CENTRO UNIVERSITÁRIO SENAI CIMATEC

PROGRAMA DE PÓS-GRADUAÇÃO STRICTO SENSU MODELAGEM COMPUTACIONAL E TECNOLOGIA INDUSTRIAL

ILAN SOUSA FIGUEIRÊDO

UMA NOVA ABORDAGEM DE INTELIGÊNCIA ARTIFICIAL BASEADA EM AUTOAPRENDIZAGEM PROFUNDA PARA MANUTENÇÃO PREDITIVA EM UM AMBIENTE DE PRODUÇÃO DE PETRÓLEO E GÁS *OFFSHORE*

Defesa de Doutorado

Salvador, 2023

ILAN SOUSA FIGUEIRÊDO

UMA NOVA ABORDAGEM DE INTELIGÊNCIA ARTIFICIAL BASEADA EM AUTOAPRENDIZAGEM PROFUNDA PARA MANUTENÇÃO PREDITIVA EM UM AMBIENTE DE PRODUÇÃO DE PETRÓLEO E GÁS *OFFSHORE*

Defesa de Doutorado apresentada ao Programa de Pós-Graduação Stricto Sensu do Centro Universitário SENAI CIMATEC como requisito parcial para a obtenção do título de Doutor em Modelagem Computacional e Tecnologia Industrial

Orientador: Prof. Dr. Erick Sperandio Nascimento

Coorientador: Prof.ª Dr.ª Lílian Lefol Nani Guarieiro

Salvador, 2023

Ficha catalográfica elaborada pela Biblioteca do Centro Universitário SENAI CIMATEC

F475n Figueirêdo, Ilan Sousa

Uma nova abordagem de inteligência artificial baseada em autoaprendizagem profunda para manutenção preditiva em um ambiente de produção de petróleo e gás *Offshore* / Ilan Sousa Figueirêdo. – Salvador, 2023.

132 f. : il. color.

Orientador: Prof. Dr. Erick Giovani Sperandio Nascimento. Coorientadora: Prof.ª Dr.ª Lílian Lefol Nani Guarieiro.

Tese (Doutorado em Modelagem Computacional e Tecnologia Industrial) – Programa de Pós-Graduação, Centro Universitário SENAI CIMATEC, Salvador, 2023.

Inclui referências.

1. Aprendizado profundo. 2. Séries temporais multivariadas. 3. Diagnóstico de falhas. 4. Autoaprendizagem. I. Centro Universitário SENAI CIMATEC. II. Nascimento, Erick Giovani Sperandio. III. Guarieiro, Lílian Lefol Nani. IV. Título.

2.

CDD 006.3

NDI - 01

Sistema FIEB | SENAI CIMATEC
PELO FUTURO DA INOVAÇÃO

Centro Universitário SENAI CIMATEC

Doutorado em Modelagem Computacional e Tecnologia Industrial

A Banca Examinadora, constituída pelos professores abaixo listados, leu e aprovou a Tese de doutorado, intitulada "**UMA NOVA ABORDAGEM DE INTELIGÊNCIA ARTIFICIAL BASEADA EM AUTOAPRENDIZAGEM PROFUNDA PARA MANUTENÇÃO PREDITIVA EM UM AMBIENTE DE PRODUÇÃO DE PETRÓLEO E GÁS OFFSHORE**", apresentada no dia 16 de fevereiro de 2023, como parte dos requisitos necessários para a obtenção do Título de Doutor em Modelagem Computacional e Tecnologia Industrial.

Electronically signed by:
Erick Giovani Sperandio Nascimento
CPF: ***.666.177-**
Date: 2/23/2023 2:48:44 PM +00:00

Orientador: **Prof. Dr. Erick Giovani Sperandio Nascimento**
SENAI CIMATEC

Assinado eletronicamente por:
LILIAN Lefol Nani Guarieiro
CPF: ***.720.076-**
Data: 23/02/2023 11:39:07 -03:00

Coorientadora: **Prof.ª Dr.ª Lílian Lefol Nani Guarieiro**
SENAI CIMATEC

Assinado eletronicamente por:
Alex Álisson Bandeira Santos
CPF: ***.191.765-**
Data: 23/02/2023 11:09:09 -03:00

Membro Interno: **Prof. Dr. Alex Álisson Bandeira Santos**
SENAI CIMATEC

Assinado eletronicamente por:
Fernando Luiz Pellegrini Pessoa
CPF: ***.470.585-**
Data: 23/02/2023 11:33:17 -03:00

Membro Interno: **Prof. Dr. Fernando Luiz Pellegrini Pessoa**
SENAI CIMATEC

Electronically signed by:
Fábio Borges de Oliveira
CPF: ***.940.659-**
Date: 3/1/2023 9:08:55 PM -03:00

Membro Externo: **Prof. Dr. Fábio Borges de Oliveira**
LNCC

Assinado eletronicamente por:
Ricardo Emanuel Vaz Vargas
CPF: ***.331.306-**
Data: 23/02/2023 20:41:50 -03:00

Membro Externo: **Prof. Dr. Ricardo Emanuel Vaz Vargas**
PETROBRAS

Agradecimentos

Primeiramente, agradeço a Deus por tudo que já fez por mim. Sou grato pela vida que me deu, sou grato por ter saúde e por ter a capacidade de aprender com tudo aquilo que a vida coloca em meu caminho, e com todas as escolhas que faço, nas certas e nas erradas.

Segundamente, agradeço à minha esposa Gabriela Figueirêdo, por estar sempre ao meu lado em todos os momentos, assim como pela compreensão e paciência demonstrada durante todos esses anos de estudo.

Em especial aos meus pais Jorge Figueirêdo, Diana Figueirêdo e meu irmão Judah Figueirêdo, pela fé depositada em mim, pelo amor incondicional e pela educação passada. Se hoje sou capaz de realizar meus sonhos, devo muito a vocês por me ensinar no dia a dia que posso tudo quando se coloca amor e dedicação.

Agradeço a todos os meus amigos por nunca me deixarem na solidão. Apesar de todos os meus defeitos, escolheram dividir as suas vidas comigo.

Sou imensamente grato à confiança depositada dos meus orientadores Prof. Dr. Erick Nascimento e Prof.ª Dr.ª Lílian Guarieiro em minha pessoa. Fundamental a orientação exercida durante todos esses anos, sempre muito presentes e exigentes nos estudos e trabalhos acadêmicos.

Agradeço ao Centro de Supercomputação e Inovação Industrial do SENAI CIMATEC pela disponibilização de computadores de alto desempenho, essenciais para o desenvolvimento desta pesquisa.

Sou grato a todos os membros da banca por se disponibilizarem para colaborar na construção desta pesquisa.

Meus agradecimentos também para os meus colegas de trabalho, profissionais de alto nível, que me motivaram a estudar rotineiramente e por terem tornado a minha caminhada mais suave.

Agradeço também a todo o corpo docente do PPG MCTI que contribuíram vastamente nos meus conhecimentos acadêmicos.

Enfim, tudo o que eu fiz, tudo o que eu tenho, tudo o que eu sou, é graças ao Senhor, meu bom Deus. Sou grato pela dádiva da vida que o Senhor me concedeu.

“Por isso não tema, pois estou com você; não tenha medo, pois sou o seu Deus. Eu o fortalecerei e o ajudarei; Eu o segurarei com a minha mão direita vitoriosa”.

Isaías 41:10

“Eis que o Leão da tribo de Judá, a Raiz de Davi, venceu para abrir o livro e os seus sete selos”.

Apocalipse 5:5

Resumo

Atualmente, a indústria passa por uma transição da Indústria 3.0 para a Indústria 4.0, também chamada de "Quarta Revolução Industrial". Essa mudança está associada à integração entre sistemas físicos e digitais dos ambientes de manufatura. Essa integração permite a aquisição, armazenamento, processamento e análise de uma grande quantidade de dados de diversos processos industriais. A título de exemplo, histórico de dados de uma linha de produção industrial contém informações sobre processos, equipamentos, eventos e alarmes que são essenciais para uma tomada de decisão assertiva. Logo, quando processados e analisados, podem trazer informações e conhecimentos valiosos sobre o estado de integridade dos ativos. Dessa forma, as Redes Neurais Artificiais (RNA) tornam-se tecnologias chaves para o desenvolvimento de soluções da manutenção preditiva para a Indústria 4.0, uma vez que possuem a capacidade de processar grande volume de dados para suporte a tomada de decisão. No entanto, dados rotulados são escassos ou até inexistentes na indústria, pois há uma limitação de tempo quanto à capacidade dos especialistas em processar tal quantidade de dados. Nesse contexto, o objetivo desta tese foi pesquisar, estudar e desenvolver um modelo de autoaprendizagem profunda para classificar falhas em séries temporais multivariadas. A autoaprendizagem é uma combinação de aprendizagem de máquina não supervisionada e supervisionada. A principal vantagem é a não necessidade de dados rotulados na etapa de treinamento do modelo. A abordagem proposta foi desenvolvida e validada utilizando dados reais e públicos da Petrobras, conhecida como base 3W. Esses dados possuem o histórico de falhas de produção de óleo e gás (O&G) de poços *offshore*. Logo, o desenvolvimento do modelo contém quatro principais etapas. Primeiramente, separou-se aleatoriamente uma pequena porção de dados não rotulados da base. Posteriormente, utilizou-se a aprendizagem de máquina não supervisionada para pseudo-rotulagem dos dados por meio do reconhecimento de padrões anômalos e normais. Em seguida, inicia-se o ciclo iterativo de desenvolvimento da aprendizagem de máquina profunda supervisionada. Nesse ciclo, o modelo é treinado com dados pseudo-rotulados. Após o treinamento, realiza-se predições em novos conjuntos de dados não rotulados para gerar novos pseudo-rótulos. Os novos dados não rotulados são acrescentados gradativamente a cada iteração de treinamento do ciclo. Para tanto, por meio dos dados de probabilidade da função de ativação Softmax, uma Camada de Confiança foi projetada para pseudo-rotulagem nas predições mais confiáveis. Por fim, o modelo finaliza o desenvolvimento quando o critério de parada for cumprido e interromper o *loop* de treinamento. Como resultado, inicialmente, tem-se um conjunto de dados rotulados e um modelo que pode ser aprimorado com mais dados não rotulados. Inclusive, através de novos ciclos de desenvolvimento, pode-se utilizar *Transfer Learning* para ensinar outros tipos de falhas ao modelo e transformá-lo em multiclasse. Portanto, o modelo proposto foi capaz de classificar a iminência de múltiplas falhas de O&G em séries temporais multivariadas. O modelo semi-supervisionado de autoaprendizagem profunda apresentou um desempenho equiparável à aprendizagem profunda supervisionada. Os modelos semi-supervisionados e supervisionados apresentaram acurácia de 96% e 99%, respectivamente. Desse modo, o modelo proposto apresentou recursos para viabilizar a aprendizagem de máquina em um ambiente operacional industrial, uma vez que foi superada a carência de dados anotados.

Palavras-chave: Aprendizado Profundo, Séries Temporais Multivariadas, Diagnóstico de Falhas, Autoaprendizagem.

Abstract

Currently, the industry is undergoing a transition from Industry 3.0 to Industry 4.0, also called the "Fourth Industrial Revolution". This change is associated with the integration between physical and digital systems in manufacturing environments. This integration allows the acquisition, storage, processing and analysis of a large amount of data from various industrial processes. For example, historical data from an industrial production line contains information about processes, equipment, events and alarms that are essential for assertive decision making. Therefore, when processed and analyzed, they can bring valuable information and knowledge about the state of integrity of the assets. In this way, Artificial Neural Networks (ANN) become key technologies for the development of predictive maintenance solutions for Industry 4.0, since they have the ability to process large volumes of data to support decision making. However, labeled data is scarce or even non-existent in the industry, as there is a time constraint on the ability of specialists to process such an amount of data. In this context, the objective of this thesis was to research, study and develop a deep self-learning model to classify failures in multivariate time series. Self-learning is a combination of unsupervised and supervised machine learning. The main advantage is no need for labeled data in the model training step. The proposed approach was developed and validated using real and public data from Petrobras, known as the 3W base. This data has the history of oil and gas (O&G) production failures from offshore wells. Therefore, the development of the model contains four main steps. First, a small portion of unlabeled data from the database was randomly separated. Subsequently, unsupervised machine learning was used to pseudo-label the data through the recognition of anomalous and normal patterns. Then the supervised deep machine learning iterative development cycle begins. In this cycle, the model is trained with pseudo-labeled data. After training, predictions are performed on new unlabeled datasets to generate new pseudo-labels. New unlabeled data is gradually added to each training iteration of the cycle. For that, through the probability data of the Softmax activation function, a Confidence Layer was designed for pseudo-labeling in the most reliable predictions. Finally, the model ends development when the stopping criterion is met and breaks the training loop. As a result, initially, we have a labeled dataset and a model that can be improved with more unlabeled data. Even, through new development cycles, Transfer Learning can be used to teach other types of failures to the model and transform it into a multiclass. Therefore, the proposed model was able to classify the imminence of multiple O&G failures in multivariate time series. The semi-supervised deep self-learning model performed on par with supervised deep learning. The semi-supervised and supervised models showed an accuracy of 96% and 99%, respectively. Thus, the proposed model presented resources to enable machine learning in an industrial operational environment, since the lack of annotated data was overcome.

Keywords: Deep Learning, Multivariate Time-Series, Fault Diagnosis, Self-Learning.

