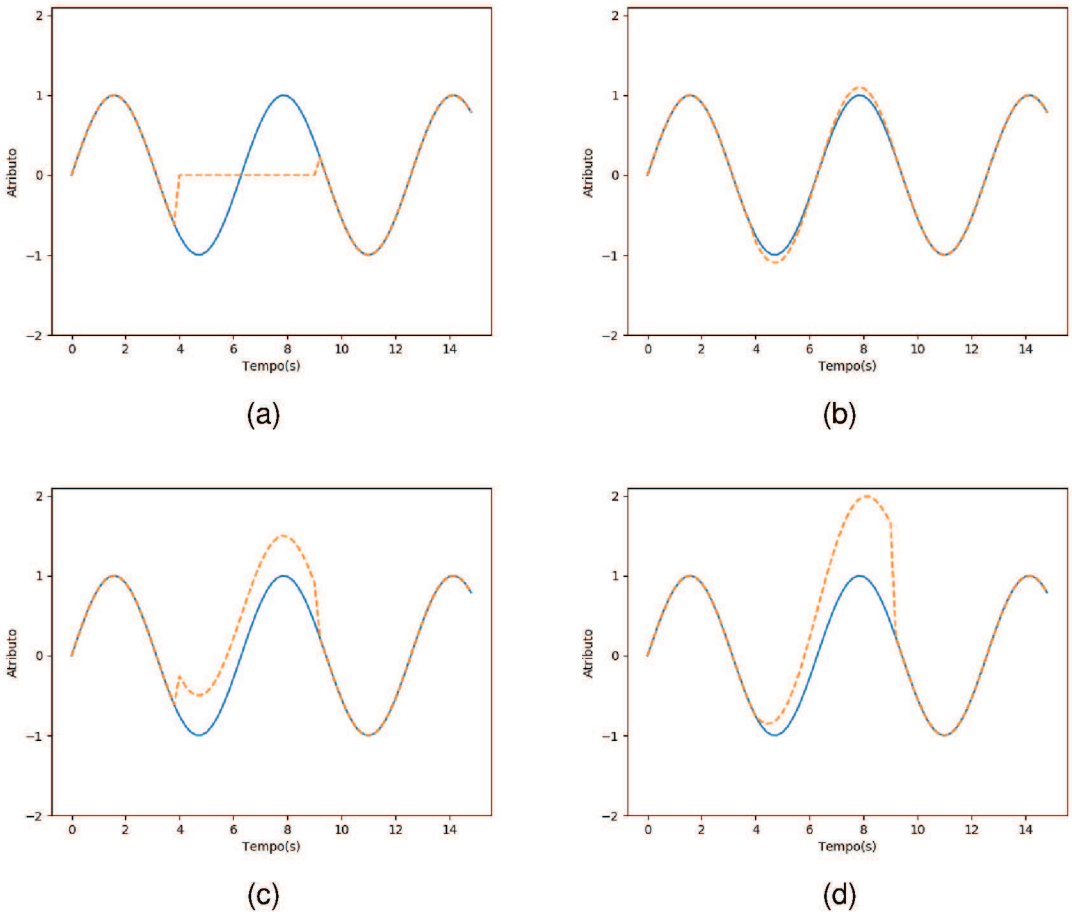


Figura 17 – Exemplos dos tipos de falhas inseridas.(a)- Stuck-at, b) - Ganho constante, (c) - Viés constante, (d) - Drift.

5.7 VISÃO GERAL

No início deste Capítulo, na Figura 12, é apresentado o sistema de detecção de falhas na sua fase de operação, após a conclusão de todos os passos discorridos até aqui. Além disto, é igualmente importante realizarmos uma sumarização de todas as etapas envolvidas no processo de construção da estratégia. Na Figura 18, uma visão geral das principais etapas da estratégia, bem como suas inter-relações, é apresentada.

A etapa inicial consiste na Caracterização Dinâmica. Esta fase é composta pelas análises de Informação Mútua e Autocorrelação Parcial.A Informação Mútua é utilizada para dois propósitos. O primeiro consiste no deslocamento temporal dos atributos, para que a etapa subsequente de Seleção de Atributos possa ser realizada adequadamente. O segundo consiste na definição das unidades de atraso para cada atributo. Já a Autocorrelação Parcial é utilizada para a determinação das unidades de atraso das variáveis autoregressivas GYROX, GYROY e GYROZ.

Tendo selecionado o subconjunto de atributo e linhas de atraso, prossegue-se com a seleção e treinamento do modelo, através da busca em *grid* e validação cruzada. Ao fim desta fase, temos todos os parâmetros do modelo de regressão definidos e os modelos finais já treinados. Em seguida, a etapa de avaliação residual recebe os resíduos gerados e fornece uma saída final de classificação de falha. Finalmente, a estratégia é validada mediante a injeção de falhas, conforme discutido anteriormente.

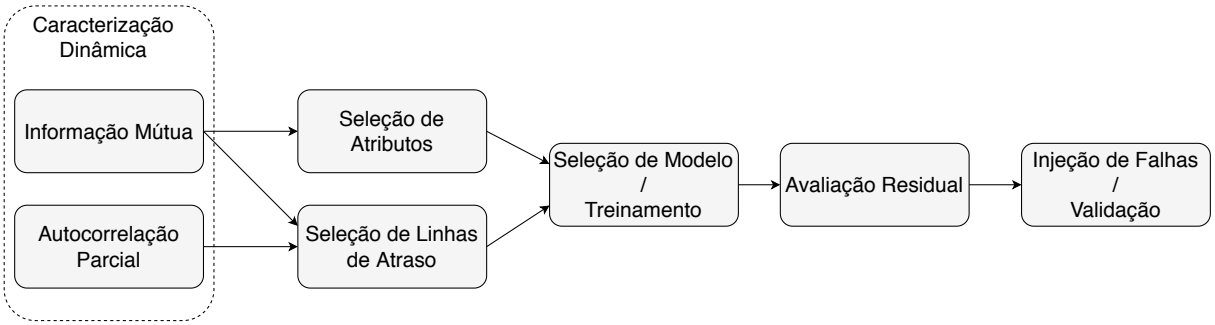


Figura 18 – Visão Geral da Estratégia Proposta

6 A ESTRATÉGIA APLICADA AO OPENROV

Neste capítulo, serão apresentados os procedimentos e resultados obtidos ao aplicar a estratégia proposta ao veículo OpenROV, começando por uma série de etapas preliminares realizadas para garantir que sejam obtidos os dados necessários para a correta implementação da estratégia. Em seguida, serão apresentados e discutidos os resultados referentes a diversos aspectos da estratégia: análises de correlação, seleção de atributos, potencial preditivo e capacidade de detecção.

6.1 ETAPAS PRELIMINARES

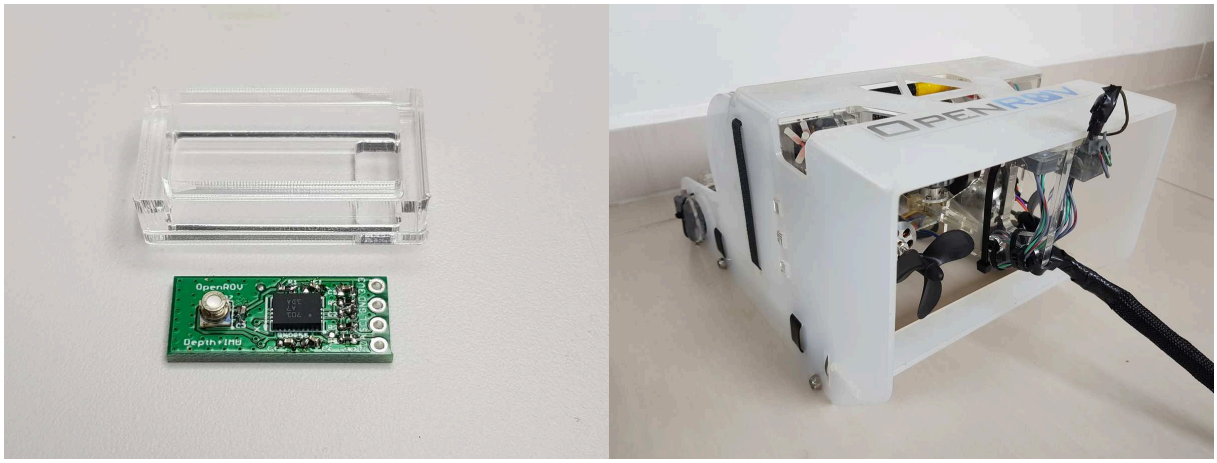
Para que o sistema de detecção possa ser implementado em um caso prático, uma série de etapas prévias devem ser realizadas com o objetivo de fornecer um conjunto de dados cuja quantidade, diversidade, qualidade e estrutura sejam adequadas para a boa operação da estratégia. As etapas preliminares envolvidas para a implementação do sistema de detecção proposto foram definidas como: modificação do ROV, coleta de dados e preparação dos dados. Cada uma das etapas mencionadas será abordada em maiores detalhes nas seções subsequentes.

6.1.1 Modificação do ROV

Como mencionado, o OpenROV V2.8 fornece dados de telemetria relacionados a sensores internos e status de inicialização do veículo. No entanto, dados de navegação, como leituras de profundidade, *roll*, *pitch* e *yaw*, não estão disponíveis inicialmente. Além disto, em experimentações preliminares, foi constatado que a taxa de amostragem de 1Hz fornecido pelo OpenROV é demasiadamente baixa para refletir o comportamento do ROV, prejudicando a capacidade de detecção de falhas do sistema. Portanto, modificações e inclusões foram feitas no veículo, tanto em nível de hardware quanto em software.

6.1.1.1 Hardware

Como detalhado na Seção 2.2.3, a empresa OpenROV disponibiliza o esquemático, arquivos e listas de componentes necessários para a fabricação da placa de IMU de maneira gratuita e pública. Após a confecção do circuito impresso de maneira terceirizada e a aquisição dos sensores e componentes passivos necessários, a montagem da placa foi realizada. O conjunto foi então integrado ao sistema do OpenROV através da utilização dos cabos dedicados para comunicação I2C fornecidos no veículo. Para garantir o isolamento da placa IMU com a água, recipientes de acrílico foram projetados e fabricados. Após a inserção da placa, um adesivo epóxi foi utilizado para o preenchimento do recipiente, prevenindo a entrada de água. Apesar de a placa ter



(a) (b)

Figura 19 – Montagem e instalação da placa de IMU/Profundidade. (a) - Componentes utilizados para a montagem da placa de IMU. (b) - Conjunto de placas IMU montadas e integradas ao OpenROV.

sido projetada para abrigar os dois sensores (MS5837 e BNO055), duas placas foram montadas separadamente, uma para cada sensor, e conectadas serialmente com o barramento I2C, de maneira que eventuais trocas e manutenções possam ser feitas individualmente. Os componentes utilizados e montagem final das placas no veículo podem ser vistos nas Figuras 19a e 19b, respectivamente.

6.1.1.2 Software

Uma vez integrada a unidade de medição inercial ao ROV, os seguintes atributos estarão disponíveis para visualização e coleta através do cockpit: profundidade e os ângulos de euler *roll*, *pitch* e *yaw*. No entanto, o sensor BNO055 é capaz de realizar leituras de atributos adicionais que podem auxiliar no sistema de detecção proposto. Leituras de aceleração, por exemplo, podem fornecer informações importantes com relação a choques físicos do veículo e/ou vibração dos conjuntos motores/hélice. Além disto, em sistemas robóticos, a taxa de variação dos ângulos de euler pode ser mais relevante para a detecção de falhas do que as próprias leituras dos ângulos absolutos (KHALASTCHI *et al.*, 2011). Portanto, o firmware do veículo OpenROV foi alterado para habilitar a aquisição de dados referentes à aceleração linear, em m/s^2, e taxa angular dos ângulos de *roll*, *pitch* e *yaw*, em rotações por segundo (rps).

Outra modificação importante foi a alteração da frequência de amostragem dos dados. Originalmente, os dados de navegação eram atualizados com uma frequência de 10Hz, enquanto que os dados de telemetria apresentavam uma frequência de 1Hz. O firmware do veículo foi alterado para que o ciclo de leitura dos sensores pela placa de controle e envio da lista de atributos para o registrador de dados seja realizado com uma frequência de 5 Hz para todos os dados (navegação e telemetria).

6.1.2 Coleta dos Dados

Uma vez garantida a disponibilidade dos atributos e frequência de amostragem apropriados, pode-se iniciar a etapa de coleta dos dados. Para tal, o ROV foi operado diversas vezes em uma piscina de dimensões 4,54m x 2,30m e 1,20m de profundidade. Por conta da baixa profundidade do local de operação, decidiu-se limitar a sequência de manobras ao plano horizontal, com atuações do motor vertical (*mtarg2*) sendo realizadas apenas para correção da profundidade durante a operação.

A coleta dos dados foi realizada executando sequências de manobras pré-estabelecidas, de maneira que uma ampla combinação de movimentos possíveis no plano horizontal fosse capturada. As seguintes sequências de manobras foram utilizadas:

1. Direção para a frente, seguido de movimento a ré.
2. Giro em torno de si mesmo, realizando uma volta completa à direita e outra à esquerda.
3. Movimento completo de “oito”.
4. Movimento de “ziguezague”.
5. Direção para a frente, seguido por meia volta e para a frente novamente, voltando pelo mesmo trajeto.
6. Movimento circular em volta da piscina, em sentido horário e anti-horário.

Dentro de uma mesma sequência de manobras, o fator de propulsão (velocidade) utilizado é mantido constante.

Ao total, 393221 pontos de amostragem da operação normal do ROV foram coletados em 12 dias de operação distintos, sendo que 382860 destes foram obtidos com uma frequência de 5Hz, totalizando aproximadamente 24 horas de operação. As sessões de operação foram realizadas segmentadas em sequências de manobras, como definido anteriormente, com velocidade dos propulsores variando entre os níveis 2, 3 e 4.

6.1.3 Pré-processamento dos Dados

Com o objetivo de tornar os dados disponíveis mais adequados à tarefa de detecção de falhas, uma série de etapas de pré-processamento foram realizadas.

6.1.3.1 Tempo de validade

Como mencionado, o conjunto de dados é obtido a partir de diversas sessões de operação, realizadas ao longo de diversos dias. Decorre disto que, no conjunto

de dados, existem regiões de transição, em que um intervalo grande de tempo (horas ou dias) separa um ponto de outro subsequente. Caso um cuidado especial não seja tomado, é possível que o modelo NARX receba como entrada valores espúrios, prejudicando o desempenho da estratégia. Por este motivo, deve-se garantir que os atributos de entrada sejam apresentados ao modelo preditivo em intervalos de tempo apropriados. Assim, foi imposta a condição de que uma unidade de atraso de qualquer atributo deve corresponder a um intervalo de tempo de no máximo 0.5 segundos. A estratégia passa a realizar a detecção somente quando todos os atributos necessários estão disponíveis, considerando as respectivas unidades de atraso.

6.1.3.2 Remoção de Falhas no Conjunto de Treino

Como os dados coletados são usados para representar um comportamento sem falhas do veículo, é importante realizar alguns procedimentos simples, para minimizar a presença de eventuais falhas no conjunto de dados. Por este motivo, foram removidas as amostras cujo atributo *CPUusage* apresenta um valor igual a 100%. Assume-se que esta situação denota algum defeito que tenha levado ao esgotamento dos recursos de processamento do microcomputador utilizado. Além disto, amostras que apresentaram *BT1I*, *BT2I* ou *iout* nulos também foram removidas, assumindo-se que estes valores de corrente devem sempre apresentar valores não-nulos enquanto o veículo estiver energizado, em um estado normal.

Em seguida, pontos que apresentaram atributos com valores inválidos ou ausentes são eliminados.

6.1.3.3 Normalização

Por fim, para neutralizar os efeitos decorrentes de diferentes atributos quantitativos serem medidos em escalas diferentes (FLACH, 2012), a padronização do conjunto de dados é realizada através da normalização *min-max*, dada pela seguinte relação

$$U^{'} = \frac{U - \min(U)}{\max(U) - \min(U)} \,,$$

em que $U^{'}$ é o atributo normalizado e $\max(U)$ e $\min(U)$ são os valores máximo e mínimo encontrados de U no conjunto de treino.

6.2 A ESTRATÉGIA APLICADA AO OPENROV

Neste trabalho, a estratégia proposta foi aplicada em um estudo de caso real, utilizando o ROV de baixo custo OpenROV. Foram utilizados três modelos de regressão para a geração de resíduos relativos a três atributos alvo: as velocidades angulares ao redor do eixo X, Y, e Z, intituladas GYROX, GYROY e GYROZ, respectivamente. Neste

capítulo, serão apresentados os resultados obtidos ao longo de diversas etapas do trabalho: o processo de validação e seleção do modelo, análise de correlação, seleção de atributos e capacidade de detecção. Cada etapa será discutida e analisada de acordo com os resultados obtidos para cada um dos atributos alvo em questão.

6.2.1 Pré-seleção dos Atributos

Inicialmente, uma pré-filtragem dos atributos disponíveis pode ser realizada, com base em conhecimento de domínio. Por exemplo, o ângulo *yaw* pode ser descartado como um potencial preditor, uma vez que o ROV pode iniciar a operação em qualquer dada orientação. Outra consideração é que, como os pontos foram coletados em uma piscina, a extensão de valores coletados de profundidade não reflete as condições reais de operação de um ROV. Pelo mesmo motivo, a temperatura ambiente *temp* também pode ser removida de análises futuras.

Por fim, deve-se levar em conta o fato de que as velocidades angulares GYROX, GYROY e GYROZ são usadas pelo SiP BNO055 para o cálculo das posições angulares dos seus respectivos eixos. Portanto, o atributo *pitch* não será usado para a construção do modelo preditivo de GYROX, o atributo *roll* não será usado para a construção do modelo preditivo de GYROY e o atributo *yaw*, para o modelo preditivo de GYROZ (este último já tendo sido removido para todos os atributos alvo, devido a considerações anteriores).

Assim, a Tabela 3 pode ser atualizada, removendo os atributos que não foram utilizados na análise. Desta forma, são apresentados na Tabela 4 os atributos que serão utilizados nas etapas subsequentes da estratégia.

6.2.2 Seleção e Validação do Modelo Preditivo

O processo de seleção e validação do modelo foi realizado de acordo com o explicado na Seção 5.4. Uma única camada oculta foi utilizada no processo, sendo que foram variados os valores de três hiperparâmetros: o número de neurônios na camada oculta (15, 30, 50 ou 100) a função de ativação dos neurônios na camada oculta (Rectified Linear Unit - ReLu ou tangente hiperbólica - tanh) e a taxa de aprendizagem η (0.01, 0.1 e 0.2), totalizando 24 possíveis combinações. Para o processo de treinamento, 100 épocas foram utilizadas, com tamanho do lote de 200. A construção da RNA foi feita utilizando a biblioteca Keras (CHOLLET *et al.*, 2018), para a linguagem Python (ROSSUM, 1995).

O processo foi repetido para cada um dos quatro métodos de seleção de atributos testados: Relieff (**RF**), Correlation-based Feature Selection (**CFS**), Stepwise (**SW**) e sem seleção de atributos (**ALL**). As análises prévias de correlação utilizando Informação Mútua e Autocorrelação Parcial, como descrito no Capítulo 5, foram aplicadas da mesma maneira, independentemente do método de seleção de atributo. É importante

Tabela 4 – Atributos disponíveis para análise após pré-seleção.