Lista de Figuras

Lista de Siglas

AE - Autoencoder
AQV - Aprendizado por Quantização Vetorial
ANN - Artificial Neural Networks
C-AMDATS - A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance
CNN – Convolution Neural Network
CPU - Central Processing Unit
CS2I - Centro de Supercomputação para Inovação Industrial
FPSO - Floating, Production, Storage and Offloading
GPU - Graphics Processing Unit
GTC - GPU Technology Conference
HPC - High Performance Computing
IoT - Internet of Things
JSCI - Journal on Systemics, Cybernetics and Informatics
LSTM - Long Short-Term Memory
MC - Manutenção Corretiva
MLP - Multilayer Perceptron
MPd - Manutenção Preditiva
MPv - Manutenção Preventiva
O&G - Óleo e Gás
OTC - Offshore Technology Conference
QR-PCK - Quick Restriction in Production Choke
RBF - Radial-basis function
RNA - Redes Neurais Artificiais
RNR - Redes Neurais Recorrentes
SC-DHSV - Spurious Closure Down Safety Valve
SOM - Self-Organized Map
SVC – Suport Vector Machine
TI - Tecnologia da Informação
TPAMI - Transactions on Patterns Analysis and Machine Intelligence
VUR - Vida Útil Remanescente
4IR - Fourth Industrial Revolution

Sumário

1. Introdução

O termo “Quarta Revolução Industrial'', do inglês *Fourth Industrial Revolution* (4IR) surgiu na Alemanha para atender às novas demandas de inovação, tais como: conectividade, quantidade de dados, redução de estoque, customização e produção controlada (BEHRENS; VIETE, 2020). A customização e disponibilidade de dados são elementos essenciais da Indústria 4.0 (JIN et al., 2017, LEE et al., 2014). Às diferenças entre as indústrias atuais e o modelo 4.0 dividem-se principalmente em três linhas: componentes (autoconsciente, autopreditivo); máquinas (autoconscientes, autopreditivas, autocomparadas) e sistema produtivo (auto-configurado, auto-mantido, auto-organizado) (ZONTA *et al*., 2020).

Com o surgimento das novas demandas 4.0, a disponibilidade de grande quantidade de dados se torna elemento chave para gerar informações a fim de assistir ou antecipar tomadas de decisões de especialistas em diversas áreas do conhecimento (*e.g.* manutenção industrial, meteorologia, saúde). Portanto, por meio do advento das tecnologias da Indústria 4.0 (*e.g.* inteligência artificial, robótica, internet das coisas, computação em nuvem, sistema distribuído e computação de alto desempenho), tornou-se possível analisar uma grande quantidade de dados e o intercâmbio de informações entre pessoas, máquinas e produtos mais rápido e direcionado (BORGI et al., 2017; RAUCH; LINDER; DALLASEGA, 2020).

Os dados adquiridos por meio do monitoramento de ativos industriais contêm informações sobre processos, equipamentos, eventos e alarmes que ocorrem ao longo de uma linha de produção industrial. Esses dados, quando processados e analisados adequadamente, podem trazer informações e conhecimentos do estado de saúde do ativo (CARVALHO et al., 2019). Entretanto, muitas vezes não é viável analisá-los e rotulá-los devido aos grandes volumes de dados gerados. Pois, há uma limitação quanto à capacidade dos especialistas em processar uma grande quantidade de dados, exigindo muitas horas de trabalho manual e tedioso que, em geral, estão envolvidos com outras atividades e não dispõem de tempo necessário para essa atividade tão relevante. Assim posto, percebe-se a necessidade de automatizar o processo de identificação de padrões de interesse ou anômalos presentes em dados de séries temporais (NASCIMENTO; TAVARES; DE SOUZA, 2015).

A Manufatura Moderna ou Manufatura Preditiva exige um nível de qualidade alta no seu processo de fabricação de produtos e/ou tecnologias envolvidas. Portanto, para atender os requisitos exigidos de produtividade, confiabilidade e segurança operacional, é fundamental um plano de gerenciamento que coordene as diretrizes e os recursos de forma proativa para a garantir a eficácia da manufatura e os custos de manutenção (GAO et al., 2015). No entanto, um fato inevitável e que ameaça o plano de gerenciamento de manutenção é a ocorrência de paradas dispendiosas, não programadas e quebras inesperadas (AMRUTHNATH; GUPTA, 2018). Em tal caso, o mecanismo responsável por mitigar essas ocorrências é a manutenção industrial que executa a substituição ou restauração de componentes ou elementos dos equipamentos e o descarte de peças deterioradas (DIN, 2018).

A Manutenção Preditiva tem sido um método eficaz no gerenciamento e monitoramento seguro de ativos industriais e, logo, tornou-se uma ferramenta de destaque na indústria moderna (FERRERO BERMEJO et al., 2019). A Manutenção Preditiva tem como princípio a medição das condições dos equipamentos para diagnósticos e prognósticos antecipados de qualquer comportamento de falha no futuro em ativos. Entre os tipos de manutenção, esse método apresenta uma série de benefícios, como por exemplo: uso otimizado de peças, custos reduzidos, aumento da vida útil do equipamento, segurança da planta, qualidade do produto e número reduzido de acidentes (LEE et al., 2013). Entretanto, ainda há limitações inerentes às tecnologias da Manutenção Preditiva na Indústria 4.0, como por exemplo: desafios na implementação dos sistemas legados nas empresas devido aos requisitos de integração das diversas tecnologias da Indústria 4.0 (LEE; KAO; YANG, 2014); processamento e armazenamento de grandes volume de dados oriundo do múltiplo sensoriamento dos ativos industriais (RUIZ-SARMIENTO; GALINDO; GONZALEZ-JIMENEZ, 2017) e, por fim, modelo preditivo capaz de se autoajustar com novos conjuntos de dados para generalizar suas ações em diferentes situações (HOERL; SNEE; DE VEAUX, 2014).

Diante das limitações apresentadas, o uso de dados rotulados com modos de falha anotados para realizar previsões com base em modelos de aprendizagem de máquina pré-treinados demonstram ser uma solução eficaz. Pois, o aprendizado de máquina é uma subárea da Inteligência Artificial que permite que um algoritmo seja desenvolvido a partir de dados, e não por meio de regras codificadas criadas por

programadores (conhecido como Inteligência Artificial Simbólica). Embora a Inteligência Artificial Simbólica tenha se mostrado adequada para resolver problemas lógicos bem definidos, como jogar xadrez, tornou-se intratável descobrir regras explícitas para resolver problemas difusos e mais complexos, como classificação de imagem, reconhecimento de fala, tradução de linguagem e detecção de anomalia (CHOLLET, 2021).

O aprendizado profundo é uma subárea de aprendizado de máquina que incorpora redes neurais em camadas sucessivas para aprender com os dados de maneira iterativa. Esse método atingiu um nível de atenção do público e investimento das indústrias nunca visto na história da Inteligência Artificial, principalmente devido a sua capacidade de aprendizagem em dados complexos. Para tanto, a aprendizagem profunda foi capaz de melhorar consideravelmente o estado da arte para diversas áreas do conhecimento, *e.g.* reconhecimento de fala, reconhecimento facial, detecção de objetos e entre outros (CHOLLET, 2021).

No entanto, dados rotulados são escassos, não confiáveis ou até inexistentes na indústria. Entretanto, a aplicação de aprendizagem profunda em dados complexos não estruturados e não rotulados foi pouco explorado na literatura. Nesses casos, é comum utilizar técnicas de aprendizagem de máquina não supervisionada para agrupamento dos dados em regiões de similaridade. No entanto, em cenário do mundo real, os dados não são estacionários e há desvios normais a serem considerados. Portanto, métodos de processamento de sinais e análises de séries temporais mais específicas podem exigir uma abordagem complementar. Um método mais flexível seria usar a aprendizagem profunda para recursos de autoaprendizagem. A autoaprendizagem é um tipo Inteligência Artificial de aprendizagem semi-supervisionada que pode se treinar usando dados não rotulados. Portanto, ao usar esse método, espera-se que o algoritmo extraia padrões diretamente dos dados amostrados brutos, o que evita a necessidade de um especialista para criá-los manualmente (ROSAPEREZ, 2020).

Além disso, quando se trata de uma abordagem para ambiente de produção industrial o desafio científico e tecnológico se torna ainda maior. Pois, existem muitas particularidades e variáveis que devem ser observadas, assim como: tempo de resposta e confiabilidade do modelo; diferentes tipos de equipamentos, frequências e modos de falha e volume e qualidade dos dados. Portanto, é importante desenvolver e avaliar

diferentes técnicas de aprendizado de máquina para alavancar as tecnologias de 4.0 no que tange as atuais lacunas científicas e tecnológicas.

Portanto, a hipótese desta tese consiste na modelagem da combinação de abordagens de aprendizagem de máquina não supervisionada e supervisionada para desenvolver um modelo semi-supervisionado de autoaprendizagem capaz de aprender com dados não rotulados. Portanto, a presente tese apresenta uma nova abordagem de desenvolvimento de aprendizagem profunda, visando viabilizar o desenvolvimento de modelos preditivos inteligentes em cenários em que não há dados rotulados.

1.1 Objetivos

1.1.1 Objetivo Geral

Pesquisar, estudar e desenvolver modelo de Autoaprendizagem (Self-Learning) de máquina profunda sem dados rotulados, baseado em aprendizagem de máquina semi-supervisionada, combinando as aprendizagens não supervisionada e supervisionada, para classificação de falhas em linhas de produção de O&G *offshore*, diminuindo, assim, a necessidade de grande esforço de especialistas humanos na tarefa de rotulação de dados, visando o aumento da eficiência e da automatização do processo de manutenção preditiva em ambientes industriais.

1.1.2 Objetivos Específicos

- Pesquisar e implementar aprendizagem de máquina não supervisionada para detectar padrões de normalidade e anormalidade em séries temporais multivariadas de produção de O&G oriunda de poços surgentes offshore;
- Pesquisar e desenvolver modelo de aprendizagem de máquina semi-supervisionada de autoaprendizagem profunda para automatizar a anotação de dados de séries temporais multivariadas;
- Classificar falhas na linha de produção O&G offshore utilizando aprendizagem de máquina semi-supervisionada e supervisionada;
- Refinar, testar e validar o modelo desenvolvido em dados reais da indústria de O&G offshore, apresentando e evidenciando sua aplicabilidade a casos reais.

1.2 Questões Norteadoras

1. Quais seriam as vantagens e limitações da aprendizagem de máquina não supervisionada e da aprendizagem de máquina supervisionada?
2. Seria possível desenvolver um modelo de aprendizagem de máquina semi-supervisionada com dados não rotulados?
3. Esse modelo seria capaz de aprender a classificar fenômenos físicos complexos a partir de dados de séries temporais multivariadas de um ativo industrial?
4. Qual seria o nível de desempenho desse modelo?
5. O desempenho seria equiparável a uma aprendizagem de máquina supervisionada?

1.3 Organização da Tese

O Capítulo 2 apresenta uma fundamentação teórica em manutenção industrial e inteligência artificial e uma revisão bibliográfica com o estado da arte no tema inteligência artificial aplicada na manutenção industrial. O Capítulo 3 introduz quatro manuscritos publicados em revistas e congressos que fazem parte da construção desta tese. O capítulo é dividido em dois subcapítulos: (i) aprendizagem de máquina não supervisionada e aprendizagem de máquina semi-supervisionada. Ambos os subcapítulos fazem uma contextualização do histórico da pesquisa e seus objetivos a serem alcançados por manuscrito. Em seguida, os artigos são apresentados na íntegra. No primeiro subcapítulo são apresentados três manuscritos e no segundo é apresentado um manuscrito. Por fim, o Capítulo 4 apresenta a conclusão da tese e propostas de trabalhos futuros.

2. REVISÃO BIBLIOGRÁFICA

2.1 Séries Temporais

Séries temporais são um conjunto de observações amostradas ao longo do tempo e são frequentemente utilizadas para representar graficamente a evolução de uma variável, como por exemplo, as séries que representam os índices de variação da bolsa de valores ou até mesmo as variações de temperatura que ocorrem ao longo do ano. Em outras palavras, as séries temporais têm a capacidade de informar sobre as ocorrências históricas até o momento da observação atual (MORETTIN; TOLOI, 2006).

As observações são feitas sequencialmente em uma frequência de tempo fixo. Uma série temporal pode ser representada como um vetor:

$$X = (x_1, \ldots, x_n) \tag{1}$$

Onde *n* é o comprimento da série temporal. Se houver várias aquisições de séries temporais simultaneamente para modelar o mesmo evento, o conjunto dessas séries temporais é uma série temporal multivariada representada como:

$$\chi = \{\chi^{(1)}, \ldots, \chi^{(m)}\} \tag{2}$$

Onde cada série $\chi^{(j)}$ é uma série temporal univariada para todo *j=1,...,m*. Uma série temporal multivariada pode ser representado como uma matriz $X \in R^{n \times m}$, onde m é o número de séries temporais univariadas em uma série multivariada. As linhas desta matriz são chamadas de observações, e as colunas (também chamadas de atributo) são as séries temporais (LINTONEN e RÄTY, 2019).

O entendimento, análise e processamento de séries temporais contribuem significativamente para a compreensão dos fenômenos geradores, como também em tomadas de decisões ou até mesmo em predições do comportamento futuro.