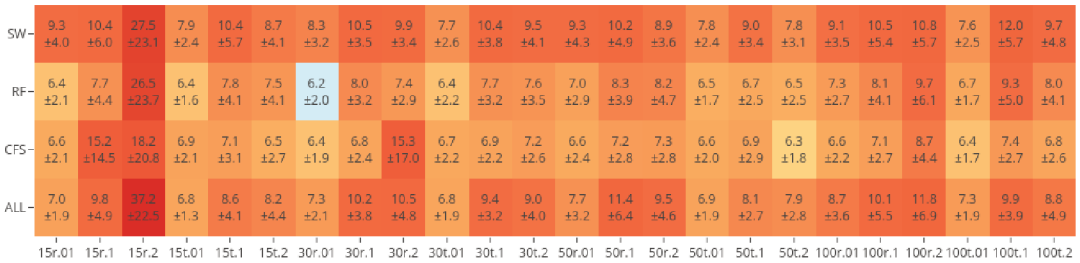
Atributo	**Informação**	**Unidade**
mtarg1	pulso PWM - motor BB	ms
mtarg2	pulso PWM - motor vertical	ms
mtarg3	pulso PWM - motor BE	ms
roll	Ângulo de roll do ROV	$graus$
pitch	Ângulo de pitch do ROV	$graus$
LACCX	Aceleração Linear no eixo X0	m/s^2
LACCY	Aceleração Linear no eixo Y0	m/s^2
LACCZ	Aceleração Linear no eixo Z0	m/s^2
GYROX	Velocidade angular de roll	$rot.por segundo(rps)$
GYROY	Velocidade angular de pitch	$rot.por segundo(rps)$
GYROZ	Velocidade angular de yaw	$rot.por segundo(rps)$
SC1I	Corrente do ESC 1 (BB)	A
SC2I	Corrente do ESC 2 (Vertical)	A
SC3I	Corrente do ESC 3 (BE)	A
BT1I	Corrente do banco de baterias 1 (BB)	A
BT2I	Corrente do banco de baterias 2 (BE)	A
vout	Tensão saída BeagleBone	V
iout	Corrente de saída BeagleBone	A
cpuUsage	Uso da CPU - Beaglebone	$\%$

lembrar que as análises de correlação e seleção de atributos são realizadas internamente ao processo de validação cruzada, dentro do conjunto de treinamento para cada uma das 5 pastas.

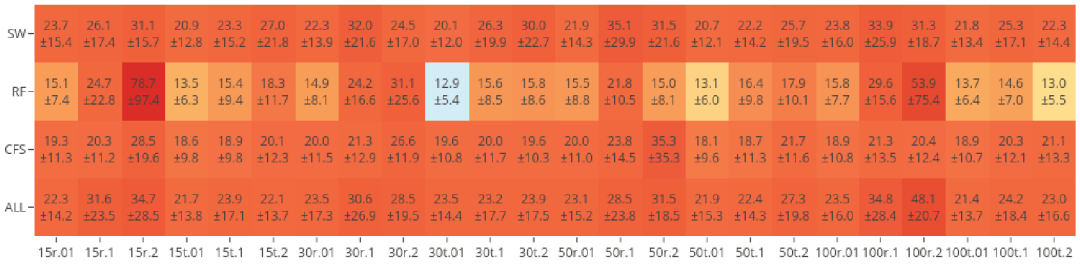
Os resultados do processo de validação para cada um dos atributos-alvo são apresentados na Figura 20, em que cada célula representa a média e o desvio padrão ($\times 10^{-4}$) do erro quadrático médio (*Mean Squared Error* - MSE) aplicado ao conjunto de validação, através das 5 pastas, para a determinada combinação de hiperparâmetros e método de seleção de atributo. A legenda dos hiperparâmetros indicam o número de neurônios e função de ativação da camada escondida ('r'-relu ou 't'-tanh), seguido pela taxa de aprendizado utilizada (0.01,0.1 or 0.2). Desta maneira, o rótulo {15r.01} significa que foi utilizado 15 neurônios, com função de ativação ReLu e η de 0.01, por exemplo.

Os resultados mostram que, de uma maneira geral, a capacidade preditiva da rede NARX é maior para GYROZ, seguido de GYROX e GYROY, e que os melhores resultados ocorreram em combinações de hiperparâmetros diferentes, mas todos para o método RF. Nota-se, também, que as diferenças causadas no MSE pelo método de seleção de atributos utilizado variam de acordo com o atributo alvo. Para GYROX, por exemplo, nota-se que RF e CFS obtiveram resultados similares entre si, e melhores do que os apresentados pelos métodos SW e ALL, ao passo que para GYROY, o método RF apresenta uma clara distinção com relação aos métodos restantes. Já para GYROZ, a diferença de acordo com o método de seleção é mais tênue.

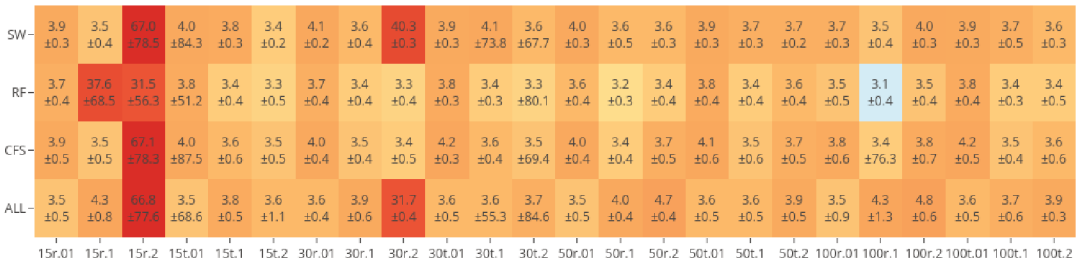
Com relação à escolha dos hiperparâmetros, nota-se que não houve uma combinação superior para todos os casos. No entanto, certas combinações, como {15r.2},



(a) GYROX



(b) GYROY



(c) GYROZ

Figura 20 – Resultados do processo de validação cruzada para cada um dos atributos-alvo. Cada célula apresenta a média e desvio padrão ($\times 10^{-4}$) do erro quadrático médio (mean squared error - MSE) através das 5 pastas.

se provaram particularmente instáveis.

De maneira a tornar mais clara a comparação entre os métodos de seleção de atributos com relação à desempenho e consistência, um processo de *ranking* foi realizado, ordenando os subconjuntos de atributos de acordo com cada combinação de hiperparâmetros. Em casos de valores iguais entre dois ou mais métodos, o *rank* de cada um é atribuído como sendo a média entre os *ranks* dos valores. Os resultados obtidos são sumarizados e apresentados na Tabela 5, sendo apresentados os valores de média e desvio padrão das posições de cada método através das 24 combinações.

O conjunto dos resultados demonstrou que a estratégia apresenta uma boa capacidade de predição, considerando que os atributos alvo encontram-se normalizados, e que os MSEs obtidos se encontram na ordem de $\times 10^{-4}$. Os valores de MSE menores encontrados para GYROZ correspondem com aspectos de manobrabilidade do ROV, uma vez que a posição angular do eixo Z_0 é a mais diretamente relacionada com os

Tabela 5 – *Rank* médio e desvio padrão através das 5 pastas para cada atributo alvo, de acordo com o método de seleção utilizado.

	GYROX	**GYROY**	**GYROZ**
SW	3.67±0.56	3.17±0.82	3.04±0.62
RF	1.67±0.48	1.50±1.02	1.42±0.72
CFS	1.54±0.93	1.88±0.61	2.83±0.96
ALL	3.12±0.54	3.46±0.51	2.71±1.30

atuadores horizontais. Com base na Tabela 5 e comparando-se as Figuras 20a, 20b e 20c, a heterogeneidade observada nos resultados denota os diferentes aspectos dinâmicos entre os três eixos do veículo, além da capacidade da estratégia de capturar estas diferentes dependências.

6.2.3 Treinamento dos Modelos Finais

Uma vez obtido as combinações de hiperparâmetros que forneceram os melhores resultados, o processo de treinamento dos doze modelos finais (4 métodos de selecão · 3 atributos alvo) pode ser realizado. De acordo com os resultados apresentados e seguindo a nomenclatura apresentada anteriormente, temos as seguintes combinações selecionadas:

- **GYROX**: SW-{100t.01}, RF-{30r.01}, CFS-{50t.2}, ALL-{15t.01}
- **GYROY**: SW-{30t.01}, RF-{30t.01}, CFS-{50t.01}, ALL-{100t.01}
- **GYROZ**: SW-{15t.2}, RF-{100r.1}, CFS-{30r.2}, ALL-{50r.01}

6.2.3.1 Linhas de Atraso e Seleção de Atributos

Uma vez definidos os hiperparâmetros a serem utilizados para cada modelos, os procedimentos para a seleção de linhas de atraso e seleção de atributos devem ser realizados novamente, desta vez para o conjunto completo de treino. Primeiramente, a análise de autocorrelação parcial é realizada para a determinação das unidades de atraso para as variáveis autoregressivas (GYROX, GYROY e GYROZ). A seguir, é apresentado os resultados da análise de Informação Mútua para a determinação dos deslocamentos temporais entre as variáveis exógenas e o atributo alvo. Os resultados obtidos nesta etapa são, então, utilizados para o treinamento dos modelos NARX finais da estratégia.

Linha de Atraso - Variável Autoregressiva

Para a definição das unidades de atraso para a variável autoregressiva, a análise de autocorrelação parcial é realizada de acordo com o apresentado na Seção 5.3. Na

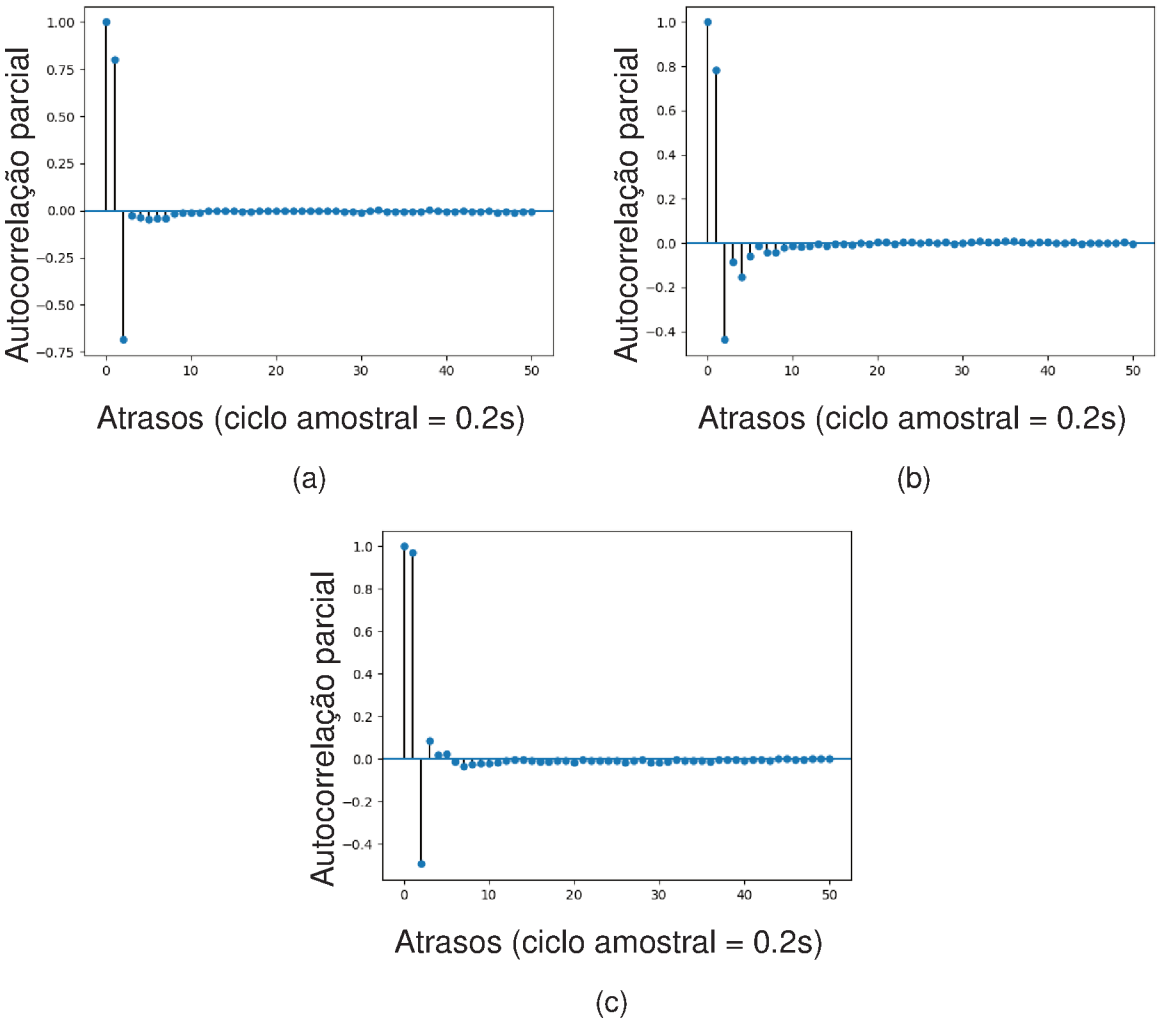


Figura 21 – Valores de autocorrelação parcial para diferentes atrasos de tempo. (a) GYROX, (b) GYROY e (c) GYROZ.

Figura 21, são apresentados os resultados de autocorrelação parcial para os atributos GYROX, GYROY e GYROZ, para até 50 intervalos de tempo passados (10 segundos). Nota-se que, para os três atributos, as autocorrelações são consideravelmente atenuadas para intervalos de tempo maiores. Para todos os casos, a partir de cerca do oitavo intervalo, percebe-se um efeito sazonal, mas de baixa magnitude. A isto, atribui-se idiossincrasias do conjunto de treino, como os padrões de manobras adotadas durante as sessões de operação e limitações de movimentos impostas pelas dimensões do ambiente de operação (piscina) utilizado para a coleta de dados. Estes efeitos, embora existentes, não se prestam para a generalização da estratégia, e não reflete o comportamento dinâmico do veículo. Desta maneira, o número de unidades selecionadas foi de 7, 8 e 7 para GYROX, GYROY e GYROZ, respectivamente.

Linhas de Atraso e Seleção das Variáveis Exógenas

Nas Tabelas 6, 7 e 8, os resultados das análises de caracterização dinâmica e seleção de atributos são apresentados para os atributos GYROX, GYROY e GYROZ, respectivamente. Como descrito no Capítulo 5, a medida de Informação Mútua foi utilizada para a caracterização dinâmica entre os atributos. Para tal, foi utilizado o pacote *infotheo* (MEYER, Patrick E, 2014) para a linguagem R (IHAKA; GENTLEMAN, 1996). Para o cálculo da MI, as variáveis foram previamente discretizadas, também com o uso do pacote *infotheo*, utilizando o método *Equalfrequency* (DOUGHERTY; KOHAVI; SAHAMI, 1995).

O método de seleção SW foi implementado de acordo com o proposto na Seção 5.2.1, por meio do pacote *statsmodels* (SEABOLD; PERKTOLD, 2010) para a linguagem Python (ROSSUM, 1995).

Já o método RF foi implementado utilizando o pacote CORElearn (ROBNIK-SIKONJA; SAVICKY, 2012) para a linguagem R, utilizando decaimento exponencial dos pesos (*ReliefFexpRank*) e número de iterações igual ao conjunto completo de treinamento. O número de vizinhos (*kNearestExpRank*) e quociente para avaliação da distância (*quotientExpRankDistance*) utilizados foi de 70 e 20, respectivamente, de acordo com o sugerido por Robnik-Sikonja e Kononenko (2003).

O método CFS foi implementado utilizando o software de mineração de dados Weka (HALL *et al.*, 2009), com o processo de busca sendo interrompido após 5 expansões consecutivas não resultarem em uma melhoria.

Percebe-se que diferentes dependências temporais foram encontradas para cada um dos atributos alvo. Para GYROX, determinados atributos apresentaram uma correlação máxima de até 5 intervalos de tempo passados (1 segundo), enquanto que para GYROY e GYROZ, dependências de até 4 (0.8s) e 2 (0.4s) intervalos de tempo foram encontradas, respectivamente. Apesar de a análise de MI ter sido utilizada apenas para a determinação dos atrasos temporais, os valores encontrados nos dão uma indicação do grau de dependência de cada atributo com o alvo. As medidas de MI para GYROZ apresentam, de maneira geral, valores mais elevados, o que corrobora com os erros menores encontrados, como visto na Figura 20c. Além disto, percebe-se que os comandos *mtarg1* e *mtarg3* apresentam valores mais elevados para o caso de GYROZ. Este resultado corresponde com o comportamento dinâmico esperado do veículo, como visto na Figura 5.

Já a respeito dos métodos de seleção de atributos, de maneira geral, o método CFS apresentou uma remoção de atributos mais agressiva, se comparado com os outros métodos. Já o método RF removeu, para todos os alvos, os atributos de aceleração linear, além de outros atributos que, verificado pela análise de FIV para o método SW, apresentaram efeitos de redundância com os atributos restantes.

Tabela 6 – Atrasos temporais definidos pela análise de correlação cruzada e atributos selecionados, de acordo com cada método de seleção utilizado, para o atributo alvo GYROX. SW: Stepwise, RF: ReliefF, CFS: Correlation-based, ALL: Todos os atributos, MI: Informação Mútua.

Atributo	ALL	CFS	SW	RF	MI	Atraso
mtarg1	x		x	x	0.36	5
mtarg2	x		x	x	0.02	1
mtarg3	x	x	x	x	0.34	5
pitch	x		x	x	0.15	5
LACCY	x	x	x		0.14	1
LACCX	x	x	x		0.12	1
LACCZ	x		x		0.14	1
GYROZ	x	x	x	x	0.42	3
GYROY	x	x	x	x	0.28	1
SC1I	x	x	x	x	0.30	5
SC2I	x		x		0.04	4
SC3I	x	x		x	0.32	4
BT1I	x		x	x	0.36	4
BT2I	x		x	x	0.35	5
vout	x			x	0.32	4
iout	x		x	x	0.06	4
cpuUsage	x		x	x	0.13	1

Tabela 7 – Atrasos temporais definidos pela análise de correlação cruzada e atributos selecionados, de acordo com cada método de seleção utilizado, para o atributo alvo GYROY. SW: Stepwise, RF: ReliefF, CFS: Correlation-based, ALL: Todos os atributos, MI: Informação Mútua.

Atributo	ALL	CFS	SW	RF	MI	Atraso
mtarg1	x	x	x	x	0.22	1
mtarg2	x	x	x	x	0.04	2
mtarg3	x	x	x	x	0.21	3
roll	x	x	x	x	0.16	1
LACCY	x				0.10	3
LACCX	x	x	x		0.09	1
LACCZ	x	x	x		0.12	1
GYROZ	x		x	x	0.30	1
GYROX	x		x	x	0.27	1
SC1I	x		x		0.19	4
SC2I	x	x		x	0.04	2
SC3I	x		x		0.20	3
BT1I	x			x	0.25	4
BT2I	x		x		0.25	4
vout	x			x	0.23	3
iout	x			x	0.04	4
cpuUsage	x		x	x	0.09	1

Tabela 8 – Atrasos temporais definidos pela análise de correlação cruzada e atributos selecionados, de acordo com cada método de seleção utilizado, para o atributo alvo GYROZ. SW: Stepwise, RF: ReliefF, CFS: Correlation-based, ALL: Todos os atributos, MI: Informação Mútua.

Atributo	ALL	CFS	SW	RF	MI	Atraso
mtarg1	x	x	x	x	0.61	2
mtarg2	x			x	0.06	2
mtarg3	x	x	x	x	0.57	2
roll	x		x	x	0.23	1
pitch	x		x	x	0.15	1
LACCY	x	x	x		0.20	1
LACCX	x	x	x		0.16	1
LACCZ	x		x		0.17	1
GYROY	x		x	x	0.31	2
GYROX	x		x	x	0.38	1
SC1I	x		x	x	0.46	2
SC2I	x		x		0.07	2
SC3I	x			x	0.48	2
BT1I	x		x	x	0.49	2
BT2I	x			x	0.48	2
vout	x		x	x	0.39	1
iout	x		x	x	0.06	1
cpuUsage	x		x	x	0.11	1

Durante a execução do método SW, observou-se que apenas para GYROY houve casos em que atributos foram removidos por conta de sua baixa significância, para os atributos iout, LACCX e BT1I. Em todos os outros casos, os atributos foram removidos durante a etapa de análise de multicolinearidade, sendo que os restantes apresentaram valor-p < 0.01. De acordo com Sullivan e Feinn (2012), para um número de amostras suficientemente grande, um teste de significância irá quase sempre indicar uma diferença significativa, mesmo que a correlação seja extremamente pequena. Considerando que um total de 260811 amostras foram utilizadas para a execução do método SW, é possível que o teste tenha capturado dependências existentes, mas não relacionadas à resposta dinâmica do veículo, o que coaduna com os resultados apresentados para a análise de autocorrelação parcial, discutidos anteriormente nesta mesma Seção.

6.2.4 Detecção de Falhas - Parâmetros dos Experimentos

Com cada um dos modelos finais obtidos na etapa anterior, pode-se agora avaliar a estratégia de detecção como um todo. De maneira a verificar o comportamento do detector para diferentes configurações, os resultados serão apresentados de acordo com a adoção de diferentes valores para o limiar para o qual a falha é indicada, como visto na Seção 5.5:

$$\mathit{th} = \{0.06, 0.07, 0.08, 0.09\}\,.$$

O intervalo dos valores testados foi definido mediante experimentos preliminares, partindo dos valores médios de MSE encontrados na etapa de treinamento dos modelos preditivos, como visto na Seção 6.2.2, até valores que, de maneira geral, resultaram em um número de falhas detectadas de 0.

Uma segunda consideração é a de que, apesar de a capacidade de predição da rede NARX ser satisfatória de um modo geral, existem momentos pontuais em que a predição se distancia do real, mesmo em condições sem falha, levando a falsos positivos. Por este motivo, uma média móvel é utilizada (CAPRIGLIONE *et al.*, 2018), de maneira que a indicação de falha seja baseada não no resíduo instantâneo $\phi_i(k)$, mas sim na média dos resíduos dos últimos wd intervalos de tempo, como mostrado na Equação (26):

$$\phi_{m,i}(k) = \frac{1}{wd_i} \sum_{j=0}^{wd_i-1} \phi_i(k-j) \tag{26}$$

Em que $\phi_{m,i}$ representa o resíduo do i-ésimo atributo alvo após aplicação da média móvel e wd_i o tamanho da janela deslizante usada para o cálculo da média móvel. De maneira análoga ao limiar th, resultados serão apresentados para diferentes tamanhos da janela:

$$\mathit{wd} = \{7, 10, 13\}\,.$$

Neste trabalho, um detector é caracterizado pela combinação de um modelo de regresão, limiar th e tamanho da janela wd. Como temos 4 modelos de regressão (um para cada subconjunto de atributos selecionados), um detector pode assumir $4{\cdot}3{\cdot}4 = 48$ configurações distintas.

Os valores dos parâmetros acima foram escolhidos mediante experimentos prévios, para a definição de um intervalo pertinente para a presente aplicação.

Como visto na Seção 5.5, como a falha é indicada caso qualquer um dos resíduos $\phi_{m,1}$ ultrapasse determinado limiar, os resultados serão apresentados de maneira independente para GYROX, GYROY e GYROZ.

6.2.4.1 Parâmetros das Falhas Injetadas

Além dos parâmetros do detector em si, os parâmetros das falhas injetadas no conjunto de teste também serão variados, sendo possível, assim, avaliar o comportamento da estratégia na presença de diferentes tipos, magnitudes e duração das falhas. Desta maneira, cada tipo de falha, como visto na Seção 5.6, será injetada com diferentes durações:

$$d = \{10, 25, 50, 100\} = \{2s, 5s, 10s, 20s\}$$

e diferentes magnitudes:

1. Viés constante: $\delta = \{0.1, 0.5, 1.0\}$
2. Ganho constante: $\beta = \{1.1, 1.5, 2.0\}$
3. Drift: $c = \{0.01, 0.05, 0.1\}$
4. Stuck-at: $\alpha = \{0, max, last\}$

Desta maneira, deseja-se testar falhas de baixa, média e alta magnitudes para cada tipo de falha. Para a falha viés constante, comparando-se com a amplitude das leituras encontradas no conjunto de teste, os valores de δ adotados representam aproximadamente 4%, 19% e 38% para GYROX; 4%, 20% e 41% para GYROY; e 3%, 14% e 28% para GYROZ. Já as falhas de drift correspondem a 0.4%, 1.9% e 3.8% para GYROX; 0.4%, 2.0% e 4.1% para GYROY; e 0.3%, 1.4% e 2.8% para GYROZ. Para Ganho constante, os valores de β representam um acréscimo de 10%, 50% e 100% da leitura original, independente do alvo. A falha Stuck-at, diferentemente das anteriores, não é variada de acordo com a sua intensidade, mas sim simulando-se três condições diferentes: para $\alpha = 0$, o atributo alvo mantém o valor 0 por toda a duração da falha, enquanto que para $\alpha = max$, o valor é ajustado para o valor máximo da faixa de atuação do giroscópio. Já para $\alpha = last$, o atributo mantém o valor lido no instante em que a falha se inicia, até o término da falha.

Assim, temos 12 magnitudes distintas de falha, para 4 durações diferentes. Desta maneira, 48 cenários distintos são planejados. Em cada cenário, 100 falhas serão injetadas aleatoriamente no conjunto original de testes (sem falhas), sendo que todas as falhas dentro de cada cenário correspondem a uma combinação específica de magnitude e duração. Desta forma, temos um conjunto de dados com um total de 4800 falhas injetadas. Este conjunto de dados é, então, utilizado para avaliar o desempenho de cada um dos 48 detectores, variando-se o modelo de regressão e os valores th e wd, como visto anteriormente. Como este processo é repetido para cada atributo alvo (GYROX, GYROY e GYROZ) analisado, ao total 14400 falhas são injetadas e 144 detectores avaliados.

No ato de inserção das falhas, alguns critérios são observados. Um intervalo mínimo de 10 segundos é imposto entre o fim de uma falha e início de outra subsequente. Além disto, falhas não são inseridas nos primeiros e últimos 5 segundos do conjunto de teste. Por fim, como a estratégia se baseia no comportamento dinâmico do veículo, falhas não são inseridas em momentos em que o veículo se encontra ocioso, i.e., quando nenhum comando é enviado aos motores. Como exemplo, uma falha de

stuck-at com $\alpha = 0$ ou $\alpha = last$, quando inserida em condições de repouso, seriam indistinguíveis do comportamento sem falhas. Como exemplo, a estrutura utilizada para a configuração dos parâmetros de um cenário específico pode ser vista no Apêndice A, na página 128.

6.2.4.2 Métricas de Desempenho

Para a avaliação da capacidade de detecção, quatro métricas foram utilizadas: Precisão, Recall, medida F (ou F-measure) (BLAIR, 1979), e Latência de detecção.

A precisão nos dá a proporção entre o número de vezes que a estratégia foi capaz de detectar a falha corretamente e o número de vezes que a estratégia detectou uma falha, corretamente ou não, e sua expressão é dada por

$$\text{Precisão} = \frac{TP}{TP + FP},$$

em que TP e FP representam o número de verdadeiros positivos (falhas detectadas) e falsos positivos (condições normais detectadas como falha), respectivamente.

Já o Recall, também chamado de cobertura (SOARES, 2009), nos informa a proporção entre as falhas que foram detectadas e as falhas totais, detectadas ou não, e sua expressão é dada por

$$\text{Recall} = \frac{TP}{TP + FN},$$

em que FN representa o número de falsos negativos (falhas não detectadas).

As duas métricas podem ser combinadas em uma, por meio relação

$$F = 2 \cdot \frac{\text{Precisão} \cdot \text{Recall}}{\text{Precisão} + \text{Recall}},$$

que consiste na média harmônica entre Precisão e Recall.

Em um sistema de detecção de falhas, um bom equilíbrio entre precisão e recall deve ser alcançado, uma vez que uma baixa taxa de detecção implica em um sistema ineficiente, ao passo que uma alta taxa de falsos positivos leva o operador a ignorar o sistema (KHALASTCHI *et al.*, 2011). Dependendo da aplicação, um esforço adicional é realizado com a intenção de minimizar os falsos positivos (CHRISTENSEN *et al.*, 2008; GOLOMBEK, 2014), ou inclusive de zerá-los (ZHANG; POLYCARPOU; PARISINI, 2002), à custa de outros aspectos de desempenho, como latência e recall. Neste trabalho, procura-se minimizar os falsos positivos através de medidas como a adoção de média móvel para o cálculo dos resíduos. No entanto, a principal métrica utilizada para fins de comparação entre os detectores e métodos de seleção de atributos será a *F-measure*, dando portanto um peso equivalente às métricas de precisão e recall.

Antes de os resultados serem apresentados, é necessária uma discussão sobre a maneira como são contabilizadas as taxas de TP, FP e FN. A estratégia proposta

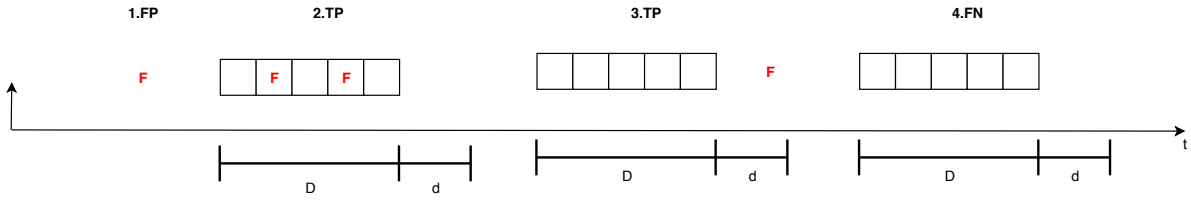


Figura 22 – A contabilização das taxas de FP, TP e FN.

utiliza valores de atributos passados, tanto na implementação do modelo preditivo quanto para o cálculo da média móvel dos resíduos. Assim, é possível que uma falha seja detectada mesmo após o seu término. Por este motivo, é definido uma janela de tempo na qual determinada falha é capaz de influenciar o detector, cujo tamanho consiste em $D+d$, sendo que D é a duração da própria falha, e d é a soma do tamanho da janela deslizante wd com a máxima unidade de atraso empregada na Rede NARX, como visto no Capítulo 5.

Para auxiliar a compreensão, a Figura 22 apresenta quatro situações hipotéticas de detecção. Na figura, os retângulos denotam os instantes em que a falha é inserida, e a letra F em vermelho representa os instantes em que a estratégia acusa um estado de falha. Assim, neste trecho hipotético, temos 3 falhas inseridas, 4 instantes em que o detector indica uma falha e 4 situações a serem analisadas. Na situação 1, a estratégia indica uma falha em um instante fora da zona de influência de quaisquer falhas inseridas, sendo, portanto, contabilizada como um falso positivo (FP). Já na segunda situação, ao menos uma indicação de falha está presente dentro do intervalo de duração da falha e, portanto, é contabilizada como um verdadeiro positivo (TP). O terceiro caso nos mostra a indicação de falha após a falha ter terminado, mas ainda dentro de sua zona de influência $D + d$, sendo assim também contabilizada como TP. O último caso representa uma situação em que uma falha é inserida e não detectada, sendo assim classificada como um falso negativo (FN).

Por fim, a rapidez da detecção também deve ser levada em conta, como visto no Capítulo 3. Por este motivo, resultados acerca da latência da detecção também serão apresentados. Neste trabalho, a latência de detecção foi definida como sendo o intervalo de tempo entre o início da falha e a primeira indicação de falha dentro de sua respectiva zona de influência. Voltando ao exemplo da Figura 22, a latência da falha detectada na situação 2 seria de 1 intervalo de tempo (ou 0.2s), e na situação 3 de 6 intervalos de tempo (ou 1.2s). Na situação 4, não há latência definida, uma vez que a falha não foi detectada.

6.2.5 Avaliação do Desempenho

Nesta seção, os resultados com relação à capacidade de detecção serão apresentados. Em um primeiro momento, os diferentes detectores serão avaliados de

Tabela 9 – F-measure para cada combinação de limiar *th* e janela *wd*, de acordo com cada subonjunto de atributos utilizado.

	th	0.06			0.07			0.08			0.09		
	wd	7	10	13	7	10	13	7	10	13	7	10	13
GYROX	SW	0.24	0.29	0.38	0.34	0.42	0.47	0.42	0.48	0.52	0.48	0.49	0.48
	RF	0.33	0.41	0.44	0.44	0.50	**0.56**	0.51	**0.56**	**0.56**	0.54	0.55	0.54
	CFS	0.25	0.31	0.39	0.37	0.46	0.52	0.47	0.52	0.54	0.49	0.50	0.50
	ALL	0.27	0.34	0.41	0.39	0.45	0.49	0.45	0.50	0.55	0.50	0.51	0.51
GYROY	SW	0.16	0.24	0.30	0.28	0.37	0.51	0.41	0.51	**0.62**	0.49	0.59	0.60
	RF	0.16	0.23	0.29	0.28	0.37	0.51	0.40	0.51	**0.62**	0.50	0.60	0.61
	CFS	0.15	0.22	0.28	0.27	0.36	0.48	0.39	0.50	0.61	0.48	0.59	0.60
	ALL	0.16	0.24	0.30	0.29	0.37	0.51	0.40	0.51	**0.62**	0.49	0.59	0.59
GYROZ	SW	0.37	0.43	0.53	0.47	0.51	0.54	0.50	0.52	0.48	0.49	0.45	0.42
	RF	0.45	0.52	0.60	0.54	**0.61**	**0.61**	0.58	0.58	0.55	0.57	0.52	0.48
	CFS	0.34	0.42	0.51	0.46	0.52	0.55	0.51	0.53	0.51	0.51	0.48	0.45
	ALL	0.33	0.43	0.51	0.47	0.51	0.52	0.49	0.50	0.46	0.47	0.43	0.41

acordo com as métricas de precisão/recall, F-measure e latência, sem realizar a distinção entre os vários tipos de falhas injetadas. Com base no F-measure, o detector que obteve melhor desempenho será avaliado em maiores detalhes, analisando o seu comportamento de acordo com a duração, tipo e magnitude das falhas injetadas. Este procedimento será realizado para cada um dos atributos alvo.

Na Tabela 9, é apresentada a F-measure para cada combinação de limiar (*th*) e tamanho da média móvel (*wd*), de acordo com o modelo obtido para cada subconjunto de atributos selecionado. Percebe-se que, de uma maneira, geral, valores da F-measure similares foram encontradas para os três alvos, com as melhores combinações apresentando resultados de 0.56, 0.62 e 0.61, para GYROX, GYROY, GYROZ, respectivamente. Considerando as questões de controle e manobrabilidade do OpenROV, em conjunto com os resultados passados apresentados, poderia se esperar um desempenho superior para o alvo GYROZ, em relação aos alvos restantes. Outros fatores podem ter contribuído para a equiparação dos desempenhos, como a importância relativa de cada variável de entrada na rede NARX para a predição e o impacto da interação entre as falhas inseridas com cada atributo alvo.

Já com relação ao subconjunto de atributos, percebe-se que o método RF apresentou melhores resultados tanto para GYROX quanto para GYROZ. Para GYROY, no entanto, o método de seleção de atributos não apresentou grande impacto no desempenho, com valores similares entre si. Neste caso, não foi possível estabelecer uma superioridade clara de um método com relação aos restantes.

Na Tabela 10, os valores de precisão e recall são reportados, permitindo uma maior granularidade nas análises. De maneira geral, foram encontrados valores elevados de precisão, com várias combinações chegando ao valor máximo de 1. Isto significa um número baixo de falsos positivos, chegando a 0 para GYROY e GYROZ.

Já os resultados para o recall não foram tão expressivos quanto para a precisão,

Tabela 10 – Precisão/recall para cada combinação de limiar *th* e janela *wd*, de acordo com cada subconjunto de atributos utilizado.

	th	0.06			0.07			0.08			0.09		
	wd	7	10	13	7	10	13	7	10	13	7	10	13
GYROX	SW	.15/.57	.20/.53	.30/.51	.25/.53	.37/.49	.47/.47	.38/.49	.52/.44	.66/.43	.55/.42	.65/.39	.70/.37
	RF	.23/.58	.32/.54	.39/.52	.37/.54	.49/.50	.67/.48	.53/.50	.67/.48	.73/.46	.66/.47	.72/.44	.73/.43
	CFS	.16/.57	.22/.53	.31/.50	.28/.53	.43/.49	.59/.47	.45/.49	.60/.45	.72/.43	.58/.43	.69/.40	.72/.38
	ALL	.17/.57	.25/.53	.35/.50	.31/.53	.41/.49	.51/.48	.43/.48	.55/.46	.72/.44	.56/.44	.66/.41	.71/.39
GYROY	SW	.10/.55	.16/.52	.22/.49	.19/.51	.31/.48	.57/.46	.35/.48	.58/.46	1.0/.45	.53/.46	.89/.44	1.0/.43
	RF	.09/.56	.15/.52	.21/.50	.19/.51	.30/.49	.55/.47	.35/.49	.58/.46	.98/.45	.54/.47	.91/.45	1.0/.44
	CFS	.09/.55	.14/.52	.20/.49	.18/.51	.29/.48	.50/.47	.33/.48	.55/.46	.93/.45	.49/.46	.88/.44	1.0/.43
	ALL	.10/.55	.16/.51	.21/.49	.20/.51	.31/.48	.56/.46	.35/.48	.57/.46	1.0/.45	.53/.46	.92/.44	1.0/.42
GYROZ	SW	.28/.54	.39/.49	.63/.46	.46/.48	.66/.42	.88/.39	.62/.42	1.0/.35	1.0/.32	.85/.35	1.0/.29	1.0/.26
	RF	.37/.57	.52/.53	.76/.50	.57/.52	.86/.47	.98/.44	.77/.47	1.0/.41	1.0/.38	.98/.40	1.0/.35	1.0/.31
	CFS	.24/.55	.37/.50	.56/.47	.44/.49	.65/.44	.90/.40	.64/.43	.93/.37	1.0/.34	.83/.37	1.0/.32	1.0/.29
	ALL	.24/.53	.38/.49	.58/.45	.46/.47	.67/.41	.87/.37	.64/.40	1.0/.33	1.0/.30	.80/.33	1.0/.28	1.0/.26

com valores na faixa de 15% a 58%. Para as melhores combinações do detector, de acordo com a F-measure, 46%, 45% e 44% das falhas inseridas foram recuperadas para GYROX, GYROY e GYROZ respectivamente. Percebe-se, também, que existe sempre um balanço entre precisão e recall. À medida que *th* e *wd* aumentam, os valores de precisão sobem, à custa de uma queda nas medidas de recall.

Por fim, a média e desvio-padrão (em número de intevalos de tempo) das latências são reportados na Tabela 11. Apesar de haver um aumento da latência à medida que *th* e *wd* aumentam, a diferença não é expressiva. Percebe-se que os maiores valores encontrados apresentam média e desvio padrão de 18 e 20 intervalos de tempo (3.6 e 4 segundos), respectivamente, enquanto que a duração média das falhas inseridas é de 9.25 segundos. Estes resultados podem ser considerados aceitáveis, ou não, de acordo com a aplicação em questão. É possível, por exemplo, que para determinadas classes de veículos autônomos, requisitos mais severos quanto à rapidez de detecção sejam encontrados. No entanto, para a presente aplicação, em que a detecção de falhas é realizada com a intenção de auxiliar o piloto do veículo no processo de tomada de decisão, os valores encontrados de latência foram considerados aceitáveis.