2.2 Manutenção Industrial

Falhas induzem a indisponibilidade de ativos e precisam ser identificadas precocemente e solucionadas para evitar o desligamento inesperado de uma linha de produção, portanto, a manutenção industrial é essencial para aumentar a eficiência e disponibilidade de ativos industriais. (WAN et al., 2017). A título de exemplo, Vafaei et al., (2019) propõem um sistema de alarme difuso para prever a degradação precoce de equipamentos em linha de produção de automóveis para reduzir custos com paradas súbitas. Wei et al., (2019), a fim de minimizar a taxa de custo média, propõem uma estratégia de manutenção baseada em condições para determinar a ação ideal utilizando o estado atual do sistema. Por outro lado, Dong et al., (2019) desenvolveram uma estrutura de prognóstico e gerenciamento de saúde para detectar a degradação do ativo no sistema de manufatura para otimizar o cronograma de manutenção e reduzir o custo, evitando tempos de inatividade desnecessárias e assistir tomadas de decisões.

O impacto financeiro de uma manutenção tardia pode representar entre 15 e 60% dos custos totais de operação de toda a produção (HARRMAN et al., 2017, MOBLEY, 2002). No entanto, as empresas não mensuram corretamente o valor despendido relacionado. Com isso, justifica-se a necessidade de estudos sobre como utilizar novas tecnologias que possam mudar esse cenário.

Na literatura existem diferentes nomenclaturas que se referem a estratégias de gerenciamento de manutenção, à vista disso, este trabalho classifica os procedimentos de manutenção da seguinte maneira:

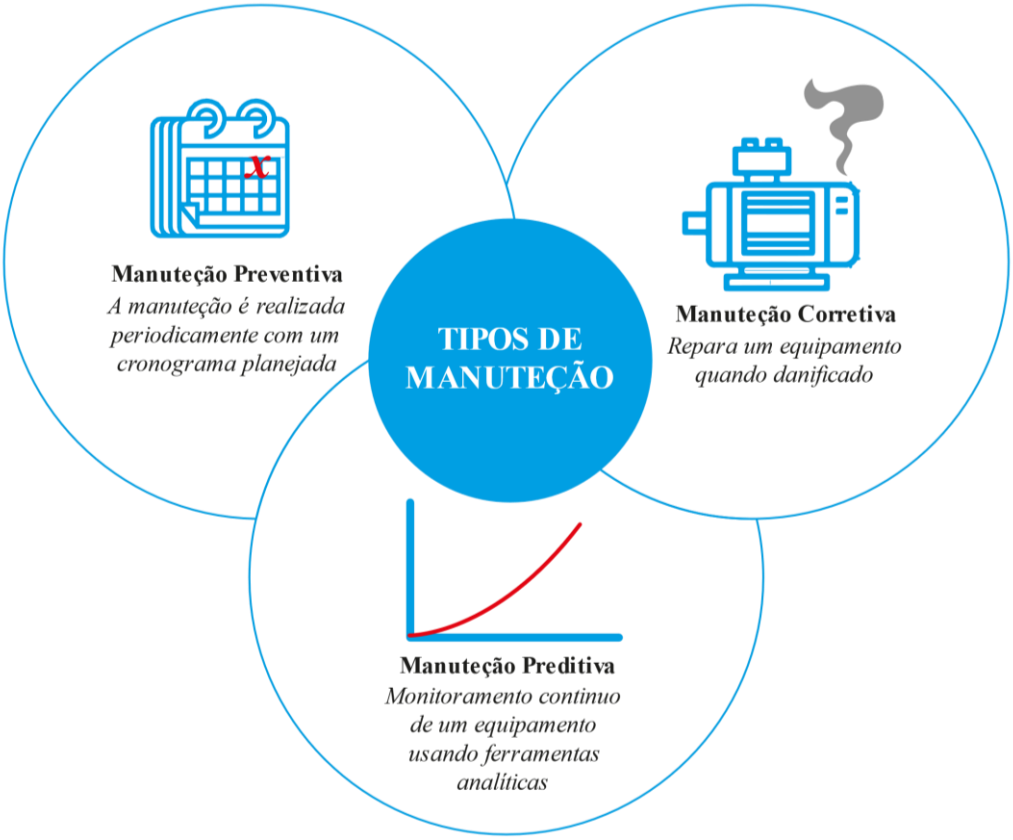
- Manutenção Corretiva (MC) ocorre apenas quando um equipamento para de funcionar. Essa é a estratégia de manutenção mais simples, pois é necessário a parada da produção para reparos ou substituição de mecanismos no sistema, agregando diretamente alto custo e risco de acidentes a esse tipo de procedimento;
- Manutenção Preventiva (MPv), Manutenção Baseada em Tempo ou Manutenção Programada é uma técnica de manutenção executada periodicamente e estaticamente com base em planejamentos cronológicos para antecipar falhas em ativos ou sistemas. Geralmente, é uma abordagem eficaz para evitar falhas. No entanto, devido ao planejamento estático, ações corretivas

desnecessárias são tomadas, agregando ao procedimento aumento nos custos operacionais;

- Manutenção Preditiva (MPd) usa métodos para determinar quando as ações de manutenção são necessárias. Baseia-se no monitoramento contínuo de uma máquina ou na integridade de um processo, permitindo que a manutenção seja executada na iminência da falha. Além disso, permite a detecção iminente de falhas por meio de análises com base em dados históricos (*e.g*. técnicas de Inteligência Artificial), fatores de integridade (*e.g.* aspectos visuais, desgaste, coloração diferente do original, entre outros), métodos de inferência estatística e abordagens de engenharia.

A Figura 1 fornece uma visão geral dos tipos de manutenção, cada tipo tem sua relevância. Ao optar pela MC, os setores atrasam as ações de manutenção, porém em contrapartida assumem maior o risco de indisponibilidade de seus ativos. A MPv antecipa as intervenções de manutenção, entretanto, problemas como substituição de componentes ainda em boas condições operacionais e paradas não programadas devido a uma degradação mais rápida do que o esperado ainda podem ocorrer (ERBE et al., 2005). Isso posto, é necessário tecnologias de manutenção, com base em dados de monitoramento da saúde do ativo, para melhorar as condições do equipamento, reduzir as taxas de falha, minimizar os custos de manutenção e maximizar a vida útil do equipamento (FERRERO BERMEJO et al., 2019). Devido a essa necessidade, a estratégia da MPd ganhou destaque entre os outros métodos de manutenção e está atraindo atenção na era da Indústria 4.0 devido à sua capacidade de otimizar a aplicação e gerenciamento de ativos (KUMAR; CHINNAM; TSENG, 2019). Suas vantagens incluem: uso otimizado de peças, custos reduzidos de material e mão de obra, aumento da vida útil do equipamento, segurança da planta, qualidade do produto (fabricação de falha quase zero), número reduzido de acidentes e entre outros (LEE et al., 2013).

Figura 1. Visão geral dos tipos de manutenção.



Fonte: Adaptado de Carvalho et al., (2019).

2.2.1 Manutenção Preditiva

O método de MPd tem como estratégia a medição do estado de saúde dos equipamentos para predizer a tendência de comportamento do ativo, visando identificar se se há algum risco de falha. Por conseguinte, a ferramenta tem como objetivo auxiliar a tomada de decisão da equipe de gestão de manutenção para impedir o acaso de paradas não programadas e aumentar a disponibilidade do equipamento (CARVALHO, et al., 2019).

As ferramentas preditivas podem atuar no monitoramento das condições dos equipamentos para fins de diagnóstico e prognóstico. O diagnóstico de falhas concentra-se na detecção, isolamento e identificação de falhas quando elas ocorrem. No entanto, o prognóstico prevê falhas antes que elas ocorram. As abordagens de prognósticos são geralmente mais desejadas, no sentido de que os prognósticos podem evitar falhas, além de auxiliar no planejamento da gestão da manutenção com peças sobressalentes prontas e recursos humano disponível para o devido tratamento do problema e, assim, economizar custos de manutenção não planejados. No entanto, os prognósticos não

podem substituir completamente os diagnósticos, pois, na prática, sempre existem algumas falhas que não são previsíveis. Além disso, os prognósticos, como qualquer outra técnica de previsão, estão suscetíveis a erros na previsão de falhas. Portanto, no caso de uma previsão mal-sucedida, o diagnóstico pode ser uma ferramenta complementar para fornecer suporte à decisão de manutenção. Ainda, o diagnóstico também é útil para melhorar o prognóstico, na medida em que as informações de diagnóstico podem ser úteis para preparar dados de eventos mais precisos e, portanto, aprimorar o modelo preditivo de prognóstico (JARDINE; LIN; BANJEVIC, 2006).

O diagnóstico de falha em máquinas é uma ação de classificação ou reconhecimento de padrões. Tradicionalmente, esse trabalho é realizado manualmente com ferramentas gráficas auxiliares. No entanto, o reconhecimento de padrões de forma manual requer experiência na área específica do sistema a ser diagnosticado, necessitando de mão de obra altamente treinada e qualificada. Desse modo, cria-se um interesse na indústria de automatizar o reconhecimento de padrões, esse interesse pode ser alcançado através da classificação de sinais com base nas informações e/ou recursos extraídos dos sinais emitidos pela máquina (*e.g*. vibração, temperatura, pressão etc.) (KORBICZ, J.M. KOŚCIELNY, 2005).

No que tange às estratégias de prognóstico, a abordagem mais amplamente utilizada na literatura é a previsão do tempo sobressalente antes que ocorra uma ou mais falhas no equipamento (DE OLIVEIRA DA COSTA et al., 2020; LI et al., 2020; RUIZ-SARMIENTO et al., 2020). O tempo restante para observar uma falha é conhecido como vida útil remanescente (VUR). Em ambientes de alta periculosidade e insalubridade (*e.g.* plataforma offshore de petróleo e gás, usina nuclear, refinaria etc.) é altamente desejável prever o tempo de operação sem falha de um ativo ou processo até a próxima inspeção, dado a condição de integridade atual e o histórico de degradação. Sobretudo, em qualquer ambiente industrial é desejado uma previsão precisa da VUR e da probabilidade de uma máquina operar sem falhas até a próxima inspeção, auxiliando a equipe de manutenção a determinar se o intervalo de inspeção é apropriado ou não.

Entretanto, ainda há desafios inerentes a implantação das tecnologias da MPd em ambiente operacional, principalmente devido aos requisitos de integração das diversas tecnologias da Indústria 4.0 (LEE; KAO; YANG, 2014). Os principais desafios são: processar grandes repositórios de dados de séries temporais, provenientes de sensores

ruidosos instalados nos ativos, a fim de serem adquiridos digitalmente de forma útil para as análises preditivas (RUIZ-SARMIENTO; GALINDO; GONZALEZ-JIMENEZ, 2017) e desenvolver um modelo preditivo, baseado em dados, que possa estimar e executar uma tomada de decisão eficaz e em um curto intervalo de tempo. Além disso, o modelo preditivo deve ser capaz de reaprender com novos conjuntos de dados e generalizar suas ações em diferentes situações (HOERL; SNEE; DE VEAUX, 2014). Esses problemas precisam ser enfrentados não apenas na MPd, mas em diversos setores da indústria (e.g.: saúde e meio ambiente) (NIKOLIC et al., 2017).

2.3 Inteligência Artificial

A Inteligência Artificial nasceu na década de 1950 por pesquisadores da área de ciência da computação (recém-criada), com a proposta de automatizar tarefas intelectuais normalmente realizadas por humanos. A Inteligência Artificial é um campo geral que engloba aprendizado de máquina e aprendizado profundo, mas também inclui muitas outras abordagens que não envolvem nenhum aprendizado. A título de exemplo, os primeiros programas de xadrez envolviam apenas regras codificadas criadas por programadores e não se qualificam como aprendizado de máquina. Por muitos anos, especialistas pensavam que a Inteligência Artificial de nível humano poderia ser alcançada fazendo com que os programadores criassem um conjunto suficiente grande de regras explícitas para manipular o conhecimento. Essa abordagem foi o paradigma dominante na Inteligência Artificial na década de 1950 até o final da década de 1980. Atingiu seu auge de notoriedade na década de 1980 através da competência sistemas especialistas (CHOLLET, 2021).

Em 1950 o pesquisador Alan Turing concluiu que os computadores poderiam ser capazes de aprender. Por meio do seu artigo “Computing Machinery and Intelligence”, introduziu o teste de Turing e conceitos que se tornaram fundamentos da Inteligência Artificial. Desse modo, a aprendizagem de máquina nasceu com a hipótese que um computador deveria aprender automaticamente regras observadas em dados, ao invés de programadores criarem regras de processamento de dados manualmente. Essa hipótese gerou oportunidades para um novo paradigma de programação (CHOLLET, 2021).

Na programação clássica (paradigma da Inteligência Artificial), os humanos inserem regras e dados a serem processados de acordo com essas regras, e obtêm respostas. Com a aprendizagem de máquina, os humanos inserem dados, bem como as respostas esperadas dos dados, e saem as regras (GUTZWILLER, et al, 2017).