6.2.5.1 O desempenho de acordo com a falha

Para os resultados apresentados a seguir, faz-se necessário definir uma configuração específica do detector para cada atributo alvo. Os critérios adotados para esta escolha foram baseados no F-measure, como mostrado na Tabela 9. Como, para todos os casos, o método RF figurou entre os valores mais elevados, optou-se por avaliar detectores que utilizem este método de seleção. Em caso de empate, optou-se por aquele com maior valor de precisão, com a intenção de minimizar-se os falsos positivos. Desta maneira, os seguintes detectores foram selecionados para análise mais aprofundada: GYROX - Método RF, *th* 0.08 e *wd* 13, GYROY - Método RF, *th* 0.08

Tabela 11 – Latências (média ± desvio-padrão) para cada combinação de limiar *th* e janela *wd*, de acordo com cada subconjunto de atributos utilizado.

	th	0.06			0.07			0.08			0.09		
	wd	7	10	13	7	10	13	7	10	13	7	10	13
GYROX	SW	10±15	11±15	11±14	10±15	11±15	12±14	11±16	12±15	12±14	12±17	12±16	13±16
	RF	9±13	9±13	10±12	9±14	10±13	11±13	10±13	11±14	12±13	10±13	11±13	12±13
	CFS	9±14	10±14	11±13	10±14	11±13	11±13	11±14	11±14	12±14	12±16	12±16	13±16
	ALL	9±14	10±12	11±13	10±14	11±13	11±13	10±14	11±13	12±13	12±15	13±15	13±15
GYROY	SW	8±14	9±13	10±13	9±14	10±13	10±13	9±13	10±13	11±13	10±14	11±13	12±13
	RF	9±14	9±12	9±12	8±13	9±12	10±12	9±12	10±13	11±12	10±13	10±13	11±12
	CFS	8±14	9±13	9±12	9±13	9±12	10±13	9±13	10±13	11±13	10±14	11±13	11±13
	ALL	9±14	9±13	10±12	9±13	10±13	10±12	9±13	10±13	11±13	10±14	11±13	11±13
GYROZ	SW	13±19	15±20	16±20	15±21	16±20	17±20	15±20	17±19	17±20	16±20	16±18	16±17
	RF	12±18	14±19	15±19	14±20	15±20	17±20	15±20	17±20	17±20	15±19	16±19	17±18
	CFS	13±19	14±19	15±19	14±20	15±19	16±19	14±19	17±20	18±20	16±20	17±20	17±18
	ALL	13±20	15±21	17±21	15±22	16±20	17±19	16±20	16±18	16±17	16±18	16±18	17±17

e *wd* 13, GYROZ - Método RF, *th* 0.07 e *wd* 13.

Na Figura 23, é apresentado o F-measure de acordo com a duração das falhas, com colunas segmentadas de acordo com o atributo alvo. Percebe-se que, de maneira geral, falhas de maiores durações são mais facilmente detectáveis. O impacto da duração é mais acentuado para GYROZ, com a F-measure apresentando um aumento constante e atingindo um valor superior a 0.7 para a duração de 20s. Já para GYROX e GYROY, as falhas são mais facilmente detectáveis até a duração de 10s, atingindo valores próximos a 0.6 para GYROX e 0.7 para GYROZ, permanecendo aproximadamente constantes de 10s para 20s.

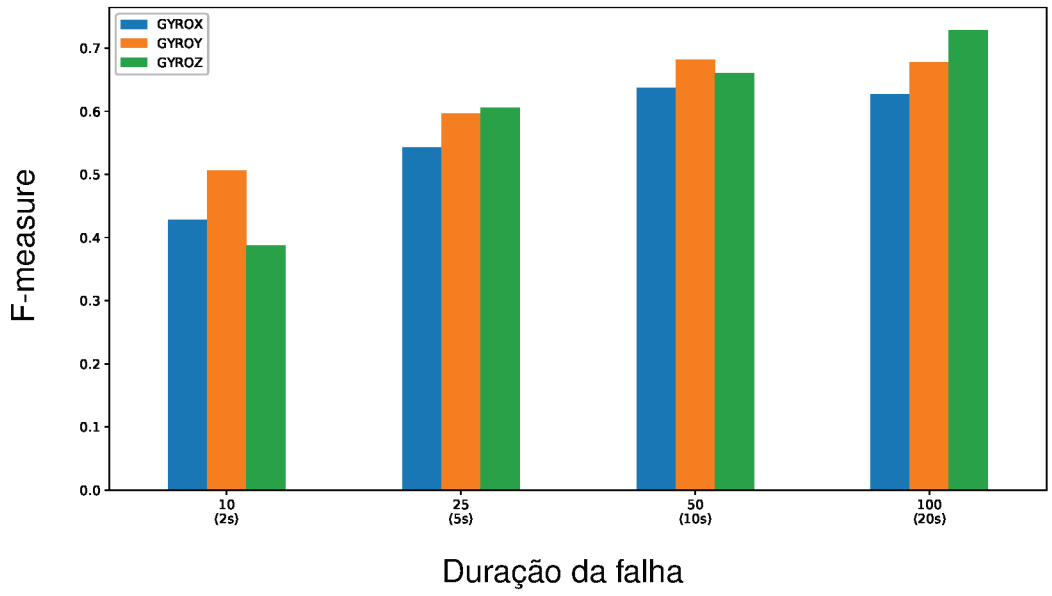


Figura 23 – F-measure de acordo com a duração das falhas, para GYROX, GYROY e GYROZ.

As Figuras 24, 25 e 26 relacionam o F-measure de acordo com o tipo de falha e, para cada tipo de falha, suas respectivas magnitudes, para as diferentes durações.

Os resultados nos mostram que para diversos tipos de falha o desempenho da estratégia foi ideal, ou próximo disto, independente da duração da falha, apresentando F-measure superior a 0.9. Estes são os casos das falhas de viés ($\delta = 1.0$), drift ($c = 0.1$), stuck-at ($\alpha = max$) para GYROX; das falhas de viés ($\delta = 0.5e1.0$), drift ($c = 0.1$), stuck-at ($\alpha = max$) para GYROY; e das falhas de viés ($\delta = 1.0$) e stuck-at ($\alpha = max$) para GYROZ.

Em outros casos, o desempenho da estratégia foi ideal, ou próximo disto, após determinada duração, a partir da qual os valores de F-measure encontrados superam 0.9. Estes são os casos das falhas de viés ($\delta = 0.5$), drift ($c = 0.01$ e 0.05) para GYROX; das falhas de drift ($c = 0.01$ e 0.05) para GYROY; e das falhas de drift ($c = 0.01, 0.05$ e 0.1) para GYROZ. Neste grupo, nota-se um comportamento similar entre as falhas de drift: quanto maior a magnitude, as falhas de menores duração são mais facilmente detectadas, como era de se esperar, devido à natureza da falha.

Percebe-se, também, que certas condições passaram despercebidas pela estratégia, com F-measure próximo a 0, como é o caso das falhas de viés ($\delta = 0.1$), ganho ($\beta = 1.1$), stuck-at ($\alpha = 0$) para GYROX; falhas de viés ($\delta = 0.1$), ganho ($\beta = 1.1$), stuck-at ($\alpha = max$ e $last$) para GYROY; e falhas de viés ($\delta = 0.1$), ganho ($\beta = 1.1$) para GYROZ.

De maneira geral, as falhas de ganho constante se provaram as mais desafiadoras de serem detectadas, com desempenho aceitável apenas para determinadas magnitudes para GYROZ. As falhas do tipo stuck-at também apresentou desempenho variável, de acordo o valor de δ adotado. A magnitude se provou o fator determinante para as falhas de viés, enquanto que a duração é um fator crucial para a detecção das falhas de drift.

Por fim, a latência também é analisada, em termos de média e desvio-padrão, de acordo com o tipo, magnitude e duração das falhas inseridas, como apresentado nas Figuras 27, 28 e 29. Para os casos em que nenhuma falha é detectada, a latência não é definida e, portanto, são omitidas nos gráficos. Além disto, também são omitidos os pontos para o qual o número de falhas detectadas é pequeno (≤ 3), por considerar que as medidas de média e desvio padrão seriam pouco representativas. Nas ocasiões em que determinados tipos de falhas não são detectados para duração alguma, a falha inteira é definida como ND (não detectada).

Percebe-se que para todos os casos, a média da latência se encontra próxima ou abaixo da própria duração da falha. Isto significa que, na média, as falhas são detectadas ainda antes de seu término. De maneira geral, observa-se dois comportamentos distintos quanto à latência: falhas que apresentam média e desvio-padrão aproximadamente constantes, com relação à sua duração, como nas falhas de viés

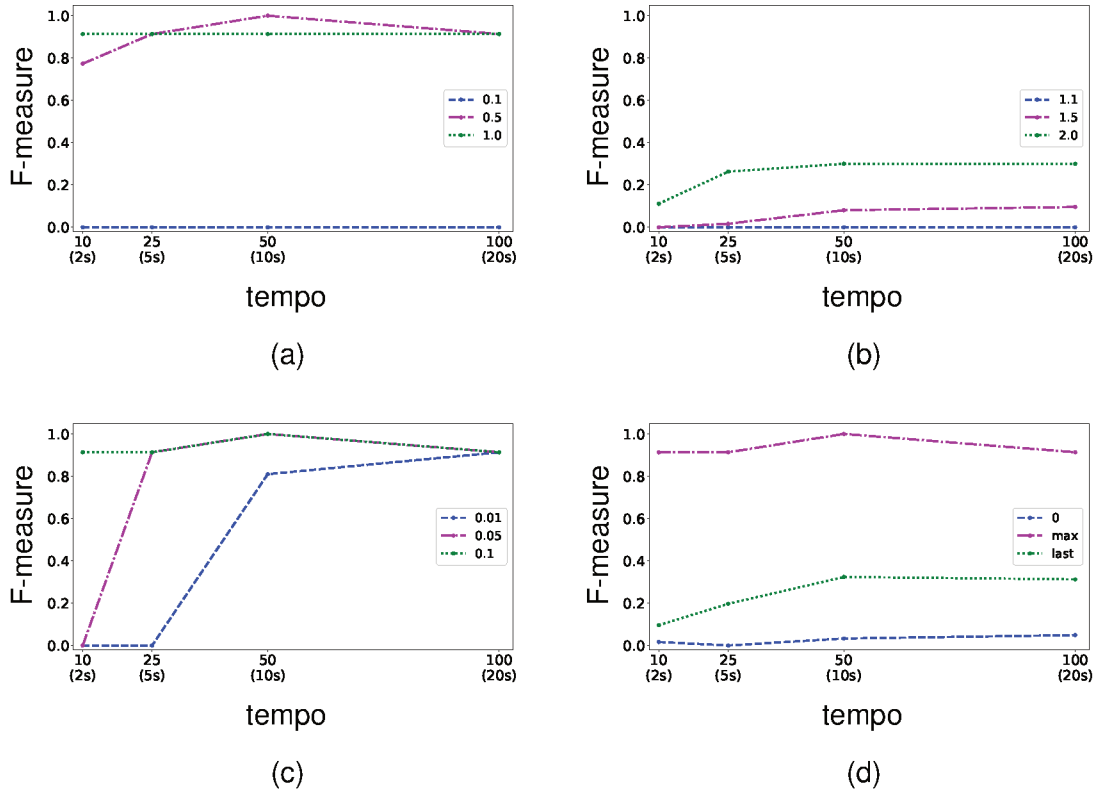


Figura 24 – F-measure de acordo com o tipo, magnitude e duração de falha para o atributo GYROX.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

e drift para GYROX e GYROY e de stuck-at para todos os alvos; e falhas cuja média e dispersão aumentam conforme a duração aumenta, como as falhas de ganho para todos os alvos e determinadas falhas de stuck-at, para GYROX e GYROZ.

Dentre os casos reportados, o valor mais alto encontrado refere-se à falha de drift, para a duração de 20s e $c = 0.01$, para o atributo GYROZ. Neste ponto, a latência, considerando-se o desvio padrão, ultrapassa a própria duração da falha. Isto se deve à estrutura de malha aberta da rede NARX utilizada, o que significa que as falhas inseridas são introduzidas também na entrada da rede. Por conta disto, a rede se ajusta aos pequenos incrementos graduais feitos no atributo alvo. A falha acaba por ser detectada somente no momento imediatamente após o seu término, em que o acúmulo de todos estes incrementos leva a uma grande diferença dos valores previsto e real, já que este último passa a ser o atributo sem falhas. Nota-se, portanto, que para a estratégia proposta, a natureza intermitente da falha pode se mostrar vantajosa para determinados casos.

Na Figura 30, um exemplo da evolução temporal para o atributo GYROZ (após normalização) é apresentado, quando uma falha *stuck-at* do tipo *last* é inserida, com uma duração de 50 intervalos de tempo (10s). Um trecho de aproximadamente 175 intervalos de tempo (35s) é apresentado, retirado do conjunto de teste. A curva contí-

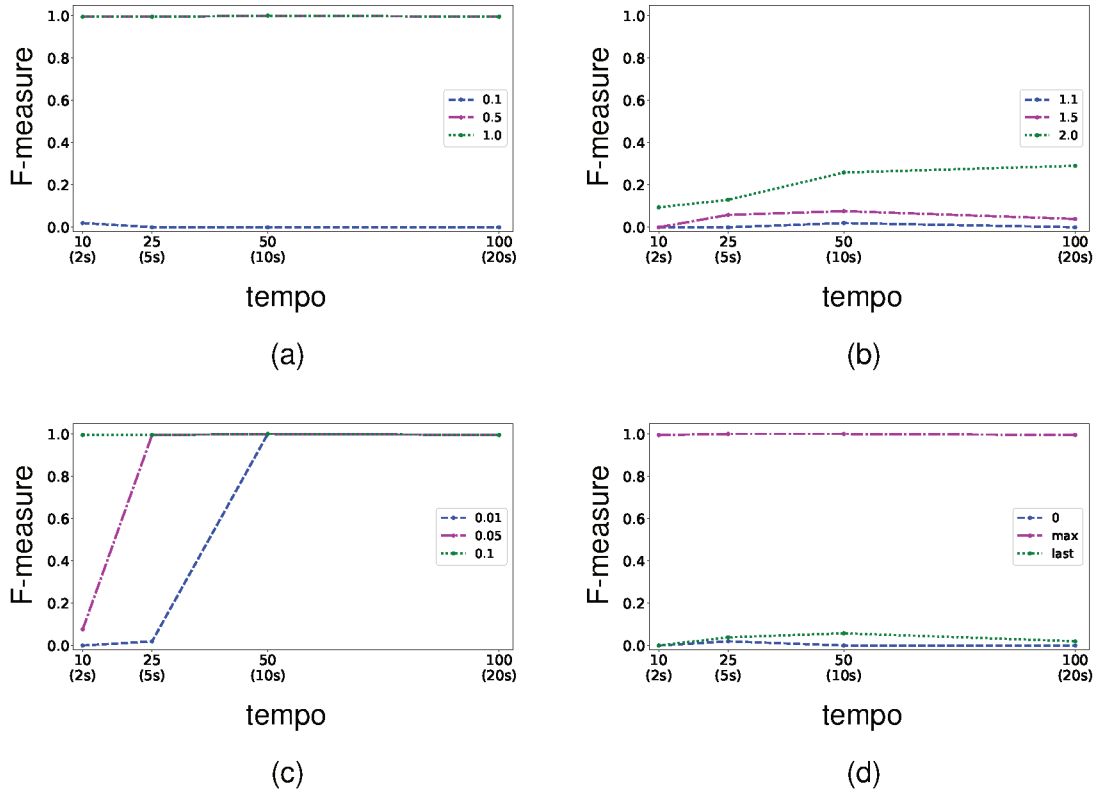


Figura 25 – F-measure de acordo com o tipo, magnitude e duração de falha para o atributo GYROY.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

nua de cor preta representa o valor real de GYROZ, após inserção da falha, enquanto que a curva tracejada de cor azul representa os valores estimados pelo modelo NARX, utilizando o método de seleção RF. Além disto, na parte inferior do gráfico, o período de duração da falha inserida é denotado, além dos períodos para o qual uma indicação de falha é acusada pela estratégia, de acordo com a adoção de diferentes tamanhos de janela deslizante - $wd = \{7, 10, 13\}$, utilizando um valor fixo de limiar - $th = 0.07$.

Através da Figura 30, importantes aspectos da estratégia podem ser visualizados. Para os intervalos isentos de falha, a proximidade entre as duas curvas demonstra que o modelo preditivo foi capaz de estimar o atributo alvo de maneira satisfatória. Já o intervalo com a presença da falha demonstra a capacidade de detecção da estratégia. Apesar de a falha se estender por uma duração de 10s, a falha começa a ser detectada somente quando os valores estimados se distanciam dos valores reais, indicando um período de variação da velocidade. Caso a falha fosse inserida em um período em que a velocidade se mantivesse constante, nenhuma falha seria detectada.

Apesar de o modelo ser capaz de aproximar GYROZ de maneira adequada para a maior parte do tempo, percebe-se que há condições pontuais em que as duas curvas se distanciam, como no instante próximo a $t = 150$. Estes casos motivam o uso de uma abordagem baseada em médias móveis, a fim de evitar uma detecção errônea.

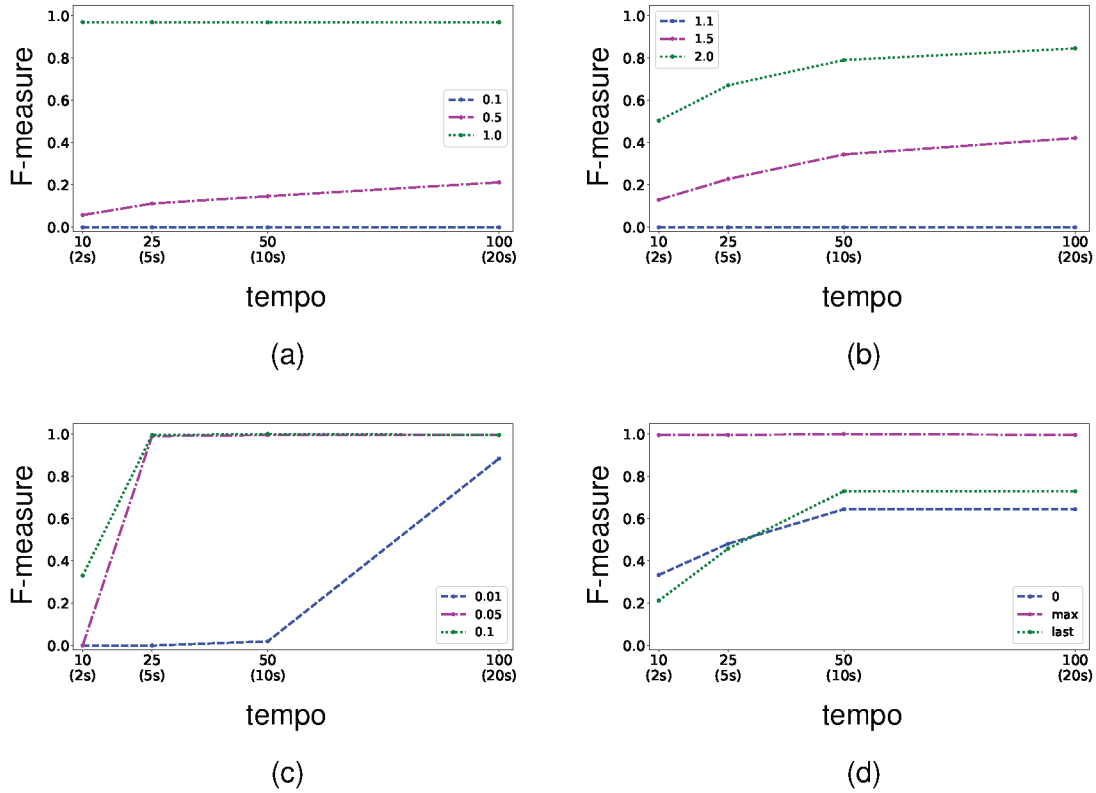


Figura 26 – F-measure de acordo com o tipo, magnitude e duração de falha para o atributo GYROZ.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

Além disto, os períodos de detecção auxiliam na visualização do impacto dos tamanhos da janela deslizante na latência da detecção. A média móvel atenua os resíduos resultantes, de maneira que uma sequência maior de resíduos anômalos é necessária para a detecção à medida em que wd cresce, atrasando o início da detecção e retardando o seu fim.

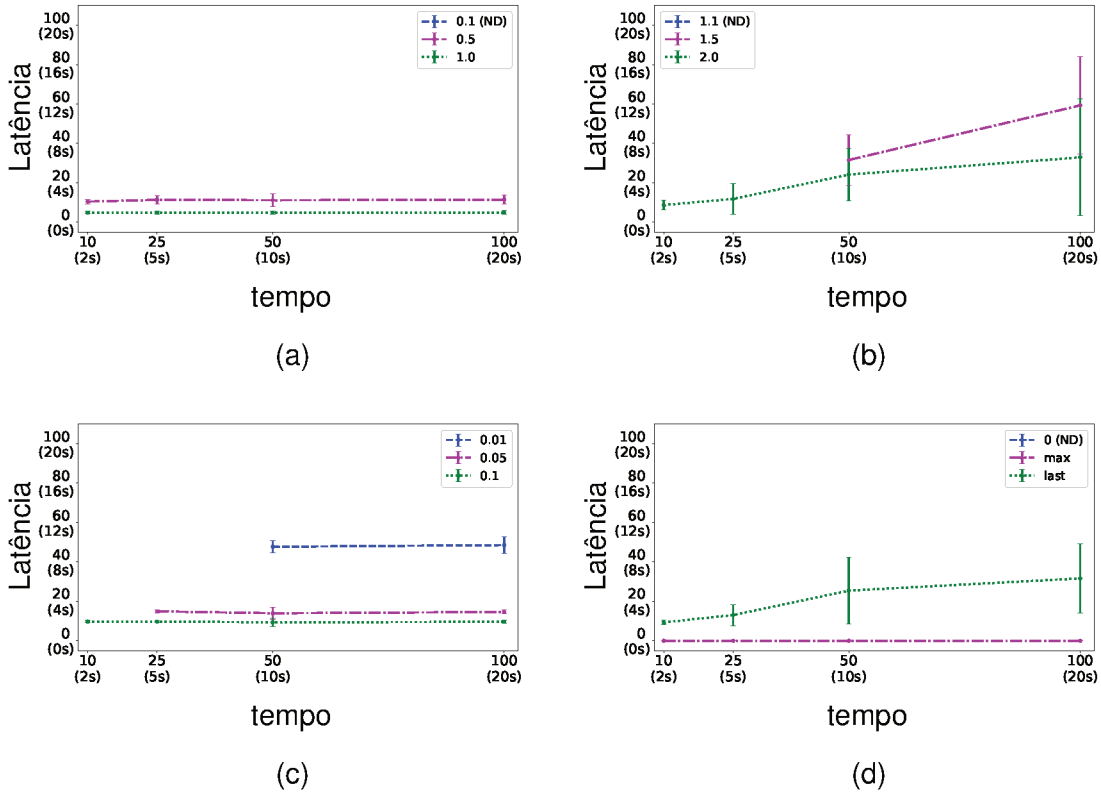


Figura 27 – Latência de detecção (média $\pm$ desvio padrão) de acordo com o tipo, magnitude e duração de falha para o atributo GYROX.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

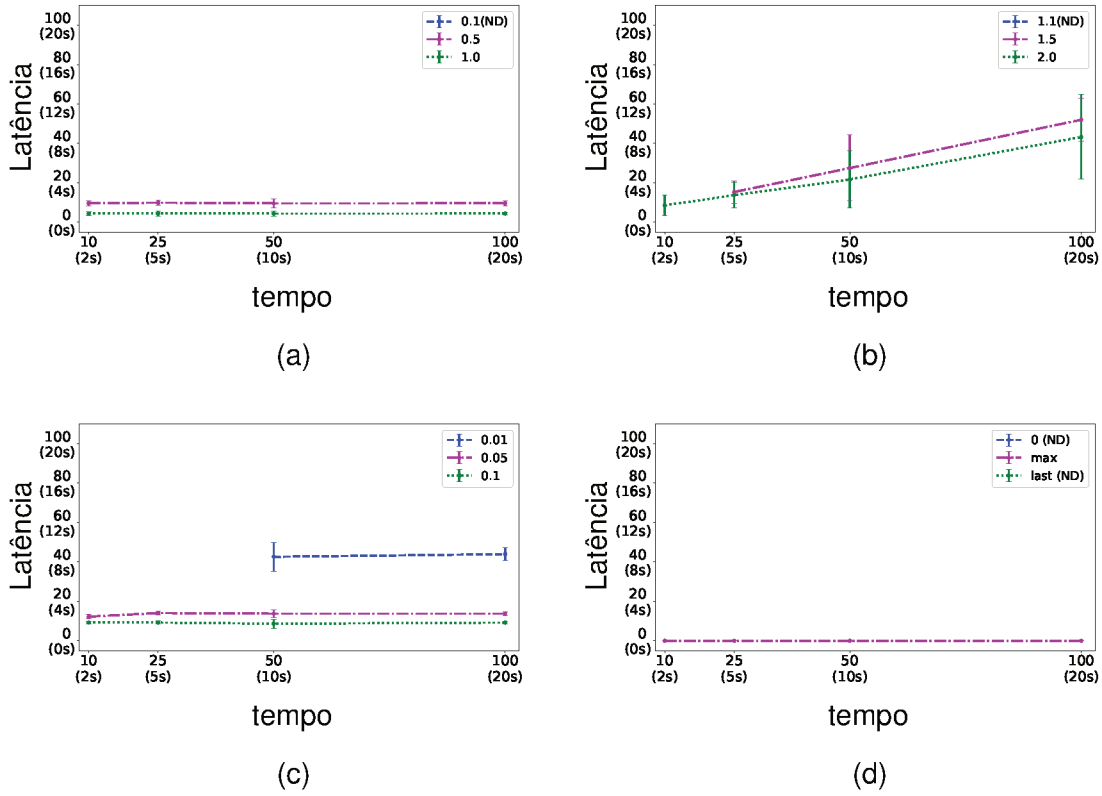


Figura 28 – Latência de detecção (média $\pm$ desvio padrão) de acordo com o tipo, magnitude e duração de falha para o atributo GYROY.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

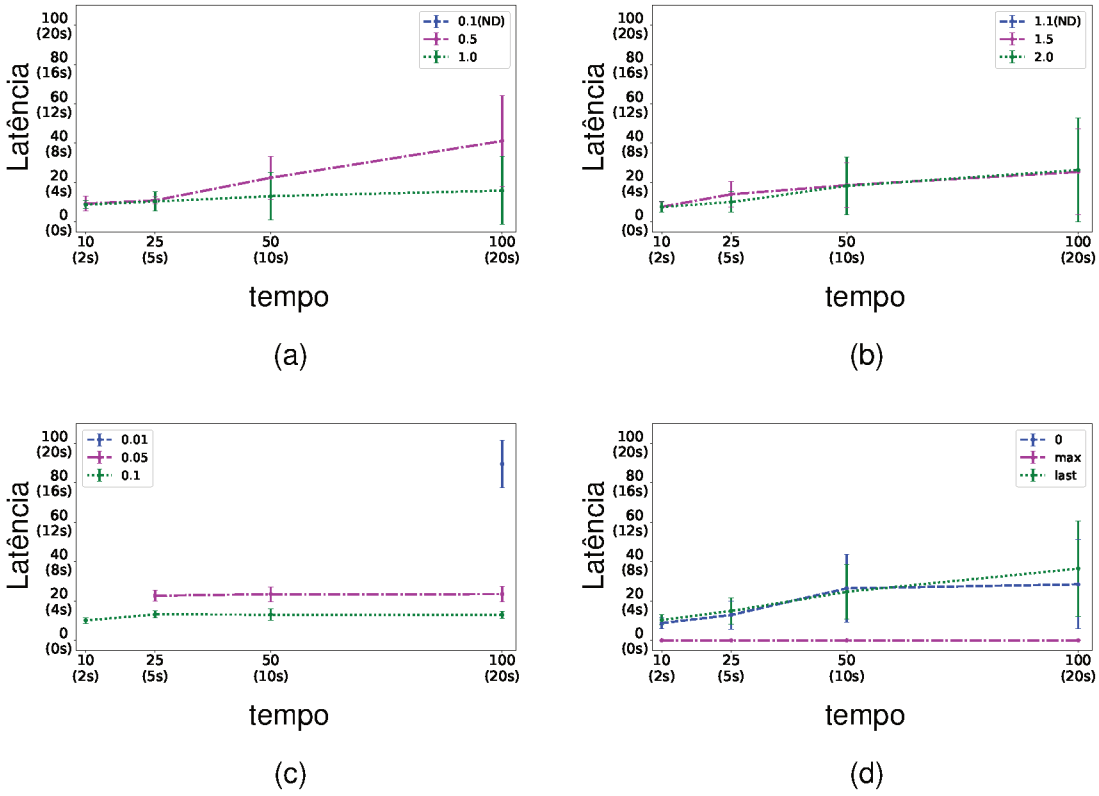


Figura 29 – Latência de detecção (média $\pm$ desvio padrão) de acordo com o tipo, magnitude e duração de falha para o atributo GYROZ.(a)- Viés constante ,b) - Ganho constante, (c) - Drift, (d) - Stuck-at.

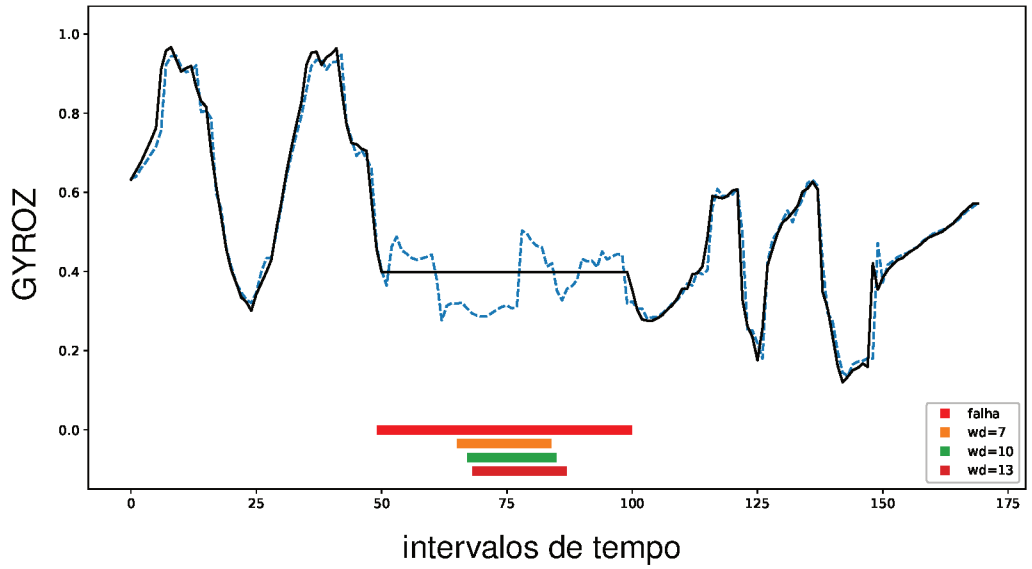


Figura 30 – Evolução temporal de GYROZ, com períodos de inserção e detecção de falha, para diferentes tamanhos de janelas deslizantes. Curvas de linha contínua - valores reais (após injeção da falha) e linha tracejada - valores estimados.

7 INTEGRAÇÃO COM A PLATAFORMA IOT

Uma vez treinados os modelos e definidos todos os parâmetros do detector, a estratégia de detecção pode ser integrada em uma infraestrutura baseada em Computação em Nuvem (*Cloud Computing*), ou seja, em um modelo que habilita o acesso de rede a um conjunto compartilhado de recursos computacionais (como armazenamento, aplicações e serviços) de maneira ubíqua e conveniente (MELL; GRANCE *et al.*, 2011).

A plataforma de sensoriamento distribuído do Laboratório de Integração Software/Hardware (LISHA) (LISHA, 2019) foi a adotada para a realização deste projeto. No servidor, a comunicação com agentes externos (gateway e interface gráfica) é realizada por meio da interface REST API, pela qual instruções de inserção e requisição de dados são realizadas por clientes. O padrão de representação de dados utilizado pela plataforma é o Smartdata, que é uma API (*Application Programming Interface*) de alto nível para redes de sensores que tem o objetivo de facilitar o desenvolvimento de aplicações de sensoriamento. Além do valor da medição em si, um SmartData contém metainformação relativa a aspectos de semântica, localização espacial, temporização e confiabilidade (RESNER *et al.*, 2018).

7.1 ESTRUTURA DOS DADOS

Os seguintes tipos de requisições estão disponíveis ao cliente para a comunicação com o servidor: *attach, create, put* e *get*. As requisições *attach* e *create* são responsáveis pela criação de séries temporais:

Código 7.1 – Representação das séries temporais no formato Series.

```
{
    "series" : Object
    {
        "version" : string
        "unit" : unsigned int
        "x" : int
        "y" : int
        "z" : int
        "r" : unsigned int
        "t0" : unsigned int
        "t1" : unsigned int
        "dev" : unsigned int
        "signature" : unsigned int
        "workflow": unsigned int
```

```
    }
    "credentials" : Object
    {
        "domain" : string
        "username" : string
        "password" : string
    }
}
```

Desta maneira, séries temporais são definidas por sua:

1. Localização espacial: O campo r define a esfera de abrangência da série, cujo centro é definido pelas coordenadas espaciais geocêntricas x, y e z (CLYNCH, 2006).
2. Localização temporal: o período de tempo delimitado pelos timestamps inicial e final t_0 e t_1, respectivamente.
3. Unidade: A unidade SI (MECHTLY, 1973) (ou digital) dos dados que compõem a série, como, por exemplo, unidade de corrente, tensão ou temperatura.

Uma vez criadas as séries, os dados enviados subsequentemente serão alocados na base de dados do servidor de acordo com suas respectivas séries, conforme suas coordenadas espaço-temporais e unidade. Os dados sensoriais, por sua vez, são representados no formato Smartdata, utilizando também o padrão JSON, conforme a seguir:

Código 7.2 – Representação dos atributos no formato SmartData.

```
{
    "smartdata" : Array
    [
        {
            "version" : string
            "unit" : unsigned int
            "value" : double
            "error" : unsigned int
            "confidence" : unsigned int
            "x" : int
            "y" : int
            "z" : int
            "t" : unsigned int
```

```
            "dev" : unsigned int
            "signature" : unsigned int
            "workflow": unsigned int
        }
    ]
    "credentials" : Object
    {
        "domain" : string
        "username" : string
        "password" : string
    }
}
```

O campo *value* especifica o valor do atributo, como a medição de um sensor ou comando enviado a um atuador. A localização temporal, por sua vez, é realizada através do campo *t*, representando o timestamp do momento de coleta do atributo. O campo dev é utilizado para desambiguação nos casos em que leituras de fontes diversas apresentem uma mesma coordenada espaço-temporal e unidade como, por exemplo, um acelerômetro que é capaz de fornecer leituras de aceleração de 3 eixos distintos. Os dados enviados são direcionados a um domínio específico, cujo acesso pode ser realizado pelos campos *username* e *password*. Alternativamente, o acesso ao domínio pode ser realizado diretamente via certificação digital, e neste caso o objeto *credentials* não é utilizado nas requisições.

7.1.1 O campo Workflow

Por meio do campo *workflow*, a plataforma IoT fornece uma maneira de estabelecer um determinado fluxo de trabalho a ser executado dentro do servidor. No ato do recebimento do smartdata, o servidor verifica se há um ou mais fluxos de trabalho associado à sua série temporal, definido no ato de criação da série. Caso afirmativo, de maneira prévia à inserção do smartdata na base de dados, o dado é encaminhado para o script associado. Após a execução do fluxo de trabalho, o dado é então inserido na base de dados. Desta maneira, o presente trabalho pretende implementar o detector de falhas como um script que é invocado a cada envio dos dados ao servidor.

7.2 ARQUITETURA DE SOFTWARE

Nesta seção, pretende-se abordar a arquitetura de software do sistema proposto, apresentando o comportamento geral do sistema, bem como seu comportamento interno e interações entre componentes de software

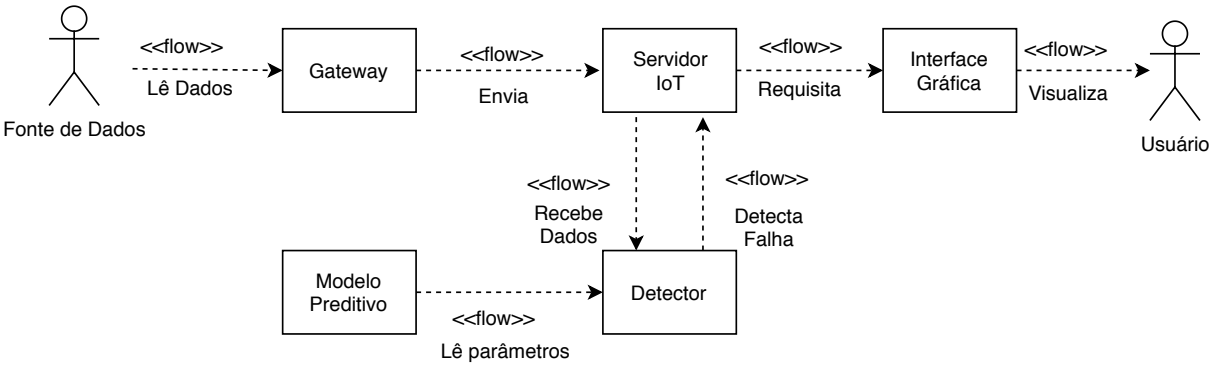


Figura 31 – Diagrama de fluxo de informação da arquitetura proposta.

Apenas a fase de detecção do sistema será implementada na plataforma IoT, i.e., o processo de treinamento do modelo preditivo usado para detectar falhas é realizado de maneira offline, envolvendo as etapas de seleção de atributos, definição da topologia, treinamento e validação. Assim, os parâmetros dos modelos, como pesos dos neurônios, topologia da rede e mapeamento dos atributos selecionados estarão disponíveis para o detector durante a operação da estratégia.

7.2.1 Descrição e Relações entre Componentes

O comportamento geral e a troca de informações entre os componentes do sistema proposto pode ser visto na Figura 31. O gateway é o componente responsável pelo interfaceamento entre o ROV e o sistema, coletando informações fornecidas pelo veículo e enviando-as para o servidor, já em um formato de smartdata. Neste trabalho o gateway acessa as informações sensoriais do ROV a partir de arquivos JSON, obtidos após a operação do ROV. Em futuras expansões deste trabalho, pretende-se substituir a fonte de dados pelo ROV em si, que fornece informações ao gateway diretamente, durante a própria operação do veículo.

O servidor, então, recebe os dados sensoriais emitidos pelo gateway e os retransmite ao detector, de acordo com o campo *workflow* da respectiva série temporal. Os dados enviados são direcionados a um domínio específico, que foi criado para este projeto (*Underwater ROV*), cujo controle de acesso é realizado via certificação digital. O detector, por sua vez, utiliza os parâmetros dos modelos preditivos (um para cada atributo alvo) já treinados para realizar o processo de detecção de falhas. Uma vez gerado o diagnóstico, o detector encaminha o resultado para o servidor, que então armazena o valor em uma base de dados, em conjunto com os dados sensoriais recebidos pelo gateway. Finalmente o usuário, que pode ser o próprio operador ou outro agente externo, pode visualizar o resultado da detecção por meio da interface gráfica.

O comportamento do gateway pode ser visto no diagrama de máquina de estados apresentado na Figura 32. Após a sinalização de início de detecção enviada pelo usuário, o gateway passa ao estado de criação de séries temporais para cada tipo

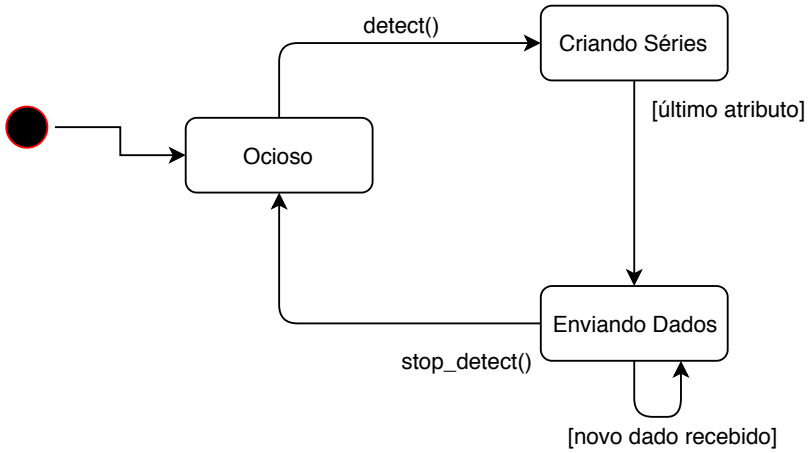


Figura 32 – Diagrama de máquina de estados do componente gateway.

de atributo fornecido pelo veículo. Como o ROV não tem a capacidade de localização espacial por operar abaixo d'água, as coordenadas espaciais x, y, z da série (e dos dados subsequentes) serão atribuídas de acordo com a posição do notebook utilizado para a operação do veículo. Assim, o campo r da série delimita a esfera de alcance do ROV centrada no notebook, de acordo com o comprimento do cabo umbilical (100m). Como, em um mesmo intervalo de tempo, a mesma localização espacial e temporal é atribuída a todos os dados sensoriais, a desambiguação para o caso em que diferentes atributos possuem a mesma unidade é realizada pelo campo dev, cujo número inteiro define o sensor que gera o smartdata em questão.