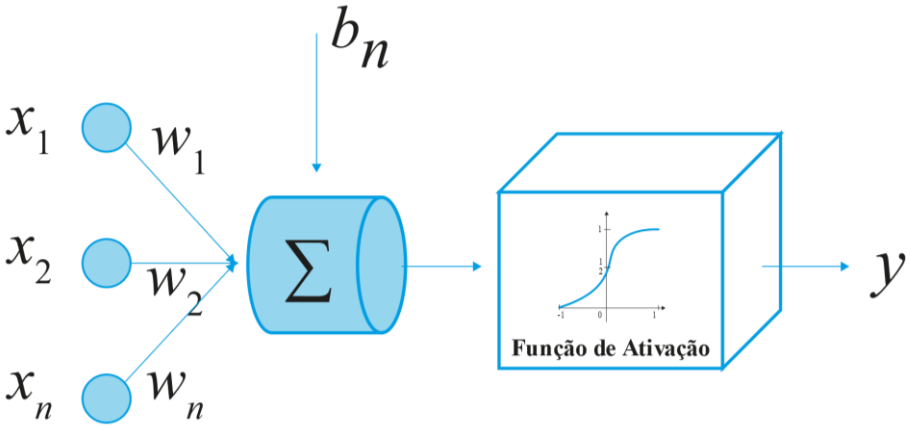
A aprendizagem de máquina (inserida na Inteligência Artificial) surgiu na década de 2010 e é uma tecnologia que ganhou muito destaque nos últimos anos. Com diversas aparições na mídia e publicações científicas, tem sido conceituado por cientistas como uma tecnologia fundamental para a Quarta Revolução Industrial. São diversas promissoras aplicações em desenvolvidas, como por exemplo chatbots, carros autônomos, assistentes virtuais, reconhecimento facial, detecção de objetos etc.

Os métodos de aprendizagem de máquina foram projetados para aprenderem a executar tarefas com base em uma grande quantidade de dados. A sua aplicação tem revelado ganhos na literatura, como proporcionar redução de risco operacional, maior confiabilidade e disponibilidade dos ativos e, consequentemente, maior vantagem competitiva (ALI, 2018). Os elementos da inteligência artificial incluem raciocínio, representação e planejamento do conhecimento, aprendizagem de máquina, RNA, aprendizado profundo, reconhecimento de padrões, soluções robóticas etc. (DANIYAN et al., 2020).

A RNA (inseridos na aprendizagem de máquina) é um modelo computacional desenvolvido inspirado na estrutura do cérebro humano (BISWAL; SABAREESH, 2015). Em geral, a capacidade cognitiva de um ser humano é o resultante das complexas ligações neurais do sistema nervoso, sendo o neurônio o elemento celular mais básico dessa estrutura. Essas células estão ligadas paralelamente e persistentemente com diversas outras através dos dendritos e dos axônios (DA SILVA FILHO; ABE, 2000).

McCulloch e Pitts (1943) desenvolveram o primeiro modelo de neurônio artificial, que ficou conhecido como Perceptron Simples por possuir na sua estruturada uma única camada de uma RNA (HAYKIN, 2008). A Figura 2 ilustra o neurônio de McCulloch e Pitts, ou perceptron simples.

Figura 2. Neurônio de McCulloch e Pitts.



A representação matematicamente do neurônio é realizada através do somatório de multiplicações ponderadas. Admitindo que as variáveis X são os estímulos do ambiente, e W são os valores dos pesos sinápticos, o resultado do somatório das multiplicações resultarão em um sinal onde sua propagação e intensidade são ponderadas pela função de ativação. O viés ou bias, representado por bn é invariável perante aos estímulos externos, entretanto proporciona um aumento ou diminuição no sinal de entrada da função de ativação (HAYKIN, 2008; LUGER, 2014).

O conhecimento agregado pela estrutura é armazenado nos pesos sinápticos, ou seja, em um processo de aprendizado, os pesos são atualizados até que a saída deste neurônio esteja dentro das condições determinadas. Porém, em um primeiro momento esses valores são gerados aleatoriamente.

A função de ativação é a formulação matemática inspirada na sinapse do cérebro humano. A função Heaviside ou função degrau, usada no neurônio de McCulloch e Pitts, tem a principal função de avaliar se o sinal proveniente dos estímulos sinápticos tem a permissão de propagar para as camadas posteriores. A estrutura da RNA pode ser descrita matematicamente da seguinte forma:

$$u_k = \sum_{j=1}^{m} w_{kj}.x_j \quad (3)$$

$$y_k = \varphi(u_k + b_k) \quad (4)$$

Na Equação (1), u_k é a representação do combinador linear que alimenta a Equação (4), onde x_j são os estímulos e w_{kj} os pesos sinápticos, sendo φ a função de ativação e b_k o bias. A função de ativação de Heaviside é dada pela Equação (5).

$$\varphi(z) = \{1 \; se \; z \geq 0 \; 0 \; se \; z < 0 \tag{5}$$

Onde z é o sinal de saída do neurônio, sendo $z = 1$ para os casos em que o resultado das multiplicações seja igual a 1 ou maior, caso contrário $z = 0$.

Entretanto, uma estrutura neural baseada em uma única célula se demonstrou limitada para problemas não lineares e que envolvem dados de alta dimensionalidade. Para tanto, surgiram as RNAs como um conjunto de células neurais artificiais (também podem ser chamadas de nós), que formam uma rede com várias conexões e camadas. As RNAs consistem em elementos de processamento conectados em uma estrutura de multicamadas que permite o modelo aproximar uma função não linear, através da utilização de múltiplos dados de entrada e saída. Um elemento de processamento compreende um nó ou neurônio e um peso. A RNA aprende a função desconhecida ajustando seus pesos com observações dos dados de entrada e saída. Esse processo é geralmente chamado de treinamento (JARDINE; LIN; BANJEVIC, 2006).

A RNA é um dos algoritmos de aprendizagem de máquina mais procurados e aplicados, possuem propostas em muitas aplicações industriais, incluindo sensoriamento virtual (GOMES SOARES; ARAÚJO, 2015) e controle preditivo (SHIN; JUN; KIM, 2018). Suas principais vantagens constituem nas seguintes abordagens: não é necessário nenhum conhecimento especializado para resolução de problemas, pois são baseadas apenas em dados históricos; no caso de abordagens de aplicação específica, o modelo pode ser aplicado em tempo real sem precisar alterar sua arquitetura a cada atualização e possui rápido processamento para tomada de decisão com base em grande volume de dados. No entanto, existem algumas limitações das RNAs, a título de exemplo, na fase de treinamento pode ser demorado e geralmente requer alto custo computacional. Além disso, a maior fragilidade de uma RNA são os dados, pois é essencial um conjunto de dados volumoso e representativo do sistema para a aprendizagem adequada do modelo.

O aprendizado profundo é um método de RNA puramente de várias camadas. Nesse método, com base nos dados, a aprendizagem ocorre em diferentes níveis de hierarquia entre as camadas, permitindo que o algoritmo desempenhe funções complexas por meio do mapeamento dos dados de entrada e saída. Dessa forma, desde 2006, o aprendizado profundo tornou-se uma direção de pesquisa em rápido crescimento, redefinindo as performances de ponta em uma ampla gama de áreas, como reconhecimento de objetos, segmentação de imagens, reconhecimento de fala, tradução automática e ferramentas preditivas (ZHAO et al., 2019). Entretanto, os modelos de aprendizado profundo, sejam métodos robustos, exigem conhecimento especializado na seleção dos diferentes níveis de hierarquia para uma aplicação específica (CARVALHO et al., 2019).

2.3.1 Rede Neural Perceptron Multicamadas

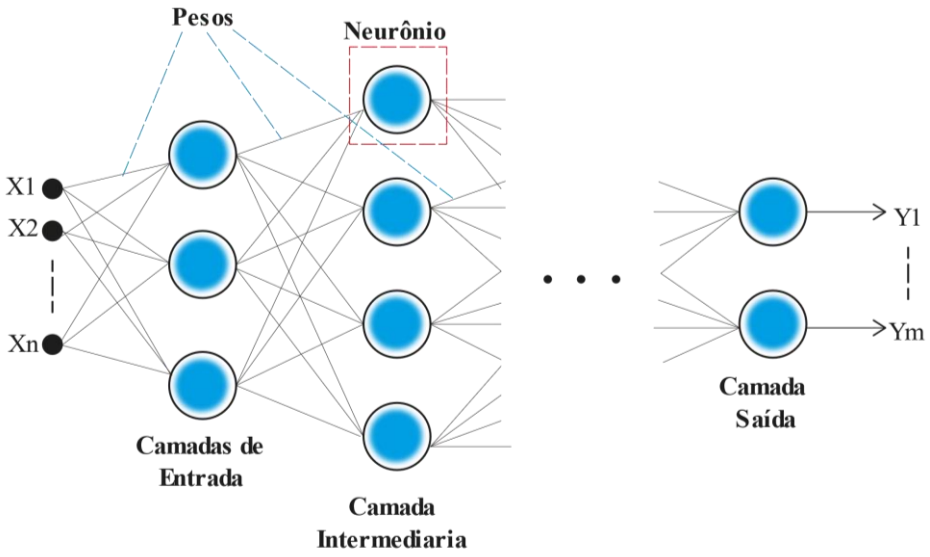
A Rede Neural Perceptron Multicamada, do inglês *Multilayer Perceptron* (MLP) é a topologia de rede neural artificial mais simples existente. É basicamente a combinação de múltiplos perceptrons, que são as unidades básicas dos neurônios que operam matematicamente. Se combinarmos a Equação (3)e Equação (4), teremos:

$$y_{x_w} = \varphi\left(\sum_{i=1}^{m} w_i x_i + b\right) \tag{6}$$

onde y_{x_w}é a saída do perceptron, φ é a função de ativação, x_ié um atributo ou característica do vetor de dados de entrada x de tamanho m, $w_i$representa cada peso do vetor de peso w, e b é o viés (ou bias). Em resumo, o objetivo é determinar se a saída da função φ dispara (ou seja, retorna um valor diferente de zero) após somar o produto das características de entrada e os pesos, que são os parâmetros que são aprendidos automaticamente por meio de um algoritmo de aprendizado supervisionado.

Uma MLP é geralmente composta por três ou mais camadas totalmente conectadas entre si. A sua estrutura é necessária ao menos três camadas, sendo essas uma camada de entrada, uma camada intermediária (ou oculta) e uma camada de saída. A Figura 3 apresenta a estrutura típica de uma arquitetura MLP

Figura 3. Diagrama esquemático representando a estrutura básica de uma arquitetura MLP. Cada círculo corresponde a um perceptron.



De forma geral, uma MLP tem como principais parâmetros o número de camadas, as funções de ativação e o número de neurônios em cada camada, com uma topologia flexível. A definição do número de camadas e neurônios é variável, e a melhor composição é específica do problema (ALVES, NASCIMENTO e MOREIRA, 2019).

O número de saídas depende dos requisitos de cada aplicação e permite a previsão multi-passo e multivariada. A configuração desses atributos para cada um dos problemas é escolhida principalmente empiricamente. Além disso, todas as conexões entre as camadas MLP são do tipo *forward*, ou seja, é necessário um algoritmo de retropropagação para o sinal propagar para trás (AHANI, SALARI e SHADMAN, 2019).

Devido à sua simplicidade e capacidade de resolver problemas complexos, a MLP tem sido empregado em muitos trabalhos para prever componentes da poluição do ar (JUNIOR, NASCIMENTO e MOREIRA, 2020; ALVES, NASCIMENTO e MOREIRA, 2019; WANG, et al., 2019; AHANI, SALARI e SHADMAN, 2019; ZHANG, et al., 2018; ZUCATELLI, et al., 2021). Embora, não tenha sido projetado especificamente para lidar com a previsão de séries temporais.

Uma RNA emprega uma função de ativação para simular uma rede neural natural. Entretanto, ao contrário do Perceptron simples de McCulloch e Pitts, que usa uma única função de ativação degrau, as estruturas multicamadas não estão limitadas a somente uma função. Portando, outras funções de ativação podem ser utilizadas para restringir o sinal de saída, a título de exemplo a função Sigmóide.

A função Sigmóide é uma das mais utilizadas nas RNAs no que diz respeito ao limiar de ativação dos neurônios. A função possui uma curva característica em forma de S, mapeia a reta numérica em um subintervalo de comprimento finito de [0,1] e apresentando achatamento em suas extremidades (não é totalmente linear). A sua forma mais comum é dada pela Equação (6), onde λ é o parâmetro de inclinação da função.

$$\varphi(z) = \frac{1}{1 + e^{-\lambda z}} \tag{7}$$

2.3.2 Redes Neurais Recorrentes

As Redes Neurais Recorrentes (RNR) são conhecidas pela dinâmica de transmissão do sinal com a propagação à frente, onde os sinais dos neurônios são propagados naturalmente para as camadas posteriores (MANTOVANI, 2011).

O principal ganho da RNR é a retenção de informação com o avanço das iterações de ajuste de peso. Contudo, a RNR é mais custosa computacionalmente, uma vez que há um aumento na complexidade da formulação do sinal de saída do neurônio (LUGER, 2014; MANTOVANI, 2011).

A principal diferença entre RNR e redes neurais básicas é o estabelecimento de conexões ponderadas entre neurônios, conectadas pelo estado oculto, que transporta informações de etapas imediatamente anteriores e sobrescreve a cada etapa sem controle especial ou seletivo sobre o que é memorizado ou esquecido (QIAO, et al., 2019).

A RNR é um tipo de RNA criada para lidar com dados sequenciais distribuídos no tempo e no espaço. No entanto, é mais provável que essa estrutura sofra com o problema do gradiente de fuga, uma característica dos métodos de aprendizado baseados em gradiente, que pode até impedir o treinamento da rede neural. Portanto, essa fragilidade limita a capacidade das RNRs tradicionais de representar corretamente relacionamentos de longo prazo presentes em séries temporais ou outros dados sequenciais.