Uma vez criado as séries temporais para todos os atributos, o gateway passa ao estado de envio de dados.

A sequência de atividades relativas ao envio dos dados para detecção pode ser vista no diagrama de atividade apresentado na Figura 33, distinguindo as atividades de acordo com o componente que as executa. O processo se inicia pelo gateway, que coleta os dados do ROV de forma contínua, até que a parada de detecção seja sinalizada pelo usuário. Após a conversão dos dados coletados para um formato de smartdata, os dados são enviados e encaminhados pelo servidor ao componente detector.

Para a execução do detector, os frameworks dos pacotes de inteligência artificial utilizados (Keras (CHOLLET *et al.*, 2018), Theano (BERGSTRA *et al.*, 2010)) devem ser inicializados previamente. Para evitar que o processo de inicialização seja feito a cada vez que o script do workflow é chamado pelo servidor, o detector é implementado como um conjunto de dois componentes: o script de workflow e um processo daemon. Caso um dado seja encaminhado ao script de workflow e o processo daemon já não esteja em execução, o mesmo é criado. Desta forma, o script encaminha os dados para o daemon, que por sua vez retorna o diagnóstico ao script.

Como visto na Seção 4.3.3, o modelo preditivo utilizado na detecção de falhas

exige os valores de momentos passados de diversos atributos. Portanto, a detecção é efetivamente realizada somente quando uma cache local do detector é populada pela informação histórica necessária. Uma vez disponível os dados necessários, uma etapa prévia de normalização é necessária, uma vez que o modelo espera um conjunto de entradas normalizado. Assim que o resíduo é calculado para cada atributo-alvo, o vetor de resíduos é encaminhado de volta ao servidor para inserção na base de dados.

7.3 O DETECTOR INTEGRADO À PLATAFORMA IOT

Como exemplo, o processo de detecção de falhas, apenas para GYROZ, foi implementado e integrado à plataforma IoT, utilizando os modelos treinados e parâmetros definidos de acordo com as Seções prévias deste trabalho.

Para ilustrar a implementação, um trecho do conjunto de teste de aproximadamente 160 segundos foi utilizado. Neste trecho, 3 falhas de viés constante ($\delta = 0.1$ e duração de 2s) foram inseridas, de acordo com o procedimento explicado em 6.2.4.1. Na Figura 34a, duas curvas distintas podem ser vistas: o atributo real *GYROZ*, coletado a partir do ROV, é representado na curva azul. O valor estimado a partir do modelo de regressão é representado em laranja. O eixo vertical consiste no atributo alvo após normalização, enquanto que o eixo horizontal representa o número de intervalos de tempo decorridos. Os pontos em vermelho na base da Figura denotam os momentos em que as falhas foram inseridas.

Na Figura 34b, é apresentado o processo de geração de resíduos, já integrado à plataforma IoT. Neste caso, o detector, presente no servidor, fornece como resultado o resíduo referente ao *GYROZ*, após aplicação da média móvel. O detector foi desenvolvido como um webservice, utilizando a biblioteca Tornado (DORY; PARRISH; BERG, 2012), para a linguagem Python. Assim, a aplicação web é chamada pelo script de workflow toda vez que um smartdata é recebido. O valor de limiar para a determinação da falha é representado pela linha vermelha horizontal. Percebe-se que, neste exemplo, todas as 3 falhas são devidamente detectadas pela estratégia. A visualização do diagnóstico resultante é feita através da plataforma aberta Grafana, utilizada para a construção de painéis para monitoramento e análise de dados de séries temporais (GRAFANA, 2019). Para tal, o Grafana, assim como o gateway, utiliza a interface REST API para a requisição dos dados presentes no servidor.

A metodologia e resultados apresentados neste Capítulo pretende demonstrar a viabilidade da plataforma para a tarefa de detecção de falhas. Através do mecanismo de *workflow*, a mesma estratégia pode ser aplicada em paralelo a múltiplos atributos, ou inclusive detectores de diferentes métodos podem ser implementados, aumentando a confiabilidade e a capacidade de isolamento de falhas da estratégia final.

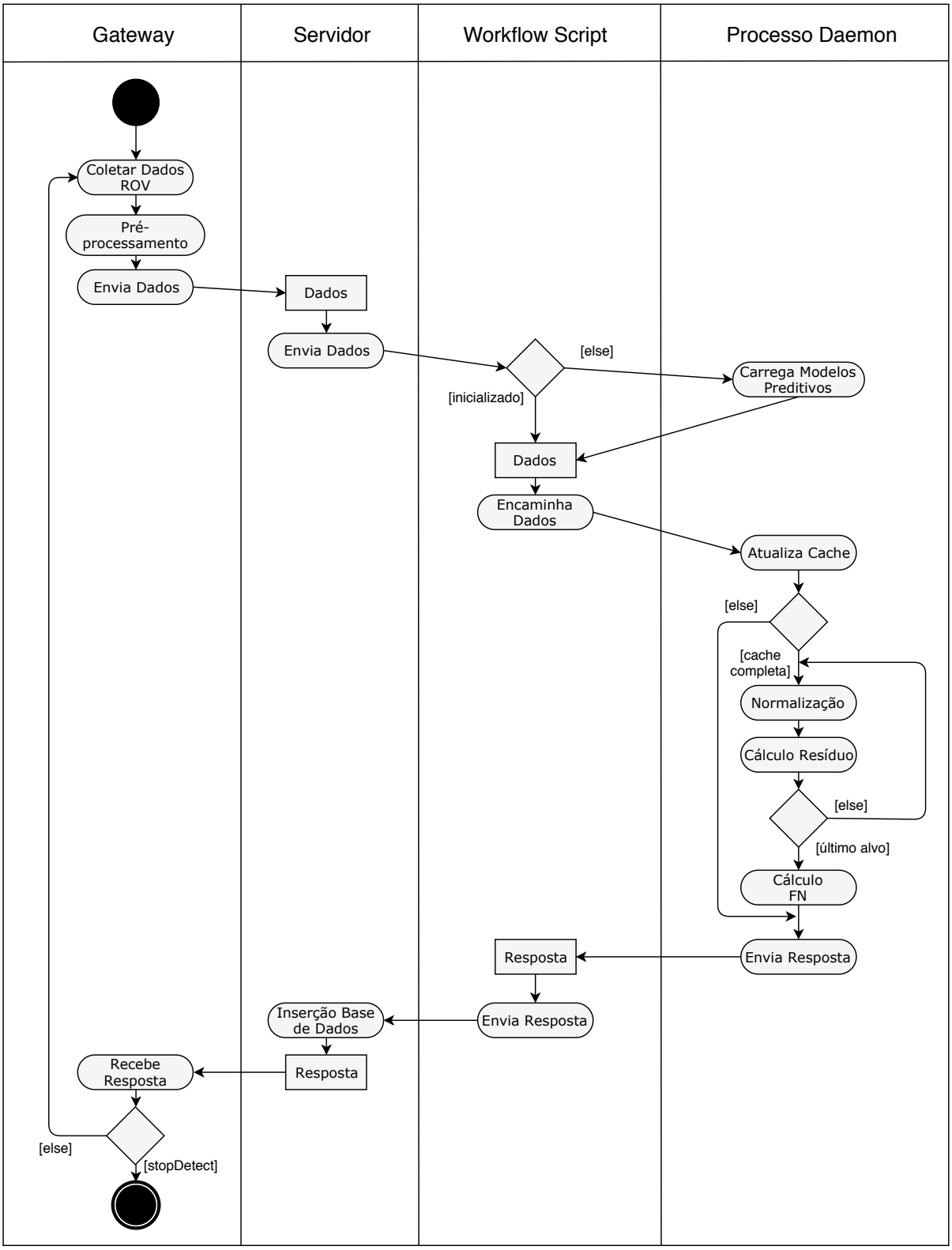
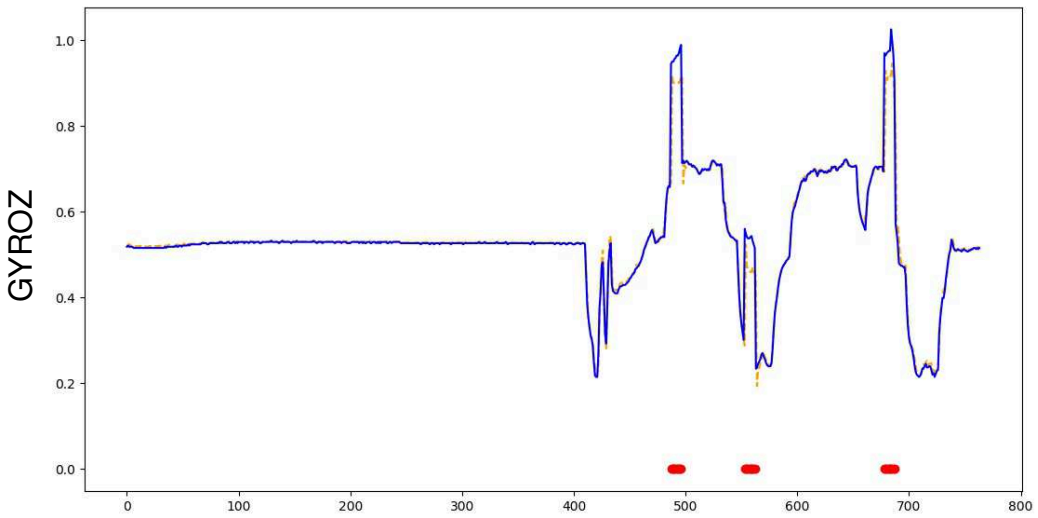
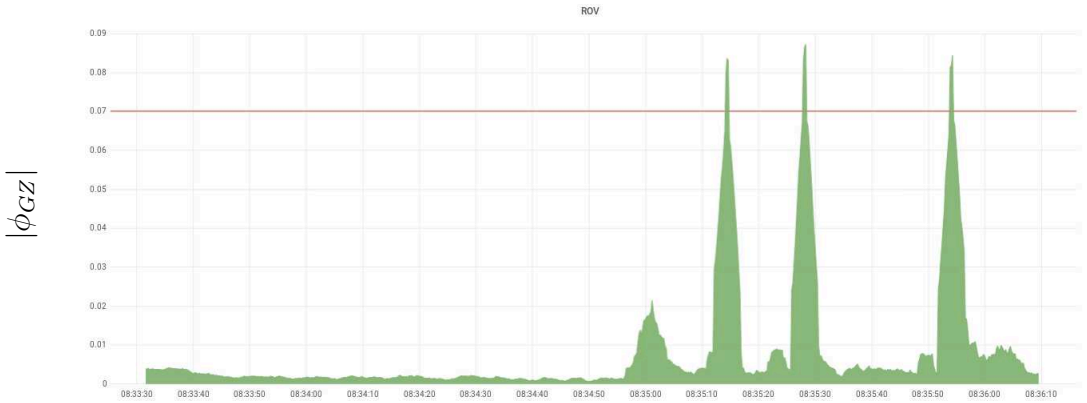


Figura 33 – Diagrama de atividade - Envio dos Dados.



(a) Estimativa de GYROZ utilizando o detector escolhido. As curvas em azul e laranja representam os valores real e estimado de GYROZ, respectivamente. Pontos em vermelho denotam os instantes em que a falha de viés ($\delta = 1$) é inserida.



(b) O processo de detecção implementado à plataforma IoT. Linha horizontal representa o valor do limiar adotado.

Figura 34 – Trecho de operação do ROV, com detector integrado à plataforma IoT.

8 CONCLUSÃO

Neste trabalho, foi apresentada uma estratégia de detecção de falhas para veículos subaquáticos não tripulados. A estratégia foi desenvolvida considerando-se algumas limitações comumente encontradas em aplicações reais: a complexidade do sistema cujas falhas devem ser detectadas, a escassez de dados referentes a estados de falhas e imprevisibilidade das possíveis condições de falha.

Os resultados foram apresentados em duas etapas distintas do trabalho. Em um primeiro momento, a capacidade de regressão dos atributos alvo GYROX, GYROY e GYROZ foi avaliada, considerando a aplicação conjunta de diversos métodos utilizados ao longo das etapas de análise de correlação, seleção de atributos e construção do modelo preditivo. Os valores de erros encontrados nesta etapa demonstram que o modelo foi capaz de estimar satisfatoriamente o valor futuro dos atributos alvo. Isto demonstra que os aspectos dinâmicos e não-lineares da presente aplicação foram bem representados pela abordagem.

Em uma segunda etapa, a capacidade de detecção de falhas da estratégia é avaliada. Para tal, falhas de sensores foram inseridas em um conjunto de teste, variando-se diferentes parâmetros das falhas, como tipo, magnitude e duração. Para diversas combinações de falhas, a estratégia foi capaz de detectar a grande maioria das falhas, além de apresentar taxa de falsos positivos próximo de zero. No entanto, outras combinações de falhas, geralmente de baixa magnitude e curta duração, se provaram imperceptíveis à estratégia utilizada.

Tomando como base as pesquisas citadas neste trabalho, a questão da intermitência das falhas é pouco abordada. Na maioria das vezes, falhas permanentes são inseridas, e os resultados são analisados com base em um trecho geralmente curto de operação do sistema analisado. No presente trabalho, o tempo de duração da falha foi um aspecto central durante a avaliação dos resultados. Além disto, um grande número de falhas foi inserido no conjunto de testes, com uma diversidade grande de falhas, com relação ao seu tipo, magnitude e duração. Desta maneira, é possível avaliarmos o detector quando submetido a diferentes condições de falha, aprofundando a análise.

Além disto, uma das contribuições deste trabalho foi avaliar o impacto de diferentes métodos de seleção de atributos utilizados na estratégia. De maneira geral, o método RReliefF demonstrou ser superior aos métodos restantes, tanto para a capacidade de regressão dos modelos preditivos quanto para a capacidade de detecção de falhas de sensores. No entanto, as comparações devem ser feitas com certo cuidado, uma vez que a variação dos parâmetros internos de cada método de seleção, além da maneira com que foram aplicados, pode levar a resultados diversos.

Por fim, a estratégia proposta, como um todo, é a contribuição principal deste trabalho. Os resultados demonstraram que a aplicação em conjunto das diversas téc-

nicas resultou em uma estratégia capaz de detectar falhas de sensores de maneira bem-sucedida, sendo capaz de auxiliar o piloto em eventuais situações de falha.

De maneira complementar, alguns aspectos da abordagem proposta são discutidos, com relação a certas características desejáveis em um sistema de detecção:

1. **Rapidez da detecção.** Para todos os casos, a latência de detecção média encontrada não foi superior a 4 segundos. Portanto, para a presente aplicação, a estratégia apresentou uma rapidez de detecção satisfatória.

2. **Adaptabilidade.** O detector proposto não assume condições definidas de falha, sendo, portanto, capaz de atuar sob condições imprevistas, apresentando boa adaptabilidade.

3. **Requisitos de modelagem.** O desenvolvimento e implementação da estratégia envolveu esforço e tempo consideráveis. No entanto, pelo fato de o fluxo de trabalho já estar bem definido e estruturado, os esforços de modelagem são consideravelmente reduzidos para etapas futuras de manutenção, em casos de adaptação da estratégia à adição de funcionalidades ou substituição de componentes do ROV, por exemplo.

8.1 LIMITAÇÕES

A estratégia proposta interpreta como falha situações em que um alto desvio do comportamento normal é observado. Portanto, o leque de situações interpretadas como falhas se torna bastante amplo. Isto pode ser considerado como uma vantagem em determinados aspectos, mas como uma desvantagem em outros. Por um lado, a estratégia é capaz de detectar não somente falhas de sensores, mas também de outras naturezas, como falhas de atuador e falhas de software, por exemplo. Por outro lado, situações anômalas podem ser, de maneira indesejada, interpretadas como uma falha. A interação do veículo com o próprio cabo umbilical, como forças de tração entre os dois, poderiam ser identificadas como falhas. Em um outro exemplo, é possível que efeitos de correntes oceânicas levem a detecções de falha pelo detector. Apesar de este exemplo poder ser considerado como um distúrbio exógeno no processo, como visto no Capítulo 1, determinados tipos de eventos podem ser considerados de interesse, ou não, de acordo com o usuário e aplicação.

Outra questão é a subjetividade do conceito de normalidade. Como visto, algumas etapas simples de pré-processamento são realizadas de maneira a eliminar possíveis pontos anômalos da base de dados que representa o comportamento normal. No entanto, em uma primeira instância, os dados coletados devem ser julgados com relação à sua normalidade, pelo piloto do veículo. Caso o ROV esteja numa condição de falha que, sob a perspectiva do piloto, não prejudique a sua adequada operação,

esta condição será incluída à base de dados de treinamento e, portanto, tratada como um comportamento normal pela estratégia.

8.2 TRABALHOS FUTUROS

A seguir, serão apresentadas possíveis extensões que, na perspectiva do autor, contribuiriam de maneira significativa para com o trabalho realizado:

1. **Estudo para outros tipos de falhas.** Como mencionado, a estratégia proposta, a princípio, é capaz de capturar falhas de diversas naturezas. Neste sentido, avaliar o desempenho da estratégia mediante a presença de outros tipos de condições, como falhas de atuador e de software, seria um importante passo para o aprofundamento da avaliação da estratégia.

2. **Isolamento de falhas.** A etapa de detecção de falhas consiste na etapa inicial em sistemas de DDF. Uma possível expansão para o projeto seria considerar a etapa posterior de isolamento de falhas, sendo possível assim determinar a origem da falha. Isto pode ser feito, por exemplo, aplicando o mesmo detector apresentado neste trabalho para diferentes sensores. Os resultados individuais para cada sensor podem ser então analisados em conjunto, permitindo uma diferenciação entre falhas de diferentes naturezas.

3. **Aplicação em outros sistemas.** Apesar de a estratégia ter sido apresentada sob o contexto de veículos subaquáticos não tripulados, não há nada que impeça a aplicação da metodologia proposta para diferentes classes de veículos e sistemas robóticos. De fato, diversos trabalhos similares foram aplicados a, por exemplo, motocicletas (CAPRIGLIONE *et al.*, 2018), sistemas robóticos autônomos (CHRISTENSEN *et al.*, 2008) e drones (KHALASTCHI *et al.*, 2011). A aplicação e avaliação da estratégia para diferentes estudos de caso é um passo importante para a evolução da estratégia proposta.

4. **Detecção em tempo real.** A implementação atual da estratégia parte de um conjunto de dados já coletados do ROV. Para a implementação da estratégia em um cenário real, no entanto, as falhas devem ser detectadas durante a operação do veículo. Neste sentido, um direcionamento futuro é a implementação da estratégia em tempo real, em conjunto com a análise de viabilidade da plataforma IoT utilizada, levando em consideração aspectos de latência da resposta e questões de escalabilidade da estratégia.

5. **Uso de diferentes métodos.** A estratégia proposta é composta pela associação de diversas análises, de forma modularizada. Desta forma, a aplicação de diferentes métodos contribui para o trabalho de duas maneiras: com a busca

por um desempenho superior da estratégia como um todo, além de comparar e discutir o desempenho individual dos métodos, quando integrado às análises restantes. Como exemplo, outras arquiteturas de RNAs podem ser utilizadas para a obtenção do modelo de regressão, como redes recorrentes (SUN; LI *et al.*, 2016; NASCIMENTO; VALDENEGRO-TORO, 2018) ou autoencoders (DAU; CIESIELSKI; SONG, 2014), por exemplo. Analogamente, métodos de seleção de atributos adicionais podem ser avaliados, além do uso de diferentes medidas de correlação para a etapa de caracterização dinâmica.

REFERÊNCIAS

AGHADAVOODI, Ehsan; SHAHGHOLIAN, Ghazanfar. A new practical feed-forward cascade analyze for close loop identification of combustion control loop system through RANFIS and NARX. **Applied Thermal Engineering**, Elsevier, v. 133, p. 381–395, 2018.

AKAIKE, Hirotugu. A new look at the statistical model identification. *In:* SELECTED Papers of Hirotugu Akaike. New York,NY: Springer, 1974. p. 215–222.

ALEKSEEV, Yu K; KOSTENKO, VV; SHUMSKY, A Ye. Use of identification and fault diagnostic methods for underwater robotics. *In:* OCEANS'94.'OCEANS Engineering for Today's Technology and Tomorrow's Preservation.'Proceedings. Brest, France: IEEE, 1994. p. ii–489.

ALESSANDRI, A; CACCIA, M; VERUGGIO, G. A model-based approach to fault diagnosis in unmanned underwater vehicles. *In:* OCEANS'98 Conference Proceedings. Nice, France: IEEE, 1998. p. 825–829.

ALESSANDRI, A; CACCIA, M; VERUGGIO, G. Fault detection of actuator faults in unmanned underwater vehicles. **Control Engineering Practice**, Pergamon, v. 7, n. 3, p. 357–368, 1999.

ALESSANDRI, A; HAWKINSON, T *et al.* Robust model-based fault diagnosis for unmanned underwater vehicles using sliding mode-observers, 1999.

ALIN, Aylin. Multicollinearity. **Wiley Interdisciplinary Reviews: Computational Statistics**, Wiley Online Library, v. 2, n. 3, p. 370–374, 2010.

ALZGHOUL, Ahmad *et al.* Comparing a knowledge-based and a data-driven method in querying data streams for system fault detection: A hydraulic drive system application. **Computers in Industry**, Elsevier, v. 65, n. 8, p. 1126–1135, 2014.

ANGELI, Chrissanthi; CHATZINIKOLAOU, Avraam. On-Line Fault Detection Techniques for Technical Systems: A Survey. **IJCSA**, v. 1, n. 1, p. 12–30, 2004.

ANTONELLI, Gianluca. A survey of fault detection/tolerance strategies for auvs and rovs. *In:* FAULT diagnosis and fault tolerance for mechatronic systems: Recent advances. Berlin: Springer, 2003. p. 109–127.

ANTONELLI, Gianluca. **Underwater Robots**. Switzerland: Springer International, jan. 2018. v. 96. DOI: 10.1007/978-3-319-02877-4.

ARDUINO. [*S.l.*: *s.n.*], 2018. https://www.arduino.cc/.

AVIZIENIS, Algirdas *et al.* Basic concepts and taxonomy of dependable and secure computing. **IEEE transactions on dependable and secure computing**, IEEE, v. 1, n. 1, p. 11–33, 2004.

BAJWA, Anupa; SWEET, Adam; KORSMEYER, David. The Livingstone model of a main propulsion system, 2003.

BAYAT, Behzad *et al.* Environmental monitoring using autonomous vehicles: a survey of recent searching techniques. **Current opinion in biotechnology**, Elsevier, v. 45, p. 76–84, 2017.

BERGSTRA, James *et al.* Theano: a CPU and GPU math expression compiler. *In:* 3.

BI, Jinbo *et al.* Dimensionality reduction via sparse support vector machines. **Journal of Machine Learning Research**, v. 3, Mar, p. 1229–1243, 2003.

BLAIR, David C. Information Retrieval. **Journal of the American Society for Information Science**, Wiley Online Library, v. 30, n. 6, p. 374–375, 1979.

BLANKE, Mogens *et al.* **Diagnosis and fault-tolerant control**. Berlin: Springer, 2006. v. 2.

BOLLINGER, Galen. **Book Review: Regression Diagnostics: Identifying Influential Data and Sources of Collinearity**. Los Angeles, CA: Sage Publication, 1981.

BONO, R.; CACCIA, M.; VERUGGIO, G. Simulation and control of an unmanned underwater vehicle. *In:* PROCEEDINGS of 1995 IEEE International Conference on Robotics and Automation. Nagoya, Japan: IEEE, mai. 1995. 1573–1578 vol.2. DOI: 10.1109/ROBOT.1995.525499.

BUENO, Elaine Inacio. **Utilização de Redes Neurais Artificiais na Monitoração e Detecção de Falhas em sensores do reator IEA-R1**. 2006. Tese (Doutorado) – Universidade de São Paulo.

CAPRIGLIONE, Domenico *et al.* NARX ANN-based instrument fault detection in motorcycle. **Measurement**, Elsevier, v. 117, p. 304–311, 2018.

CHANDOLA, Varun; BANERJEE, Arindam; KUMAR, Vipin. Anomaly detection: A survey. **ACM computing surveys (CSUR)**, ACM, v. 41, n. 3, p. 15, 2009.

CHEN, Jie; PATTON, Ron J. **Robust model-based fault diagnosis for dynamic systems**. [*S.l.*]: Springer Science & Business Media, 2012. v. 3.

CHEN, Mingzhi; ZHU, Daqi. A Novel Cooperative Hunting Algorithm for Inhomogeneous Multiple Autonomous Underwater Vehicles. **IEEE ACCESS**, IEEE-INST ELECTRICAL ELECTRONICS ENGINEERS INC 445 HOES LANE, PISCATAWAY, NJ . . ., v. 6, p. 7818–7828, 2018.

CHEN, You-Shyang. Classifying credit ratings for Asian banks using integrating feature selection and the CPDA-based rough sets approach. **Knowledge-Based Systems**, Elsevier, v. 26, p. 259–270, 2012.

CHENG, Changming; BAI, Er-Wei. Ranking the importance of variables in nonlinear system identification. **Automatica**, Elsevier, v. 103, p. 472–479, 2019.

CHOLLET, François *et al.* Keras: The python deep learning library. **Astrophysics Source Code Library**, 2018.

CHOW, EYEY; WILLSKY, AS. Analytical redundancy and the design of robust failure detection systems. **IEEE Transactions on automatic control**, IEEE, v. 29, n. 7, p. 603–614, 1984.

CHOYEKH, Mahdi *et al.* Vertical water column survey in the Gulf of Mexico using autonomous underwater vehicle SOTAB-I. **Marine Technology Society Journal**, Marine Technology Society, v. 49, n. 3, p. 88–101, 2015.

CHRIST, Robert D; WERNLI SR, Robert L. **The ROV manual: a user guide for remotely operated vehicles**. [*S.l.*]: Butterworth-Heinemann, 2013.

CHRISTENSEN, Anders Lyhne *et al.* Fault detection in autonomous robots based on fault injection and learning. **Autonomous Robots**, Springer, v. 24, n. 1, p. 49–67, 2008.

CHUNG, Junyoung *et al.* Empirical evaluation of gated recurrent neural networks on sequence modeling. **arXiv preprint arXiv:1412.3555**, 2014.

CLAESEN, Marc; DE MOOR, Bart. Hyperparameter search in machine learning. **arXiv preprint arXiv:1502.02127**, 2015.

CLOUSE, Daniel S *et al.* Time-delay neural networks: Representation and induction of finite-state machines. **IEEE Transactions on Neural Networks**, IEEE, v. 8, n. 5, p. 1065–1070, 1997.

CLYNCH, James R. Earth coordinates. **Electronic Documentation, February**, 2006.

COLEY, Gerald. Beaglebone black system reference manual. **Texas Instruments, Dallas**, 2013.

CONNECTIVITY, TE. MS5837-30BA Ultra Small Gel Filled Pressure Sensor. **Datasheet, March**, 2012.

COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, p. 21–27, jan. 1967. ISSN 0018-9448. DOI: 10.1109/TIT.1967.1053964.

DAI, X.; GAO, Z. From Model, Signal to Knowledge: A Data-Driven Perspective of Fault Detection and Diagnosis. **IEEE Transactions on Industrial Informatics**, v. 9, n. 4, p. 2226–2238, nov. 2013. ISSN 1551-3203. DOI: 10.1109/TII.2013.2243743.

DAS, Samarjit. **Time series analysis**. Princeton, NJ: Princeton University Press, 1994.

DAU, Hoang Anh; CIESIELSKI, Vic; SONG, Andy. Anomaly detection using replicator neural networks trained on examples of one class. *In:* ASIA-PACIFIC Conference on Simulated Evolution and Learning. Dunedin, NZ: Springer, 2014. p. 311–322.

DE KLEER, Johan; WILLIAMS, Brian C. Diagnosing multiple faults. **Artificial intelligence**, Citeseer, v. 32, n. 1, p. 97–130, 1987.

DEARDEN, Richard; ERNITS, Juhan. Automated fault diagnosis for an autonomous underwater vehicle. **IEEE Journal of Oceanic Engineering**, IEEE, v. 38, n. 3, p. 484–499, 2013.

DESBOULETS, Loann. A Review on Variable Selection in Regression Analysis. **Econometrics**, Multidisciplinary Digital Publishing Institute, v. 6, n. 4, p. 45, 2018.

DIEBEL, James. Representing attitude: Euler angles, unit quaternions, and rotation vectors. **Matrix**, v. 58, n. 15-16, p. 1–35, 2006.

DING, Steven X. **Model-based fault diagnosis techniques: design schemes, algorithms, and tools**. [*S.l.*]: Springer Science & Business Media, 2008.

DORY, Michael; PARRISH, Allison; BERG, Brendan. **Introduction to Tornado: Modern Web Applications with Python**. [*S.l.*]: "O'Reilly Media, Inc.", 2012.

DOUGHERTY, James; KOHAVI, Ron; SAHAMI, Mehran. Supervised and unsupervised discretization of continuous features. *In:* MACHINE Learning Proceedings 1995. Tahoe City, California: Elsevier, 1995. p. 194–202.

DRAPER, Norman R; SMITH, Harry. **Applied regression analysis**. [*S.l.*]: John Wiley & Sons, 2014. v. 326.

DUNIA, Ricardo *et al.* Identification of faulty sensors using principal component analysis. **AIChE Journal**, Wiley Online Library, v. 42, n. 10, p. 2797–2812, 1996.

EFROYMSON, MA. Multiple regression analysis. **Mathematical methods for digital computers**, John Wiley & Sons, p. 191–203, 1960.

ELMAN, Jeffrey L. Finding structure in time. **Cognitive science**, Wiley Online Library, v. 14, n. 2, p. 179–211, 1990.

FLACH, Peter. **The Art and Science of Algorithms that Make Sense of Data**. [*S.l.*]: Cambridge University Press, 2012.

FLOREZ-LOPEZ, Raquel. Modelling of insurers' rating determinants. An application of machine learning techniques and statistical models. **European Journal of Operational Research**, Elsevier, v. 183, n. 3, p. 1488–1512, 2007.

FRANK, Paul M. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy: A survey and some new results. **Automatica**, v. 26, n. 3, p. 459–474, 1990. ISSN 0005-1098. DOI: https://doi.org/10.1016/0005-1098(90)90018-D. Disponível em: http://www.sciencedirect.com/science/article/pii/000510989090018D.

FUJII, Teruo *et al.* Development of a versatile test-bed "Twin-Burger"toward realization of intelligent behaviors of autonomous underwater vehicles. **OCEANS'93. Engineering in Harmony with Ocean. Proceedings**, IEEE, p. i186–i191, 1993.

GAO, Z.; CECATI, C.; DING, S. X. A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part II: Fault Diagnosis With Knowledge-Based and Hybrid/Active Approaches. **IEEE Transactions on Industrial Electronics**, v. 62, n. 6, p. 3768–3774, jun. 2015. ISSN 0278-0046. DOI: 10.1109/TIE.2015.2419013.

GAO, Zhiwei; CECATI, Carlo; DING, Steven X. A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches. **IEEE Transactions on Industrial Electronics**, IEEE, v. 62, n. 6, p. 3757–3767, 2015.

GARCIA, Salvador *et al.* A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 25, n. 4, p. 734–750, 2012.

GERTLER, Janos. Analytical redundancy methods in fault detection and isolation-survey and synthesis. **IFAC Proceedings Volumes**, Elsevier, v. 24, n. 6, p. 9–21, 1991.

GERTLER, Janos. **Fault detection and diagnosis**. [*S.l.*]: Springer, 2013.

GOLOMBEK, Raphael. Data-driven fault detection for component based robotic systems, 2014.

GOMES, Samuel da Silva. **Modelagem e controle de atuadores robóticos e veículos subaquáticos não tripulados**. 2011. Diss. (Mestrado) – FURG.

GOVINDARAJU, Rao S; RAO, Adiseshappa Ramachandra. **Artificial neural networks in hydrology**. [*S.l.*]: Springer Science & Business Media, 2013. v. 36.

GRAFANA. **The open platform for beautiful analytics and monitoring**. [*S.l.*: *s.n.*], 2019. `https://grafana.com/`. [Online; Acessado em 18/01/2019].

GUANDALINE, Valter Hugo. HCAIM: um método de discretização supervisionado para o contexto de classificação hierárquica., 2016.

GUYON, Isabelle; ELISSEEFF, André. An introduction to variable and feature selection. **Journal of machine learning research**, v. 3, Mar, p. 1157–1182, 2003.

HABRICH, Thilo; WAGNER, Carolin; HELLINGRATH, Bernd. Qualitative Assessment of Machine Learning Techniques in the Context of Fault Diagnostics. **International Conference on Business Information Systems**, Springer, p. 359–370, 2018.

HALL, Mark Andrew. Correlation-based feature selection for machine learning. University of Waikato Hamilton, New Zealand, 1999.

HALL, Mark *et al.* The WEKA data mining software: an update. **ACM SIGKDD explorations newsletter**, ACM, v. 11, n. 1, p. 10–18, 2009.

HAWKINS, Simon *et al.* Outlier detection using replicator neural networks. **International Conference on Data Warehousing and Knowledge Discovery**, Springer, p. 170–180, 2002.

HAYKIN, Simon S *et al.* **Neural networks and learning machines/Simon Haykin.** New York: Prentice Hall, 2009.

HE, Hongwen; LIU, Zhentong; HUA, Yin. Adaptive extended kalman filter based fault detection and isolation for a lithium-ion battery pack. **Energy Procedia**, Elsevier, v. 75, p. 1950–1955, 2015.

HELBING, Georg; RITTER, Matthias. Deep Learning for fault detection in wind turbines. **Renewable and Sustainable Energy Reviews**, Elsevier, v. 98, p. 189–198, 2018.

HERTZ, John A. **Introduction to the theory of neural computation**. [*S.l.*]: CRC Press, 2018.

HIMMELBLAU, David Mautner. **Fault detection and diagnosis in chemical and petrochemical processes**. [*S.l.*]: Elsevier Science Ltd, 1978. v. 8.

HODGE, Victoria; AUSTIN, Jim. A survey of outlier detection methodologies. **Artificial intelligence review**, Springer, v. 22, n. 2, p. 85–126, 2004.

HU, Qinghua; ZHANG, Lei *et al.* Measuring relevance between discrete and continuous features based on neighborhood mutual information. **Expert Systems with Applications**, Elsevier, v. 38, n. 9, p. 10737–10750, 2011.

HU, Yang; PALMÉ, Thomas; FINK, Olga. Fault detection based on signal reconstruction with Auto-Associative Extreme Learning Machines. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 57, p. 105–117, 2017.

IHAKA, Ross; GENTLEMAN, Robert. R: a language for data analysis and graphics. **Journal of computational and graphical statistics**, Taylor & Francis Group, v. 5, n. 3, p. 299–314, 1996.

ISERMANN, Rolf. **Fault-diagnosis systems: an introduction from fault detection to fault tolerance**. [*S.l.*]: Springer Science & Business Media, 2006.

ISERMANN, Rolf. Process fault detection based on modeling and estimation methods—A survey. **automatica**, Elsevier, v. 20, n. 4, p. 387–404, 1984.

IZADY, Azizallah *et al.* Application of NN-ARX model to predict groundwater levels in the Neishaboor Plain, Iran. **Water resources management**, Springer, v. 27, n. 14, p. 4773–4794, 2013.

JÄGER, Georg *et al.* Assessing neural networks for sensor fault detection. *In:* 2014 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA). Ottawa, Canada: IEEE, 2014. p. 70–75.

JIANG, Zaihan. Underwater acoustic networks–issues and solutions. **International journal of intelligent control and systems**, v. 13, n. 3, p. 152–161, 2008.

JOHNSON, Barry. An introduction to the design and analysis of fault-tolerant systems, p. 1–87, fev. 1996.

JOLLIFFE, Ian T. **Introduction**. [*S.l.*]: Springer, 2002.

JONES, Harold Lee. **Failure detection in linear systems.** 1973. Tese (Doutorado) – Massachusetts Institute of Technology.

KALMAN, Rudolph Emil. A new approach to linear filtering and prediction problems. **Journal of basic Engineering**, American Society of Mechanical Engineers, v. 82, n. 1, p. 35–45, 1960.

KANEKO, Hiromasa; FUNATSU, Kimito. A new process variable and dynamics selection method based on a genetic algorithm-based wavelength selection method. **AIChE Journal**, Wiley Online Library, v. 58, n. 6, p. 1829–1840, 2012.

KARAGIANNOPOULOS, M *et al.* Feature selection for regression problems. **Proceedings of the 8th Hellenic European Research on Computer Mathematics & its Applications, Athens, Greece**, v. 2022, 2007.

KHALASTCHI, Eliahu *et al.* Online Anomaly Detection in Unmanned Vehicles. *In:* THE 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1. Taipei, Taiwan: International Foundation for Autonomous Agents e Multiagent Systems, 2011. (AAMAS '11), p. 115–122. Disponível em: `http://dl.acm.org/citation.cfm?id=2030470.2030487`.

KIM, Kwang-Baek; LEE, Dong-Un; SIM, Kwee-Bo. Performance improvement of fuzzy RBF networks. **International Conference on Natural Computation**, Springer, p. 237–244, 2005.

KIRA, Kenji; RENDELL, Larry A. A practical approach to feature selection. **Machine Learning Proceedings 1992**, Elsevier, p. 249–256, 1992.

KLEINBAUM, David G *et al.* **Applied regression analysis and other multivariable methods**. Belmont, CA: Duxbury Press, 1988. v. 601.

KONONENKO, Igor. Estimating attributes: analysis and extensions of RELIEF. **European conference on machine learning**, Springer, p. 171–182, 1994.

KOPRINSKA, Irena; RANA, Mashud; AGELIDIS, Vassilios G. Correlation and instance based feature selection for electricity load forecasting. **Knowledge-Based Systems**, Elsevier, v. 82, p. 29–40, 2015.

KUMAR, Senthil; LOPEZ, Daphne. Feature selection used for wind speed forecasting with data driven approaches. **Journal of Engineering Science and Technology Review**, v. 8, n. 5, p. 124–127, 2015.

KURIEN, James; NAYAK, P Pandurang. Back to the future for consistency-based trajectory tracking. *In:* p. 370–377.

LEE RODGERS, Joseph; NICEWANDER, W Alan. Thirteen ways to look at the correlation coefficient. **The American Statistician**, Taylor & Francis Group, v. 42, n. 1, p. 59–66, 1988.

LEE, Won Je *et al.* NARX modeling for real-time optimization of air and gas compression systems in chemical processes. **Computers & Chemical Engineering**, Elsevier, v. 115, p. 262–274, 2018.

LIN, Tsungnan *et al.* Learning long-term dependencies in NARX recurrent neural networks. **IEEE Transactions on Neural Networks**, IEEE, v. 7, n. 6, p. 1329–1338, 1996.

LISHA, Software/Hardware Integration Lab -. **IoT with EPOS**. [*S.l.*: *s.n.*], 2019. `https://epos.lisha.ufsc.br/IoT+with+EPOS#iot.lisha.ufsc.br_REST_API`. [Online; Acessado em 18/01/2019].

LIU, Congjian. The Sensitivity of a Test Based on Spearman's Rho in Cross-Correlation Change Point Problems, 2015.