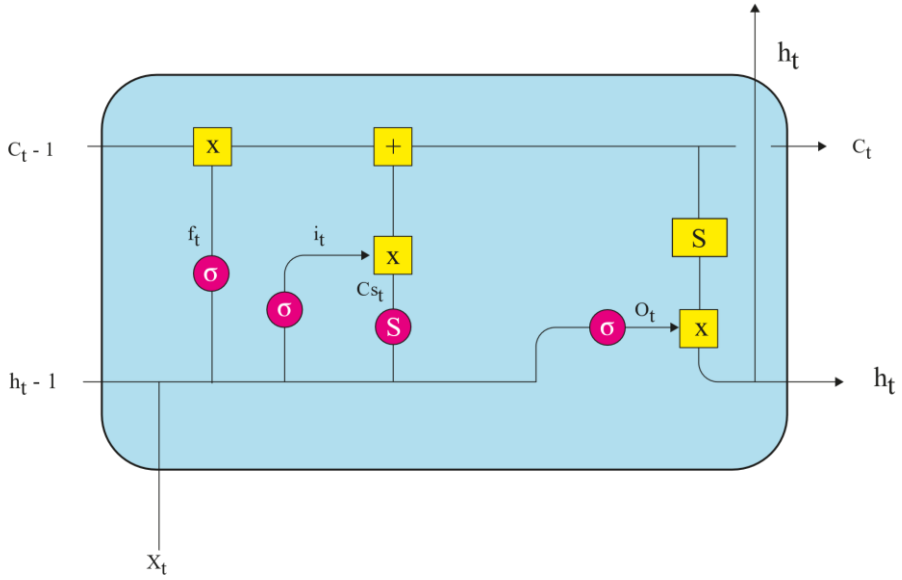
2.3.3 Redes Long Short-Term Memory

Hochreiter e Schmidhuber desenvolveram em 1997 as redes neurais de memória de longo prazo, do inglês *Long Short-Term Memory* (LSTM). (CORTE; SANTOS; CASANOVA, 2019). O LSTM é um modelo estruturado na forma de cadeias, fazendo as conexões com a próxima célula por meio do estado da célula e do estado oculto. Tem como principais partes o estado da célula, a porta de entrada, a porta de esquecimento e a porta de saída (LE, et al., 2019).

O estado da célula é um tipo de memória seletiva do passado, e os portões funcionam alternadamente para controlar o fluxo de dados no estado da célula. A porta de entrada processa a entrada e decide se é um dado relevante para alterar a memória disponível no estado da célula. O portão de esquecimento decide quais dados devem ser mantidos da saída mais antiga, e o portão de saída controla o fluxo do estado oculto, decidindo quais informações devem ser transportadas para a próxima célula (CORTE; SANTOS; CASANOVA, 2019).

Para treinar a rede neural LSTM, os dados de entrada precisam ser tridimensionais por causa da adição do lookback, que representa quantos passos para trás serão usados para prever o próximo passo ou variáveis. Portanto, o LSTM tem a capacidade de aprender relações temporais e aumentar os resultados de previsão, sendo uma ferramenta interessante para lidar com séries temporais. A Figura 4 mostra como uma célula LSTM é estruturada. Todas as linhas transportam dados que podem passar por operações pontuais, camadas de RNA, concatenações e replicações.

Figura 4. Estrutura celular LSTM. Setas, quadrados e círculos representam fluxo de dados, operações pontuais e funções de ativação, respectivamente



Fonte: Adaptado de Junior, Nascimento e Moreira (2020).

As funções sigmoides (σ) e tangente hiperbólica (S) são as responsáveis por quantificar o volume de informação que permanecerá na célula, assim com as informações que serão descartadas. Esse comportamento se repete em todos os estados temporais da LSTM.

A primeira etapa de uma célula LSTM é definir quais informações serão descartadas para o estado atual. O portão responsável é o Portão de Esquecimento dado pela Equação (8), onde $f(t)$ é a porta de esquecimento, σ é função Sigmoid, W_f é o peso da matriz de peso do x neurônio, h_t é a célula de saída em x, x_t é a entrada em x, b_x é a matriz do bias correspondente a x.

$$f(t) = \sigma(W_f.[h_{t-1}, x_t] + b_f) \quad (8)$$

A segunda etapa é executada através da utilização do Portão de Entrada, onde o seu objetivo é definir quais informações irão compor o estado atual da célula, definindo consequentemente quais desses valores serão atualizados. A formulação matemática é dada pelas Equações (9) e (10), onde i_t é a porta de entrada, $\tilde{C}_{S_t}$ e C_t são os candidatos

para o estado da célula e o respectivo estado da célula em um determinado passo de tempo t, respectivamente.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] \ + \ b_i) \tag{9}$$

$$\tilde{C}_{S_t} = S(W_c.[h_{t-1}, x_t] \ + \ b_c) \tag{10}$$

Uma vez definido o estado atual da célula, torna-se necessário atualizá-la. Em outras palavras, o estado anterior C_{t-1} será atualizado - Equação (11)- para o estado C_t, baseado nas informações que foram admitidas e excluídas pelos portões de entrada e de esquecimento.

$$C_t = f_t.C_{t-1} + i_t * \tilde{C}_t \tag{11}$$

A última etapa é dada pelo Portão de Saída - Equações (12) e (13). Neste momento executa-se uma filtragem através da aplicação de uma função sigmoidal. Esse procedimento determina qual informação será propagada do estado atual da LSTM para os estados posteriores, bem como para as camadas seguintes. A aplicação da tangente hiperbólica estabelece a característica final da saída da célula.

$$o_t = \sigma(W_o.[h_{t-1}, x_t] \ + \ b_o) \tag{12}$$

$$h_t = o_t.S(C_t) \tag{13}$$

Onde o_t é a porta de saída.

2.3.4 Redes Neurais Convolucionais

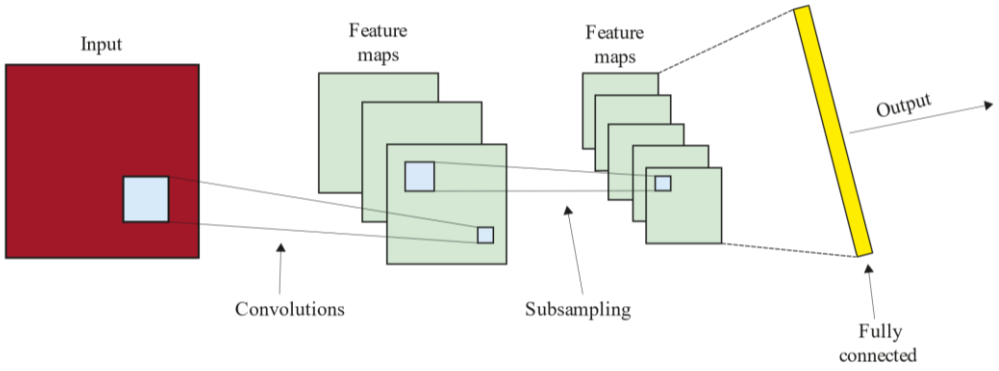
As Redes Neurais Convolucionais, do inglês *Convolutional Neural Network* (CNN) é um tipo de RNA que aprende padrões de dados por meio da aplicação de convoluções, visando aprender filtros que extraem as principais características dos dados para realizar uma tarefa específica. Os primeiros algoritmos surgiram na década de 1980 com Yann LeCun. A estrutura criada foi nomeada como LeNet5 e suas primeiras atribuições eram o reconhecimento de dígitos manuscritos (LECUN et al., 1998; HAYKIN, 2008).

As CNNs têm uma grande habilidade de aprender relações espaciais e temporais a partir de dados, apesar de ser pioneira no processamento de imagens. Seus benefícios podem ser explorados e avaliados para previsões de séries temporais, assim como

redimensionar e detectar automaticamente novos elementos e padrões dos dados. Além disso, as camadas de Pool reduzem o tamanho da sequência de entrada, seguidas pela aplicação de camadas de nivelamento que ajustam a forma dos dados para entrar em um MLP para concluir a tarefa especificada (JUNIOR, NASCIMENTO e MOREIRA, 2020).

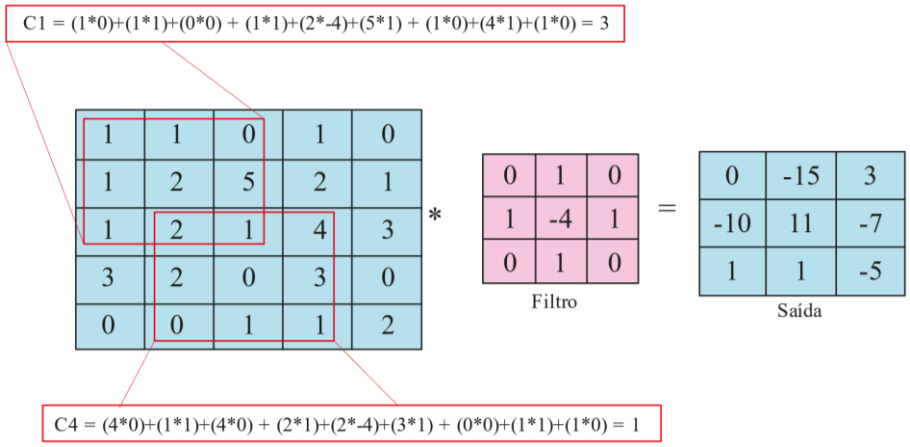
Assim como na LSTM, o lookback também é necessário nos dados de entrada de uma CNN (ARENA, et al., 2003). As interligações das redes neurais convolucionais não difere das redes multicamadas perceptron. Em outras palavras, a estrutura é totalmente conectada, desde as camadas convolutivas até a camada de saída. O modelo tradicional de redes convolutivas apresenta três regiões definidas como: convolução, Pooling (subsampling) e região de camadas totalmente conectadas como mostra a Figura 5.

Figura 5. Típica Estrutura de uma Rede Convolucional.



As camadas de convolução tem a função de extrair características do conjunto de entrada com a utilização de filtros que apresentam baixas dimensões. Quanto maior for o número de filtros utilizados, mais características podem ser extraídas para criação de um mapa, entretanto, também será maior o consumo de memória e processamento computacional. A extração da característica na etapa de convolução é feita através de um somatório do produto por variados tipos de filtros como mostra a Figura 6 (FARIA, 2018; ARAÚJO et al., 2017).

Figura 6. Procedimento de Convolução.



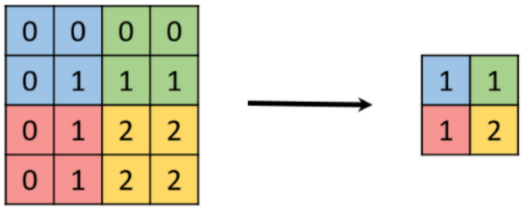
Fonte: Adaptado de Faria (2018).

As janelas de convolução, com tamanho das dimensões do filtro, são deslocadas por toda a matriz de entrada, sendo o sentido do deslocamento dependente da direção desejada. O deslocamento é controlado por um parâmetro conhecido como *stride* ou passo. Quando o *stride=1*, a janela é deslocada em uma linha ou em uma coluna. Essa operação promove uma redução da dimensionalidade da matriz resultante, no caso da não aplicar a técnica de zero-padding, em que consiste em implementar zeros na borda da matriz de entrada. O zero-padding garante que a matriz resultante da operação de convolução tenha as mesmas dimensões da matriz de entrada (ARAÚJO et al., 2017; FARIA, 2018).

Como saída da convolução, tem-se um mapa de características contendo as informações mais relevantes dos dados de entrada. Comumente, a camada de Pooling se posiciona imediatamente após uma operação de convolução. O método reduz a dimensionalidade, realça as características encontradas e ainda contribui para a redução de overfitting (ARAÚJO et al., 2017; FARIA, 2018).

O Pooling é desenvolvido com base em operações matemáticas simples, como por exemplo: a extração do maior valor de quadrantes da matriz (Maxpolling) como mostra a Figura 7.

Figura 7. Operação de Polling (Maxpolling).



Fonte: Araújo et al. (2017).

A Figura 7 mostra a redução de uma matriz 4x4 para uma matriz 2x2 realçando as características de interesse. As cores azul, amarelo, vermelho e verde representam os quadrantes de atuação do Maxpolling.

Uma vez concluídas as etapas convolutivas da rede, todas as informações extraídas são propagadas para as camadas de neurônios densamente conectados. São nessas camadas que as informações são processadas. As características extraídas das camadas de convolução entram como estímulos para os neurônios.

As seguintes equações descrevem matematicamente a camada de convolução:

$$G[m,n] = (f * k)[m,n] = \sum_{j} \sum_{i} k[j,i] f[m-j, n-i] \tag{14}$$

$$C^{l} = \sigma^{l}\left(V^{[l]}\right) \tag{15}$$

$$V^{l} = K^{l} \cdot C^{[l-1]} + b^{l} \tag{16}$$

onde G é o mapa de características, f é a entrada, k, m e n representam o kernel, e as linhas e colunas da matriz de resultados, respectivamente. Os índices j e i estão relacionados ao kernel, l é o índice da camada, V é o valor intermediário, K é o tensor que possui filtros ou kernels, C é o resultado da convolução, b é o bias e σ é a correspondente função de ativação.

Além disso, uma camada de agrupamento pode ser empregada para reduzir a dimensionalidade do tamanho da saída da convolução, extraindo o valor máximo (MaxPooling) ou médio (AvgPooling) dos kernels/filtros extraídos dentro de uma janela de tamanho fixo, diminuindo assim o poder de processamento necessário para o treinamento da rede.