LIU, Xiuhua; GAO, Xianwen; HAN, Jian. Robust unknown input observer based fault detection for high-order multi-agent systems with disturbances. **ISA transactions**, Elsevier, v. 61, p. 15–28, 2016.

LUENBERGER, David. Observers for multivariable systems. **IEEE Transactions on Automatic Control**, IEEE, v. 11, n. 2, p. 190–197, 1966.

MACREADIE, Peter I *et al.* Eyes in the sea: Unlocking the mysteries of the ocean using industrial, remotely operated vehicles (ROVs). **Science of The Total Environment**, Elsevier, v. 634, p. 1077–1091, 2018.

MAI, Christian *et al.* Modeling and Control of Industrial ROV's for Semi-Autonomous Subsea Maintenance Services. **IFAC-PapersOnLine**, Elsevier, v. 50, n. 1, p. 13686–13691, 2017.

MAIER, Holger R *et al.* Methods used for the development of neural networks for the prediction of water resource variables in river systems: current status and future directions. **Environmental modelling & software**, Elsevier, v. 25, n. 8, p. 891–909, 2010.

MARSLAND, Stephen. **Machine learning: an algorithmic perspective**. [*S.l.*]: Chapman e Hall/CRC, 2011.

MCPHAIL, Stephen. Autosub6000: A deep diving long range AUV. **Journal of Bionic Engineering**, Springer, v. 6, n. 1, p. 55–62, 2009.

MECHTLY, EA. The International System of Units: Physical Constants and Conversion Factors. Second Revision, 1973.

MELL, Peter; GRANCE, Tim *et al.* The NIST definition of cloud computing. Computer Security Division, Information Technology Laboratory, National . . ., 2011.

MENDENHALL, William; SINCICH, Terry; BOUDREAU, Nancy S. **A second course in statistics: regression analysis**. Upper Saddle River, NJ: Prentice Hall, 1996. v. 5.

MENON, Anil *et al.* Characterization of a class of sigmoid functions with applications to neural networks. **Neural Networks**, Elsevier, v. 9, n. 5, p. 819–835, 1996.

MEYER, Patrick E. Infotheo: Information-theoretic measures. **R package version**, v. 1, n. 0, 2014.

MEYER, Patrick Emmanual. Information-theoretic variable selection and network inference from microarray data. **Ph. D. Thesis. Université Libre de Bruxelles**, 2008.

NARASIMHAN, Sriram; BROWNSTON, Lee. HyDE-a general framework for stochastic and hybrid modelbased diagnosis. **Proc. DX**, v. 7, p. 162–169, 2007.

NASCIMENTO, Samy; VALDENEGRO-TORO, Matias. Modeling and Soft-fault Diagnosis of Underwater Thrusters with Recurrent Neural Networks. **IFAC-PapersOnLine**, Elsevier, v. 51, n. 29, p. 80–85, 2018.

NAVY, US. The navy unmanned undersea vehicle (UUV) master plan. **US Navy, November**, v. 9, p. 90, 2004.

NELSON-WONG, Erika *et al.* Application of autocorrelation and cross-correlation analyses in human movement and rehabilitation research. **journal of orthopaedic & sports physical therapy**, JOSPT, Inc. JOSPT, 1033 North Fairfax Street, Suite 304, Alexandria, VA . . ., v. 39, n. 4, p. 287–295, 2009.

PAN, Shuwen *et al.* A general extended Kalman filter for simultaneous estimation of system and unknown inputs. **Engineering Structures**, Elsevier, v. 109, p. 85–98, 2016.

PARTAN, Jim; KUROSE, Jim; LEVINE, Brian Neil. A survey of practical issues in underwater networks. **ACM SIGMOBILE Mobile Computing and Communications Review**, ACM, v. 11, n. 4, p. 23–33, 2007.

PATTON, Ron J; CHEN, Jie. Observer-based fault detection and isolation: Robustness and applications. **Control Engineering Practice**, Elsevier, v. 5, n. 5, p. 671–682, 1997.

PATTON, Ron J; FRANK, Paul M; CLARK, Robert N. **Issues of fault diagnosis for dynamic systems**. [*S.l.*]: Springer Science & Business Media, 2013.

PEDREGOSA, Fabian *et al.* Scikit-learn: Machine learning in Python. **Journal of machine learning research**, v. 12, Oct, p. 2825–2830, 2011.

PETER, B; BRUCE, A. **Practical statistics for data scientists**. [*S.l.*]: O'Reilly Media, Inc, 2017.

PHUA, Clifton; ALAHAKOON, Damminda; LEE, Vincent. Minority report in fraud detection: classification of skewed data. **Acm sigkdd explorations newsletter**, ACM, v. 6, n. 1, p. 50–59, 2004.

PIMENTEL, Marco AF *et al.* A review of novelty detection. **Signal Processing**, Elsevier, v. 99, p. 215–249, 2014.

PLUIM, Josien PW; MAINTZ, JB Antoine; VIERGEVER, Max A. Mutual-information-based registration of medical images: a survey. **IEEE transactions on medical imaging**, Citeseer, v. 22, n. 8, p. 986–1004, 2003.

PUTH, Marie-Therese; NEUHÄUSER, Markus; RUXTON, Graeme D. Effective use of Pearson's product–moment correlation coefficient. **Animal Behaviour**, Elsevier, v. 93, p. 183–189, 2014.

QI, Jun-Tong; HAN, Jian-Da. Fault diagnosis and fault-tolerant control of rotorcraft flying robots: a survey. **Zhineng Xitong Xuebao(CAAI Transactions on Intelligent Systems)**, Springer, Berlin, v. 2, n. 2, p. 31–39, 2007.

QIN, Yao *et al.* A dual-stage attention-based recurrent neural network for time series prediction. **arXiv preprint arXiv:1704.02971**, 2017.

RAANAN, Ben-Yair *et al.* Detection of unanticipated faults for autonomous underwater vehicles using online topic models. **Journal of Field Robotics**, Wiley Online Library, v. 35, n. 5, p. 705–716, 2018.

RAMESH, Babu N; ARULMOZHIVARMAN, P. Dynamic neural network based very short-term wind speed forecasting. **Wind Engineering**, SAGE Publications Sage UK: London, England, v. 38, n. 2, p. 121–128, 2014.

RESNER, Davi *et al.* Performance evaluation of the trustful space-time protocol, 2018.

REUNANEN, Juha. Overfitting in making comparisons between variable selection methods. **Journal of Machine Learning Research**, v. 3, Mar, p. 1371–1382, 2003.

RINGNÉR, Markus. What is principal component analysis? **Nature biotechnology**, Nature Publishing Group, v. 26, n. 3, p. 303, 2008.

ROBINSON, Cecil; SCHUMACKER, Randall E. Interaction effects: centering, variance inflation factor, and interpretation issues. **Multiple Linear Regression Viewpoints**, Citeseer, v. 35, n. 1, p. 6–11, 2009.

ROBNIK-SIKONJA, M; SAVICKY, P. CORElearn-classification, regression, feature evaluation and ordinal evaluation. **The R Project for Statistical Computing**, 2012.

ROBNIK-SIKONJA, Marko; KONONENKO, Igor. Theoretical and empirical analysis of ReliefF and RReliefF. **Machine learning**, Springer, v. 53, n. 1-2, p. 23–69, 2003.

ROGERSON, Peter A. Statistical methods for geography: a student's guide. Sage, 2014.

ROSENBLATT, Frank. The perceptron, a perceiving and recognizing automaton Project Para. Cornell Aeronautical Laboratory, 1957.

ROSENBLATT, Murray. Remarks on some nonparametric estimates of a density function. **The Annals of Mathematical Statistics**, JSTOR, p. 832–837, 1956.

ROSSUM, G. van. **Python tutorial**. Amsterdam, mai. 1995.

RUIJTERS, Enno; STOELINGA, Mariëlle. Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. **Computer science review**, Elsevier, v. 15, p. 29–62, 2015.

SÁNCHEZ-FERNÁNDEZ, A *et al.* Fault detection based on time series modeling and multivariate statistical process control. **Chemometrics and Intelligent Laboratory Systems**, Elsevier, v. 182, p. 57–69, 2018.

SARTORI, Isabel *et al.* Detecção, Diagnóstico e Correção de Falhas: Uma Proposição Consistente de Definições e Terminologias. **Ciência & Engenharia**, v. 21, n. 2, p. 41–53, 2012.

SEABOLD, Skipper; PERKTOLD, Josef. Statsmodels: Econometric and statistical modeling with python. *In:* SCIPY. PROCEEDINGS of the 9th Python in Science Conference. [*S.l.*: *s.n.*], 2010. p. 61.

SEMICONDUCTORS, Philips. The I2C-bus specification. **Philips Semiconductors**, v. 9397, n. 750, p. 00954, 2000.

SENSORTEC, Bosch. Intelligent 9-axis absolute orientation sensor. **BNO055 datasheet, November**, 2014.

SHOWSTACK, Randy. Unmanned research vessel lost on deep sea dive. **Eos, Transactions American Geophysical Union**, Wiley Online Library, v. 95, n. 20, p. 168–168, 2014.

SHUMSKY, Alexey; ZHIRABOK, Alexey; HAJIYEV, Chingiz. Observer based fault diagnosis in thrusters of autonomous underwater vehicle. **Control and Fault-Tolerant Systems (SysTol), 2010 Conference on**, IEEE, p. 11–16, 2010.

SOARES, Matheus Victor Brum. **Aprendizado de máquina parcialmente supervisionado multidescriçao para realimentaçao de relevância em recuperaçao de informaçao na WEB**. 2009. Tese (Doutorado) – Universidade de São Paulo.

SPONG, Mark W. Underactuated mechanical systems. *In:* SICILIANO, Bruno; VALAVANIS, Kimon P. (Ed.). **Control Problems in Robotics and Automation**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. p. 135–150.

STACKPOLE, E; LANG, D. **OpenROV**. [*S.l.*: *s.n.*], 2013.

SULLIVAN, Gail M; FEINN, Richard. Using effect size—or why the P value is not enough. **Journal of graduate medical education**, The Accreditation Council for Graduate Medical Education Suite 2000, 515 . . ., v. 4, n. 3, p. 279–282, 2012.

SUN, Pei; CHAWLA, Sanjay. On local spatial outliers. IEEE, 2004.

SUN, Yu-shan; LI, Yue-ming *et al.* Actuator fault diagnosis of autonomous underwater vehicle based on improved Elman neural network. **Journal of Central South University**, Springer, v. 23, n. 4, p. 808–816, 2016.

TAKAI, Motoyuki; URA, Tamaki. Development of a system to diagnose autonomous underwater vehicles. **International Journal of Systems Science**, Taylor & Francis, v. 30, n. 9, p. 981–988, 1999.

TAN, Pang-Ning. Introduction to data mining. Pearson Education India, 2018.

TAX, David MJ; DUIN, Robert PW. Support vector domain description. **Pattern recognition letters**, Elsevier, v. 20, n. 11-13, p. 1191–1199, 1999.

TIDRIRI, Khaoula *et al.* Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. **Annual Reviews in Control**, Elsevier, v. 42, p. 63–81, 2016.

TINÓS, Renato. **Detecção e diagnóstico de falhas em robôs manipuladores via redes neurais artificiais**. 1999. Tese (Doutorado) – Dissertação de mestrado, Escola de Engenharia de São Carlos, Universidade de São Paulo.

TRIANDIS, Harry C. The psychological measurement of cultural syndromes. **American psychologist**, American Psychological Association, v. 51, n. 4, p. 407, 1996.

ULERICH, Nancy H; POWERS, Gary J. On-line hazard aversion and fault diagnosis in chemical processes: the digraph+ fault-tree method. **IEEE Transactions on Reliability**, IEEE, v. 37, n. 2, p. 171–177, 1988.

URICK, Robert J. Principles of underwater sound for engineers. Tata McGraw-Hill Education, 1967.

VAPNIK, VN. The Nature of Statistical Learning. **Theory**, Springer, 1995.

VENKATASUBRAMANIAN, V. *et al.* A review of process fault detection and diagnosis part I: Quantitative model-based methods. **Computers and Chemical Engineering**, v. 27, n. 3, p. 293–311, 2003. cited By 1534. DOI: 10.1016/S0098-1354(02)00160-6.

VENKATASUBRAMANIAN, Venkat. Prognostic and diagnostic monitoring of complex systems for product lifecycle management: Challenges and opportunities. **Computers & chemical engineering**, Elsevier, v. 29, n. 6, p. 1253–1263, 2005.

VENKATASUBRAMANIAN, Venkat *et al.* A review of process fault detection and diagnosis: Part III: Process history based methods. **Computers & chemical engineering**, Elsevier, v. 27, n. 3, p. 327–346, 2003.

VENKATSUBRAMANIAN, Venkat. A review of process fault detection and diagnosis, part ii: Qualitative models and search strategics. **Computers and Chemical Engineering**, v. 27, n. 3, p. 313–326, 2003.

VERVOORT, JHAM. Modeling and control of an unmanned underwater vehicle. **Master traineeship report, University of Canterbury, Christchurch, New Zealand**, 2008.

VU, Dao Hoang; MUTTAQI, Kashem M; AGALGAONKAR, AP. A variance inflation factor and backward elimination based robust regression model for forecasting monthly electricity demand using climatic variables. **Applied Energy**, Elsevier, v. 140, p. 385–394, 2015.

WANG, Yu. A new concept using LSTM Neural Networks for dynamic system identification. **2017 American Control Conference (ACC)**, IEEE, p. 5324–5329, 2017.

WANG, Yujia; ZHANG, Mingjun. Research on test-platform and condition monitoring method for AUV. **Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference on**, IEEE, p. 1673–1678, 2006.

WEI, William WS. Time series analysis. **The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2**, 2006.

WEISS, Gary M. Mining with rarity: a unifying framework. **ACM Sigkdd Explorations Newsletter**, ACM, v. 6, n. 1, p. 7–19, 2004.

WESTFALL, Peter H; YOUNG, S Stanley *et al.* Resampling-based multiple testing: Examples and methods for p-value adjustment. John Wiley & Sons, v. 279, 1993.

WILLIAMS, Brian C; NAYAK, P Pandurang. A model-based approach to reactive self-configuring systems. **Proceedings of the national conference on artificial intelligence**, p. 971–978, 1996.

WITCZAK, Marcin. Modelling and estimation strategies for fault diagnosis of non-linear systems: from analytical to soft computing approaches. Springer Science & Business Media, v. 354, 2007.

WUNSCH, Andreas; LIESCH, Tanja; BRODA, Stefan. Forecasting groundwater levels using nonlinear autoregressive networks with exogenous input (NARX). **Journal of Hydrology**, Elsevier, v. 567, p. 743–758, 2018.

XIANG, Xianbo; YU, Caoyang; NIU, Zemin *et al.* Subsea cable tracking by autonomous underwater vehicle with magnetic sensing guidance. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 16, n. 8, p. 1335, 2016.

XIANG, Xianbo; YU, Caoyang; ZHANG, Qin. On intelligent risk analysis and critical decision of underwater robotic vehicle. **Ocean Engineering**, Elsevier, v. 140, p. 453–465, 2017.

YANG, Feihu *et al.* An improved feature selection approach based on ReliefF and Mutual Information. **International Conference on Information Science and Technology**, p. 246–250, 2011.

ZHANG, Fumin *et al.* Future trends in marine robotics [TC Spotlight]. **IEEE Robotics & Automation Magazine**, IEEE, v. 22, n. 1, p. 14–122, 2015.

ZHANG, Ming-jun; WU, Juan; CHU, Zhen-zhong. Multi-fault diagnosis for autonomous underwater vehicle based on fuzzy weighted support vector domain description. **China Ocean Engineering**, Springer, v. 28, n. 5, p. 599–616, 2014.

ZHANG, Rong *et al.* Meteorological drought forecasting based on a statistical model with machine learning techniques in Shaanxi province, China. **Science of The Total Environment**, Elsevier, v. 665, p. 338–346, 2019.

ZHANG, Xiaodong; POLYCARPOU, Marios M; PARISINI, Thomas. A robust detection and isolation scheme for abrupt and incipient faults in nonlinear systems. **IEEE transactions on automatic control**, IEEE, v. 47, n. 4, p. 576–593, 2002.

ZHU, Daqi; GU, Wei. SENSOR FAULT DIAGNOSIS OF AUTONOMOUS UNDERWATER VEHICLE BASED ON NONLINEAR PRINCIPALCOMPONENT ANALYSIS. **IFAC Proceedings Volumes**, Elsevier, v. 40, n. 15, p. 67–70, 2007.

APÊNDICE A – CENÁRIOS EXPERIMENTAIS

Código A.1 – Exemplo de definição dos parâmetros de um cenário de inserção de falhas.

```
{
                "scenario_number":8,
                "number_faults":100,
                "duration_fault":[25],
                "gap_between_faults":50,
                "insert_when_active":"True",

                "stuck_at":{
                    "active":"False",
                    "feature": ["GYROZ"],
                    "constant": [0]
                },
                "constant_gain":{
                    "active":"True",
                    "feature": ["GYROZ"],
                    "coefficient": [1.5]
                },
                "constant_bias":{
                    "active":"False",
                    "feature": ["GYROZ"],
                    "constant": [25]
                },
                "drift":{
                    "active":"False",
                    "feature": ["GYROZ"],
                    "slope": [20]
                }
```

ANEXO A – ESTRUTURA DOS DADOS COLETADOS

Código A.1 – Estrutura dos dados de navegação coletados.


```
{
    "0": {
        "data": [
                        {
                "deapth": "0.00",
                "hdgd": "0.00",
                "pitch": "0.00",
                "roll": "0.00",
                "thrust": "0.00",
                "timestamp": 1521137926880,
                "yaw": "0.00"
            }
        ],
        "name": "navdata"
    },
```


Código A.2 – Estrutura dos dados de telemetria coletados.


```
    "1": {
        "data": [
            {

                "AVCC": "5233",
                "BNO_INIT_STATUS": "FAILED",
                "BRDI": "0.63",
                "BRDT": "55.11",
                "BRDV": "8.86",
                "BT1I": "0.24",
                "BT2I": "0.55",
                "CAPA": "254",
                "Depth.C0": "INVALID",
                "Depth.C1": "INVALID",
                "Depth.C2": "INVALID",
                "Depth.C3": "INVALID",
                "Depth.C4": "INVALID",
```

```
"Depth.C5": "INVALID",
"Depth.C6": "INVALID",
"Depth.C7": "INVALID",
"Depth.CoeffFailure": "C7",
"Depth.crcCheck": "CantPerform",
"LIGP": "0.00",
"LIGT": "0",
"SC1I": "0.02",
"SC2I": "0.02",
"SC3I": "0.02",
"alps": "2941",
"atmp": "0.00",
"btti": "0.77",
"cmd": "ping(0)",
"cmpd": "Jan 24 2018, 21",
"cpuUsage": 0.6052631578947368,
"crc": "pass",
"deap": "0.00",
"fmem": "3447.00",
"fthr": "0.00",
"hdgd": "0.00",
"icmd": "start()",
"iout": 0.6,
"log": "Attempting to fetch depth cal coeff",
"modtime": "NA|0|NA|0|NA|0|NA|0|NA|0|NA|0|NA|
   0|NA|0|NA|1|NA|0|NA|0|NA|0|",
"motorAttached": "1",
"motors": "1500,1500,1500",
"mpu_init failed with code": " -1",
"mtarg": "1500,1500,1500",
"mtrmod": "1.00,1.00,-1.00,2.00,2.00,-2.00",
"norm_pitch": "0.00",
"norm_roll": "0.00",
"norm_yaw": "0.00",
"pitc": "0.00",
"pong": "1,950188",
"pres": "0.00",
"rawcmd": "9E,70,69,6E,67,28,30,29,3B,",
"roll": "0.00",
```

```
                "servo": "1500",
                "starg": "1500",
                "targetDepth": "DISABLED",
                "targetHeading": "DISABLED",
                "temp": "0.00",
                "time": "949702.00",
                "timestamp": 1521137926206,
                "ver": "CUSTOM_BUILD",
                "vout": 8.88,
                "yaw": "0.00",
                "yawCommand": "0.00",
                "yawError": "0.00"
            }
        ],
        "name": "telemetry"
    }
}
```