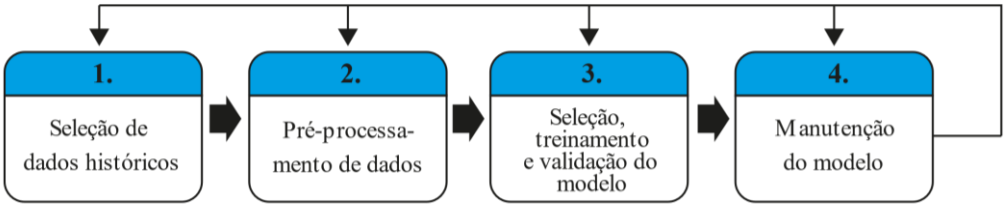
2.4 Inteligência Artificial na Manutenção Preditiva

As técnicas de aprendizado de máquina têm sido cada vez mais aplicadas, a fim de diagnóstico e prognóstico de máquinas industriais, e têm se mostrado bons desempenhos em relação às abordagens convencionais. Em destaque, as abordagens de aprendizado de máquina têm a capacidade de manipular dados de alta dimensão e reconhecer padrões ocultos nos dados em um ambiente complexo e dinâmico (CARVALHO et al., 2019).

De modo geral, a aplicação da inteligência artificial na área da MPd envolve quatro etapas (Figura 8): (i) seleção de dados históricos, (ii) pré-processamento de dados, (iii) seleção, treinamento e validação do modelo e (iv) manutenção do modelo.

A etapa de seleção de dados históricos planeja como os dados serão coletados e armazenados em repositórios seguros para que dados representativos do sistema sejam adequadamente armazenados para o desenvolvimento do modelo preditivo. Na MPd, essa etapa também é chamada de aquisição de dados e visa obter dados relevantes para o estado de integridade do sistema (JARDINE; LIN; BANJEVIC, 2006). A etapa de pré-processamento de dados otimiza e transforma os dados para que possam ser utilizados com eficiência pelo modelo. Essa etapa inclui a transformação de dados (*e.g.* normalização ou reescalonamento), limpeza de dados (*e.g:* tratamento de dados ausentes e remoção de outlier) e redução de dados (*e.g.* redução de dimensionalidade e redução de numerosidade). Em resumo, essa etapa manipula e analisa os dados coletados para melhor compreensão e interpretação dos dados, visando o aprendizado do modelo de inteligência artificial. A etapa de seleção, treinamento e validação do modelo envolve a seleção adequada do modelo de inteligência artificial, o treinamento do modelo e a validação do modelo que se refere ao procedimento que avalia a generalização do aprendizado do modelo. Essa etapa pode ser chamada de tomada de decisão de manutenção e tem como objetivo decidir o melhor algoritmo para a aplicação. A etapa de manutenção do modelo visa manter o desempenho do modelo ao longo do tempo. Isso ocorre porque os processos industriais podem mudar com o tempo, levando ao decaimento do desempenho do modelo (SOARES, 2015). A Figura 8 ilustra as etapas metodológicas que englobam as aplicações de inteligência artificial na área da MPd.

Figura 8. As principais etapas para aplicação de modelos de aprendizagem de máquina na Manutenção Preditiva.



Fonte: Adaptado de CARVALHO et al., 2019.

É possível obter maior eficiência na manutenção, operação e gerenciamento de ativos industriais através do aumento da conectividade em rede e da implantação de ferramentas de inteligência artificial. Portanto, através dos recentes avanços tecnológicos, alinhados com a Indústria 4.0, a inteligência artificial tem se consolidado cada vez mais como um potencial ferramenta para auxiliar gestores na tomada de decisão.

Os modelos de RNA geralmente usam a aprendizado supervisionada que requerem dados de entrada rotulados, *i.e.*, conhecimento a priori sobre o destino ou a saída desejada. Por exemplo, uma prática comum de treinar um modelo de RNA é usar um conjunto de dados experimentais com comportamentos de falha conhecidas (rotuladas). Esse processo de treinamento é conhecido como aprendizado de máquina supervisionado. Em cenários operacionais da indústria, leva-se muito tempo para adquirir uma quantidade de amostras suficientes para treinar uma RNA que possa, ao menos, prever a grande maioria das possíveis falhas que possam ocorrer (YUAN; LIU, 2013). Portanto, a maioria dos dados é sem rótulo, o que pode inclusive inviabilizar o desenvolvimento de métodos de aprendizagem de máquina supervisionados.

Em contrapartida, o aprendizado não supervisionado não requer informações com conhecimento da saída de dados desejado, logo, tem sido um ponto de acesso de pesquisa para extrair recursos úteis de dados brutos não rotulados. Os algoritmos não supervisionados aprendem a si mesmos usando as informações disponíveis. Essa abordagem é mais adequada para problemas em que os mapeamentos de entrada-saída de dados não estão disponíveis, *i.e.*, os dados não são rotulados, mas onde qualquer par de entrada-saída possa ser avaliado (ALDRICH; AURET, 2013).

A aprendizagem de máquina não supervisionada pode ser usada para extrair dos dados recursos úteis para distinção de diferentes tipos de falhas. Em vez de alocar os recursos de um operador humano, o aprendizado não supervisionado é inteligente e independente do conhecimento prévio das técnicas de processamento e do conhecimento da fenomenologia do sistema.

Portanto, métodos de aprendizagem de máquina não supervisionada, sem necessidade de dados rotulados, é desejado para automaticamente distinguir os diferentes comportamentos defeituosos e alavancar a anotação de dados (LIU et al., 2018). A título de exemplo, Nguyen et al., (2019) propuseram a RNA não supervisionada Autoencoder (AE) em que a fase de autoaprendizagem do AE gera um vetor final para o treinamento do classificador softmax para diagnóstico de falha de rolamento. Wang et al., (2019) usaram o algoritmo não supervisionado de agrupamento Fast K-means e função de base radial, do inglês *Radial-basis function* (RBF) para diagnóstico de falha composta de máquinas rotativas. Wang e Too (2002) aplicaram as redes neurais não supervisionadas Mapas de Kohonen, tipo de *Self-Organized Map* (SOM), e aprendizado por quantização vetorial (AQV) na detecção de falhas de máquinas rotativas. Liu et al., (2018), propuseram AE para suporte ao diagnóstico não supervisionado de falhas de rolamentos.

O uso de ferramentas de inteligência artificial tem como proposta ajudar os gestores na tomada rápida de decisões em relação à manutenção preditiva antes que os sistemas apresentem redução significativa na eficiência, economizando consideravelmente o custo de manutenção. Uma vez que a MC é geralmente mais dispendiosa, em oposição à MPd (ZHU et al., 2019). As infraestruturas de Tecnologia da Informação (TI) que suportam os elementos do setor 4.0; como sensores, ferramentas de análise de big data, Internet das Coisas, do inglês *Internet of Things* (IoT) e sistemas cibernéticos; são necessárias para a coleta, processamento, análise e distribuição de grande quantidade de dados para fins de manutenção, esses são fatores crítico para a implantação de ferramentas de Inteligência Artificial para a manutenção preditiva na indústria (LIU et al., 2018).

Uma vez mencionado os méritos do sistema de inteligência artificial, é necessário preencher a lacuna entre o academicismo e o funcionamento do sistema de inteligência

artificial nas fábricas. Pois, é essencial estreitar a sinergia entre a indústria e o ambiente acadêmico, para juntos alavancarem a transformação da Indústria 4.0.

As fábricas de aprendizado ou fábricas pilotos permitirá que os futuros engenheiros utilizem efetivamente as tecnologias emergentes para melhorar a qualidade dos resultados, logo, liberando novos recursos. É necessário enfrentar os desafios da escassez de habilidades e conhecimentos no setor da Inteligência Artificial para MPd e familiarizar os fabricantes com as tecnologias emergentes por meio de fábricas de aprendizado. A implementação das tecnologias emergentes nas fábricas de aprendizado aumentará a facilidade de conhecimento dos futuros engenheiros, promovendo total integração nas atividades de fabricação e manutenção (DANIYAN et al., 2020).

2.5 Supercomputação

A computação de alto desempenho, do inglês *High Performance Computing* (HPC) é um instrumento fundamental para o desenvolvimento de novas tecnologias da Indústria 4.0. A HPC abriu inúmeras portas de descobertas científicas, incluindo física da matéria condensada, dinâmica molecular e Inteligência Artificial. Portanto, a demanda por um supercomputador de alto desempenho continua aumentando e, consequentemente, o número de núcleos e nós de supercomputadores vêm crescendo muito a medida do tempo, entretanto, essa tendência pode causar novos problemas em termos de confiabilidade (CINI; YALCIN, 2020).

O Centro de Supercomputação para Inovação Industrial (CS2I) do SENAI CIMATEC disponibilizou supercomputadores para esta tese para o desenvolvimento dos modelos de aprendizagem de máquina com grandes volumes de dados.

O alto paralelismo, fornecido pela arquitetura da Unidade de Processamento Gráfico, do inglês *Graphics Processing Unit* (GPU), proporciona maior desempenho e granularidade mais fina em comparação aos HPCs baseados em unidade central de processamento, do inglês *Central Processing Unit* (CPU). Além disso, as GPUs oferecem melhor desempenho por *watt* em comparação com as CPUs, sendo que um dos maiores gargalos de design de HPC se concentra no alto consumo de energia (NIE et al., 2016).

Portanto, espera-se que no futuro os HPCs sejam implementados em grande parte de nós de GPU (MARTINO et al., 2014).

2.6 Estado da Arte

A presente tese apresenta uma revisão do estado da arte do tema Aprendizagem Profunda aplicada para MPd. A revisão considerou os últimos dez anos de trabalhos acadêmicos publicados na literatura.

Zhang et al., (2018) treinaram uma rede Long Short-Term Memory (LSTM)para melhor descrever a condição de integridade do sistema e rastrear a degradação histórica, para prever com precisão a condição futura de saúde do equipamento. Neste artigo, é introduzido a abordagem baseada na rede LSTM, uma arquitetura especializada em descobrir os padrões subjacentes embutidos em séries temporais é proposta para rastrear a degradação do sistema e, consequentemente, prever a VUR.

Kolokas et al. (2018) propuseram uma LSTM, RNA de tipo recorrente, para prever a condição atual de um equipamento industrial usando o mecanismo de processamento de dados em larga escala Spark. Nesse caso, os dados consistiram em três configurações operacionais e 21 medições do sensor de temperatura, pressão e combustível do motor e líquido de refrigeração. Entretanto o modelo proposto foi para realizar uma classificação binária de falha e não falha, não sendo possível ser um modelo para diagnosticar falhas.

Huuhtanen e Jung (2018) propuseram uma CNN de aprendizagem profunda para monitorar a operação de painéis fotovoltaicos e prever a curva de energia elétrica diária de um painel fotovoltaico, com base nas curvas de energia dos painéis vizinhos. O resultado do modelo foi interessante com Erro Quadrático Médio de 0,002 para dados reais.

Chen et al., (2019) desenvolveram um modelo de aprendizagem profundo que permitiu conectar à camada convolucional em camada de memória de rede para prever a VUR de rolamento contínuo. Essa arquitetura proposta apresentou ser uma opção atraente para o estudo desta tese, pois os experimentos foram conduzidos para provar que o método proposto requer menos amostras de treinamento e supera outros métodos de Aprendizagem Profunda. Em comparação com as redes tradicionais, a rede híbrida pode adicionalmente extrair as informações globais e locais no eixo vertical de recursos e as informações de contexto associadas no eixo temporal horizontal.

Marins, et al., (2020) treinaram o algoritmo de aprendizagem de máquina Floresta Aleatória, com a base 3W, para classificar sete eventos defeituosos durante a

operação prática de poços e linhas de petróleo e gás. A classificação é realizada durante a fase transitória do evento e com objetivo de ajudar os operadores a identificar qual das sete classes de eventos indesejados está ocorrendo. Os dados foram transformados em medidas estatísticas para o algoritmo identificar cada tipo de falha. Três experimentos foram elaborados para avaliar o desempenho do sistema em diferentes cenários de classificação. O modelo revelou uma taxa de precisão de 94%, indicando um desempenho bem-sucedido para o sistema proposto na detecção de eventos reais.

Turan, et al., (2021) fizeram um experimento similar a Marins, et al., (2020) utilizando a base de dados 3W. Os algoritmos Regressão logística, Classificador de vetores de suporte, do inglês *Suport Vector Machine* (SVC), Análise Discriminante Linear e Quadrática, Árvores de decisão, Floresta Aleatória e AdaBoostGeneric foram treinados e comparados para a classificação automática de eventos indesejáveis durante a operação de poços de petróleo. Os recursos são calculados com base nos dados do sensor dentro de uma janela de tempo. Dos classificadores examinados, a Árvore de Decisão revelou o melhor desempenho: a maioria dos eventos é classificada com mais de 90% de precisão, com um F1-score de 85%.

Senjoba, et al., (2021) propuseram um método para detectar falhas em brocas de percussão rotativas usando aprendizado profundo. Um modelo CNN com aceleração do tempo como dados de entrada. 18 m^3 de rocha de granito foram perfurados horizontalmente usando uma perfuratriz e brocas de carboneto de tungstênio intactas. A aceleração do tempo das vibrações da broca foi medida usando sensores de aceleração montados na célula guia da perfuratriz. O modelo de detecção de falha da broca foi avaliado em cinco condições de perfuração: normal, defeituosa, abrasão, alta pressão e desvio de direção. O modelo alcançou uma precisão de classificação de 88,7%. O modelo proposto foi comparado a três redes neurais de aprendizado profundo.

Tran, Pham e Lundgren (2022) apresentaram um método para detecção de falhas em máquinas de perfuração usando sons de perfuração na Valmet AB. O conjunto de dados de perfuração inclui duas classes: sons anômalos e sons normais. Métodos de aumento de som foram aplicados para aumentar o número de sons no conjunto de dados. Uma CNN-LSTM foi desenvolvida para extrair recursos de espectrogramas log-Mel para aprender representações globais das duas classes. Uma unidade linear retificada com vazamento (Leaky ReLU) foi utilizada como função de ativação para a CNN

proposta. Como resultado, o método proposto alcançou uma precisão geral de 92,62% para classificar duas classes de sons de máquina no conjunto de dados da Valmet. Além disso, um extenso experimento em outro conjunto de dados de perfuração com sons curtos rendeu 97,47% de precisão. Com várias classes e sons de longa duração, um experimento utilizando o conjunto de dados UrbanSound8K disponível publicamente obteve 91,45%.

Alsaif e Alothman (2022) desenvolveram um algoritmo de Inteligência Artificial para analisar ondas sonoras de equipamentos de perfuração para avaliar sua condição de saúde e prever falhas de equipamentos, detectando padrões de som anômalos nos domínios de tempo e frequência e aplicar manutenção preditiva. Os dados sonoros foram coletados via microfone e transformados para o domínio da frequência através da Transformada de Fourier de Curto. Ambos os sinais no domínio do tempo e da frequência foram combinados para formar um espectrograma bidimensional. Esse espectrograma foi treinado usando um modelo de Inteligência Artificial para aprender a diferença entre sons de equipamentos normais e anormais. Como resultado, quando comparado ao sistema auditivo humano, os resultados experimentais mostraram que o método proposto obteve melhora significativa na detecção de sons anômalos apenas para máquinas que podem produzir ruído na região do ultrassom, enquanto outras máquinas obtiveram resultados piores em relação à região do som audível.

Jian, et al., (2023) desenvolveram uma rede neural semi-supervisionada para diagnósticar falhas de rolamentos a partir de uma grande quantidade de dados não rotulados. O método teve como motivação superar a fragilidade dos métodos tradicionais de autoaprendizagem, os quais possuem pseudo-rótulos ruidosos devido o treinamento realizado com poucos dados rotulados. A rede proposta poderia autocorrigir os pseudo-rótulos gerados, substituindo gradualmente por novos, por meio de uma arquitetura Aluno-Professor. A arquitetura Aluno-Professor proposta estima uma perda de aprimoramento de consistência para fortalecer a consistência de recursos entre as redes de alunos e professores.

Por meio dos trabalhos relacionados apresentados, é possível perceber que existe uma lacuna de modelos com técnicas semi-supervisionadas de autoaprendizagem para classificar múltiplas falhas em séries temporais multivariadas. Embora os métodos acima tenham revelado excelentes resultados, todos eles precisam de dados anotados

para treinar o algoritmo de aprendizagem profunda. Como principal lacuna tecnológica, identificamos que não existe uma metodologia proposta para desenvolver um modelo de autoaprendizagem profundo sem dados anotados para classificação de falhas em séries temporais multivariadas.

3. Manuscritos

Este capítulo apresenta os manuscritos que foram elaborados pelo autor, coorientador e orientador como parte da construção desta tese. Os manuscritos foram publicados em congressos e revistas de alto impacto científico.

3.1 Aprendizagem de Máquina Não Supervisionada

Este subcapítulo apresenta os principais manuscritos que investigaram a aprendizagem de máquina não supervisionada aplicada a séries temporais multivariadas. Os estudos consistiram em levantar os principais algoritmos para detecção de anomalias ou reconhecimento de padrões de interesse e avaliá-los. O objetivo foi averiguar se há um algoritmo que se destaque e qual é o nível de desempenho para uma variedade de problemas de naturezas distintas.

3.1.1 Manuscrito 1 - Multivariate Real Time Series Data Using Six Unsupervised Machine Learning Algorithms

Este manuscrito, publicado em um capítulo de livro, iniciou o estudo de investigação da aprendizagem de máquina não supervisionada para reconhecimento de padrões de interesse em dados de séries temporais multivariadas. A fundamentação teórica introduziu os principais modelos de aprendizagem de máquina não supervisionada para distintas finalidades. Seis algoritmos foram identificados e implementados para a experimentação. O experimento consistiu em comparar o desempenho dos algoritmos para reconhecer temporadas de furacões e falhas em dados de meteorologia e monitoramento de máquinas rotativas, respectivamente. Os algoritmos demonstraram que podem identificar padrões para auxiliar o processo de anotação de dados para, posteriormente, alavancar o treinamento da aprendizagem de máquina profunda supervisionada.

Capítulo de livro publicado no livro Anomaly Detection - Recent Advances, Issues and Challenges da editora IntechOpen (ISBN 9781839681).
DOI: http://dx.doi.org/10.5772/intechopen.94944

DOI: http://dx.doi.org/10.5772/intechopen.94944

Chapter

Multivariate Real Time Series Data Using Six Unsupervised Machine Learning Algorithms

Ilan Figueirêdo, Lílian Lefol Nani Guarieiro and Erick Giovani Sperandio Nascimento

Abstract

The development of artificial intelligence (AI) algorithms for classification purpose of undesirable events has gained notoriety in the industrial world. Nevertheless, for AI algorithm training is necessary to have labeled data to identify the normal and anomalous operating conditions of the system. However, labeled data is scarce or nonexistent, as it requires a herculean effort to the specialists of labeling them. Thus, this chapter provides a comparison performance of six unsupervised Machine Learning (ML) algorithms to pattern recognition in multivariate time series data. The algorithms can identify patterns to assist in semiautomatic way the data annotating process for, subsequentially, leverage the training of AI supervised models. To verify the performance of the unsupervised ML algorithms to detect interest/anomaly pattern in real time series data, six algorithms were applied in following two identical cases (i) meteorological data from a hurricane season and (ii) monitoring data from dynamic machinery for predictive maintenance purposes. The performance evaluation was investigated with seven threshold indicators: accuracy, precision, recall, specificity, F1-Score, AUC-ROC and AUC-PRC. The results suggest that algorithms with multivariate approach can be successfully applied in the detection of anomalies in multivariate time series data.

Keywords: unsupervised learning, pattern recognition, multivariate time series, machine learning, anomaly detection

1. Introduction

Today, the industry is changing by what experts call the "Fourth Industrial Revolution", also called Industry 4.0. This change is strongly associated with the integration between physical and digital systems through, for example, installing sensors. The integration of these environments allows the collection of a large amount of acquired data in different fields such as: industrial processes, meteorological monitoring stations, stock exchanges etc. This amount of both collected and stored data enables faster and more directed information exchange [1].

In many fields, it is essential for the process to identify unusual patterns that can be generated by unpredictable or unwanted behavior. These behaviors may be due to some problem that may be occurring in the related process, for example, in an industrial environment companies can use machine monitoring data to identify

malfunction operating due its abnormal behaviors. This fact, when it is not detected in time, can generate false data and lead experts to misinterpret the operating condition of the machine. Another example would be a credit card operator who can monitor each user's transaction to look for unusual behavior that could point to fraudulent transactions. These unwanted and abnormal behaviors are often called interest patterns and can be extracted from data due a variety of reasons, all presenting a certain level of relevance to the analyst. It is important that this analysis takes into account any changes in the behavior of the parameter to identify opportunities to improve, prevent or correct any situation [2].

The detection of interest/anomaly patterns is usually carried out by specialists which comprises the dynamics of the system under analysis. However, it is often not feasible to analyze and label them due to the large volumes of data generated. Thus, there is a limitation regarding the ability of specialists to process a large amount of data, requiring many hours of work that, in general, are involved in other activities and do not have the time necessary for this relevant activity. Thereby, there is a great need to automate the process of identifying hidden interest patterns in time series data [3].

Unsupervised machine learning (ML) has been research hotspot in intelligence artificial (IA) field to extract useful features from unlabeled raw data. Instead of selecting features by a human operator, the unsupervised learning is quite intelligent and independent of specific knowledge of processing techniques and field expertise in a data-driven way. Thus, there is no escape from the requirement of labeled data to train classifiers at the phase of diagnosis problems, but it is hard to label a mass of collected data before the determination of interest patterns [4].

In an industrial environment, data collection is often carried out through multiple sensors due the possibility of a more robust representation of the phenomena involved, as example, the monitoring of industrial assets is carried out through both acquisition of vibration and temperature data of the machine. Another example is the monitoring of meteorological conditions, which generally collect data on wind speed, air humidity and ambient temperature. However, multivariate data presents a greater challenge for the application of Machine Learning (ML) algorithms, as they must be able to recognize patterns and predict behaviors in a greater amount of data and attributes to be correlated [5].

Anomaly detection algorithms seek for patterns in data that do not conform to an expected behavior. Anomaly detection is essential in industrial applications to optimize economic performance and minimize safety risks. The advent of system health monitoring methods was realized to preserve system functionality within harsh operational environments. Hence, to develop an anomaly detection model based on multi-sensor signals, three major challenges must be faced [6]:

i. online multi-sensor signals are often available in the form of complex, multivariate time series, as different sensors measure various aspects of data over fairly long periods of time, and since there is a large amount of heterogeneous data, it can become impractical to human specialists to label anomalies or unknown events within the data;

ii. especially in industrial applications, it is likely to have extremely imbalanced datasets, since far more data is obtained during normal operations rather than abnormal;

iii. the dataset may contain uncertainties and spurious data, especially when it involves manually recorded data.

One way to mitigate these problems is to perform some type of anomaly detection technique, which are usually computationally intensive algorithms, and then flag unusual patterns for further inspection by human specialists [3]. Other way of dealing with it are based on supervised machine learning anomaly detection algorithms, which require a training dataset that contains a set of instances of anomalies, and a set of instances of non-anomalous (or normal) data, at least. From the training data, the algorithm learns a model that distinguishes between the normal and the anomalous patterns. Such supervised learning algorithms typically require tens or hundreds of thousands of labeled samples to obtain good quality performance. Nevertheless, as stated before, the scarce availability of labeled data poses a challenge to the usage and application of supervised learning techniques for anomaly detection in multivariate time series data.

Hence, unsupervised learning algorithms step up as a viable and feasible alternative to tackle this challenging problem. Since they are designed to deal with unlabeled data, they are able to learn and identify interesting patterns from the data's own internal structure, meaning that they can be used to point out anomalous patterns when the labels are unknown. Thus, unsupervised machine learning models are essential to solve the addressed challenges [7].

Several works proposed the development and application of unsupervised ML algorithms over the past years to detect anomalous patterns in time series datasets, which are based on major approaches that are summarized in **Table 1**. A detailed description and evaluation of each of these approaches is beyond the scope of this chapter.

Most of the works based on unsupervised ML algorithms employs clustering techniques, which are either distance-based [10] or density-based [11]. On the one hand, the advantages of clustering methods are that they are simple, robust, and easy to program. However, the problem is the need to define the parameters related to the data observations beforehand such as defining a similarity function or the number of clusters that should exist in the data, and that becomes the responsibility of the designer to determine how these parameters should be used, even if the data has a random structure [9]. The work [12] proposed the K-means to automate diagnosis of defective rolling bearing. To overcome the sensitivity of choosing the initial clusters number, the initial centers were selected using features extracted from simulated signals. However, K-means depends mainly on distance calculation between all data points and the centers, therefore, the cost of the computational time will be higher for big data.

To reduce the time cost of K-means, [13] proposed a Fast K-means algorithm based on two stage. The first stage is a fast distance calculation using only a small fraction of the data to originate the best possible location of the centers. The second stage is a slow distance calculation in which the initial centers are taken from the first stage. Besides that, the K-means is optimized through grid search method that is efficient when the number of parameters is small.

Aiming to prove the reduction quality of the dataset, [14] demonstrated the superiority of the autoencoder in the feature dimensional reduction comparing with the PCA method. The reduced feature set obtained from the PCA method revealed overlaps of classes and features that are scattered on the large space, while the autoencoder represented the superior ability in the clear and concentrated distribution.

The work [15] presented a similar comparative study to evaluate the performance of the autoencoder to the original feature set, PCA reduction and real-valued negative selection. The dimensional reduction of the autoencoder, once again, have performed a highly improved anomaly detection compared to the others. Moreover, PCA space transformation requires complete knowledge about normal and faulty data classes.

Algorithm	Description
K-means	It is used to divide a group of data points into hard clusters. It assumes a balanced cluster size, the joint distribution of features with equal variance, and independent features with similar cluster density. Determining the optimal K can be difficult, but for small values, it is computationally fast and efficient. In addition, it is important to choose the most appropriated distance or similarity function, since it is one of the key aspects used to determine whose cluster a certain data point belongs to.
Gaussian mixture model	It uses a Gaussian distribution-based parametric model to identify the underlying populations. These can be explained by a normal distribution in the midst of many heterogeneous populations. However, in many practical situations, the data distribution may not have any explicit clusters. As a result, each point can be assigned with different weights or probabilities to soft clusters.
Random forest	It operates by constructing a multitude of decision trees at training time and outputting the class that is either the mode of the classes (classification) or the mean prediction (regression) of the individual trees. Random forest can learn arbitrary relationships between the features and the outcome, even non-monotonic relationships.
k-NN	It assigns data points according to the majority of its nearest neighbors to find anomalous data points by measuring the local deviation. A choice needs to be made on the value of *K*, i.e. the number of neighbors, to avoid overfitting/underfitting issues.
DBSCAN	It recognizes the clusters as dense regions having some coincidence that is diverse from the other sparse region. The algorithm may use a reduced number of points and a distance measure to merge the data points that are similar to each other. Moreover, DBSCAN requires two parameters to operate, which are the epsilon (*eps*) and the minimum points (*minpoints*). *Eps* determines the smallest distance existing between two points in a cluster, while *minpoints* defines the least number of points required to form a dense region [8].
PCA	Principal Component Analysis (PCA) based anomaly detection techniques are able extract the main features of a certain dataset without losing its ability to represent the original data, then using these features to analyze which constitute a normal class and applies distance metrics to identify cases that represent anomalies. This allows to train a model using existing imbalanced data.
Autoencoders	It is a neural network that attempts to reconstruct its input. Similarly to PCA, it can serve as a form of extract main features to produce a compressed representation of its input at the encoder. This representation can be mapped to its original form using a decoder. Anomaly detection uses the reconstruction error to measure how well the decoder is performing.

Adapted [9].

Table 1.
Unsupervised ML approaches found in literature for anomaly detection in time-series.

Two methods for feature selection were proposed by [15]. The first one is based on k-NN for clustering using feature similarity influence, and the second one is the pretraining using sparse autoencoders. The classification performance obtained by the k-NN algorithm is comparable to the result obtained from the autoencoder (slightly lower in accuracy). The criterion for choosing the "k" parameter is based on the combinations that frequently appear in the subsets reduced by the technique itself. In other words, the criterion adopted is purely empirical.

Furthermore, even though several unsupervised techniques have been proposed in literature, their performance depends a lot on the data and application they are being used in. This indicates that most of these methods have little systematic advantages over the other when compared across many other datasets.

In this context, this chapter discuss the level of accuracy and reliability of six unsupervised ML algorithms for pattern recognition and anomaly detection with no need of labeled data. Two real cases were applied for performance evaluation of the algorithms abilities to detect the interest patterns in the multivariate time series data.

The real cases were: (i) meteorological data from a hurricane season and (ii) monitoring data from dynamic machinery for predictive maintenance purposes.

2. Unsupervised ML algorithms

This section will review the concept and application of six unsupervised ML algorithms for anomaly/pattern detection applied in this research. It is important to inform that among the six algorithms described in this chapter, only the methods in Sections 2.1 and 2.3 has its intrinsic characteristic to perform a multivariate analysis, while the other algorithms are only able to perform a univariate analysis. Therefore, a univariate performance evaluation is computed in each dimension of the data to then calculate an average performance, except algorithms in 2.1 and 2.3 sections.

2.1 C-AMDATS

The Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance (C-AMDATS) is a clustering ML unsupervised algorithm. The model has only two hyperparameters that user can manipulate: (i) Initial Cluster Size (ICS) and Clustering Factor (CF). First the ICS clusters the observed sequences of time series data A, where each cluster may represent a behavior status. After the initial clustering, a new and better clustering in the dataset is remade according to the data points distribution over timeline. This ability is due to the usage of the Mahalanobis distance in the algorithm. In general, clustering techniques use the Euclidean distance function, which makes the clustering assumes the geometric shape of a circle, then it does not consider the variance of each dimension or feature of the dataset. However, there are situations in which the variance between each dimension (or feature) is different. Conversely, by using the covariance matrix, the Mahalanobis distance can detect the variance of each dimension. Eq. (1) presents the Mahalanobis distance formula.

$$d_m(x,\mu) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)} \tag{1}$$

Where: $d_m(x,\mu)$ is the Mahalanobis distance between a specific point in the time series and its respective centroid; $x = (x_1, x_2, \ldots, x_n)^T$ is a specific variable in the time series data, where n is the number of variables; $\mu = (\mu_1, \mu_2, \ldots, \mu_n)^T$ is a certain cluster centroid; and S is the covariance matrix relative to that cluster.

After the new clustering through the Mahalanobis distance, the algorithm calculates the similarity of each cluster in the time series A to find the respective hidden patterns P. This similarity is calculated using the standard deviation σ_y of the actual values of the A samples, the Y coordinate of each centroid and the CF. If the modulus of the difference between the y coordinate of the centroids of two cluster is less than or equal to the product of CF and σ_y, then these clusters can be merged, meaning that they will represent the same pattern P. This task is carried out until every cluster have been analyzed.

The last step of C-AMDATS is to calculate the probability of the pattern P to be an anomaly R. The Anomaly Score measures the anomaly R for each pattern P (found in the previous step). The score is calculated by the ratio of the size of the entire time series to the sum of the sizes of the clusters present in P. The anomaly score assesses the degree of relevance of P in terms of anomaly detection. Then, all set P is ordered by R in descending order, and the anomalous patterns will be those with the highest anomaly score values. The higher the anomaly score value for a pattern P, the greater probability is of being an anomaly behavior in A [2]. Eq. (2) presents the Anomaly Score formula.

$$Anomaly\ Score_{P_i} = \frac{|T|}{|P_i|} \tag{2}$$

Where $Anomaly\ Score_{P_i}$ is the anomaly score of the pattern P_i, $|P_i|$ is the size of the pattern P_i, and $|T|$ is the size of the time series T.

2.2 Luminol Bitmap

Bitmap is an available unsupervised learning algorithm in Luminol library for anomaly detection or time series correlation. The background of Bitmap algorithm is based on the idea of time series bitmaps. The logic of the algorithm is to make a feature extraction of the raw time series data - by converting them into a Symbolic Aggregate Approximation (SAX) representation - and use it to compute the information about the relative frequency of its features to color a bitmap in a principled way. SAX allows a dimensionality reduction of the raw time series C of arbitrary length n to a string arbitrary length w ($w < n$, typically $w << n$) by a vector $\overline{C}$. It transforms the data into a Piecewise Aggregate Approximations (PAA) representation to symbolize it into a discrete string [16]. Eq. (3) presents the calculation of the *ith* element of $\overline{C}$:

$$\overline{C}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} C_j \tag{3}$$

After transformed a time series dataset into PAA, the algorithm applies a further transformation to obtain a discrete representation with equiprobability [16]. The conversion of the time series into a SAX words is made by a slider window (also called feature window). Bitmap algorithm use two concatenated slider windows together across the sequence, the latter one is called lead window, showing how far to look ahead for anomalous patterns and the former one is called lag window, whose size represents how much memory of the past to remember it.

In summary, the algorithm approach is to convert both feature windows into SAX representation, then count the frequencies of SAX subwords at the desired level and get the corresponding bitmaps. The distance between the two bitmaps is measured and reported as an anomaly score at each time instance, and the bitmaps are drawn to visualize the similarities and differences between the two windows. The user must choose the length of the feature windows N and the number n of equal sized sections in which to divide N [3].

2.3 SAX-REPEAT

SAX-REPEAT algorithm is an approach that relies on extending the original SAX implementation to handle multivariable data. The algorithm takes as input a set of K multivariable time series X_i of lengths T_i, and dimensionality D, that represent different instances of the raw data to be learned. The user can set the parameters of the final string length N and an alphabet size M.

The algorithm applies SAX to each dimension of the data separately, and then combine the output string by assigning each possible combination of symbols, resulting in D strings to a unique identifier. This leads to a string of length N, but an extended alphabet of length M^D. So, to maintain the requirement of the final

string to be an alphabet of symbols M (parameter set by the user), the algorithm clusters the resulting characters into M clusters through K-means method and replace each character with the centroid of its cluster [17].

Although SAX-REPEAT can recognize interesting patterns, the original algorithm does not calculate the probability of the patterns being anomalous. Thereby, this work implemented an anomaly score for each found cluster (pattern). As C-AMDATS algorithm, the score is computed by the ratio between the size of the entire time series and the sum of the sizes of each cluster. Therefore, each cluster is sorted according to the respective anomaly score in descending order, and the anomalous patterns will be those with the highest anomaly score values.

2.4 k-NN

The k-Nearest Neighbors (k-NN) algorithm is one of the most popular method to solve both classification and regression problems. However, in this study, we will use it only for classification problem as unsupervised learning.

The algorithm assumes that similar data points exist in proximity, *i.e*, they are near to each other. The algorithms capture the idea of the similarity (also known as distance, proximity, or closeness), calculating the distance between points on a graph. Distance calculation is usually done by Euclidean distance, but it can be calculated using other distance functions. The Euclidean distance between the points $P = (p_1, p_2,\ldots,p_n)$ e $Q = (q_1, q_2,\ldots,q_n)$, in a n-dimensional Euclidean space, it is defined in Eq. (4):

$$d_e(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{4}$$

Where, $p = (p_1, p_2, \ldots, p_n)$ and $q = (q_1, q_2, \ldots, q_n)$ are two points in Euclidean n-space.

The k-NN algorithm depends on two parameters, a metric used to compute the distance between two points (in this case Euclidean function), and a value k of the number of neighbors to consider. When k is underestimated, the algorithm can overfit, *i.e*. it will classify just based on the closest neighbors instead of learning separating frontier between classes, but if k is overestimated, the algorithm will underfit, in the limit if $k = n$, the algorithm will consider every point belongs to the class that has more samples [18, 19].

2.5 Bootstrap

Bootstrap algorithm uses the computational power to estimate almost any summary statistics, such as the confidence interval, mean, or standard error. The method depends on the notion of a bootstrap sample B, which is a resampling of size n drawn to replace the original dataset $Z = (Z_1, Z_2, \ldots, Z_n)$. The bootstrap sample is represented $Z^* = \left(Z_1^*, Z_2^*, \ldots, Z_n^*\right)$. Each Z_i^* is one of the original Z values randomly selected, the selection probability for each Z value is equipollent, for example: $Z^* = Z_7, Z_2^* = Z_5, Z_3^* = Z_9, Z_4^* = Z_7$, etc. Note that the same original value can appear zero, one or more times, in the example, Z_7 appeared twice, *i.e*, the selection of Z value is not exclusive. The name Bootstrap concern to the use of the original dataset to generate new datasets Z^*. The idea is to generate a larger number of Bootstrap sample B of each size n using a random number device to perform the algorithm training. The number of bootstrap repeats defines the variance of the estimate, *i.e*, higher the number is, better is the variance, but in contrast, the computational cost increases with the increasement of the B number [20, 21].

In this sense, we are interested in calculating a confidence interval using Bootstrap, which is performed by requesting the statistics stored during the training and selecting values in the chosen percentile for the confidence interval. The chosen percentile is denoted as δ (Alpha or Significance Level). Eq. (5) defined the calculation to estimate the distribution of δ^* for each Bootstrap sample.

$$\delta^* = \overline{x}^* - \overline{x} \tag{5}$$

Where: $\overline{x}^*$ is the mean of an empirical bootstrap sample and $\overline{x}$ is the mean of the original data.

Therefore, the confidence interval for a Significance Level of 0.05 is defined by Eq. (6).

$$\text{Confidence interval} = \left[\overline{x} - \delta^*_{.05}, \overline{x} - \delta^*_{.95}\right] \tag{6}$$

Where, $\overline{x}$ is the mean of the original data, $\delta^*_{.05}$ is significance level at the 5th percentile, and $\delta^*_{.95}$ is significance level at the 95th percentile.

So, in order to obtain a very accurate estimate of $\delta^*_{.05}$ and $\delta^*_{.95}$, it is important to generate a large number of bootstrap samples.

2.6 RRCF

Robust Random Cut Forest (RRCF) algorithm is an ensemble technique for detecting outliers. The idea is based on an isolation forest algorithm that uses an ensemble of trees. In graph theory, trees are collections of vertices and edges where any two vertices are only connected by one edge, it is an ordered way of storing numerical data.

In this view, the algorithm takes a set of random data points, cuts them to the same number of points and creates trees. The algorithm starts by constructing a tree of n vertices, then it creates more trees of the same size, which in turn creates the forest. The user can choose the number of trees and the number of data point of each tree has, which is randomly sampled from the dataset. After the construction of the forest, the algorithm injects a new data point p into the trees to follow the cuts and to compute the average depth of the point across a collection of trees. The point is labeled an anomaly if the score overtake a threshold, which corresponds to the average depth across the trees [22].

3. Comparative analysis between the algorithms

This section will discuss the details of the datasets, the algorithm parameters settings, the evaluations performance method, and a comparative analysis between the algorithms.

In order to summarize the advantages and limitations of each algorithm, **Table 2** shows advantage and limitations of the ML algorithms.

The advantages and limitations of each algorithm will be discussed throughout the development of this chapter.

3.1 Characterization of case studies

This chapter brings two sets of real data, which were collected for the purpose of detecting anomalies in multivariate time series. The databases applied in this study will be detailed in the next sections.

ML algorithms	Advantage	Limitations
C-AMDATS	• Multivariate approach • Easy to parameterize • Considers the variances of each dimension	• Take more CPU time due to the need to compute the inverse of the covariance matrix for each cluster • Sensitive to noisy data, missing values and outliers
Luminol Bitmap	• Quick calculation time • Runs well on big data • Not too sensitive to parameter choices	• Univariate approach
SAX-REPEAT	• SAX Multivariate approach	• Hard to parameterize • Slow calculation time due the k-means clustering • Sensitive to missing values and outliers
k-NN	• Quick calculation time • Easy to implement • Variety of distance criteria can be chosen	• Does not work well with large dataset • Does not work well with high dimensions • Sensitive to noisy data, missing values and outliers • Univariate approach
Bootstrap	• Quick calculation time • Does not require large sample size	• Can be computationally expensive depending on the bootstrap sample number • Univariate approach
RRCF	• Different dimensions are treated independently • Designed to run in a streaming data	• Univariate approach • Can be computationally expensive depending on the number of trees • Sensitive to noisy data, missing values

Table 2.
Advantage and limitations of the unsupervised ML algorithms.

3.1.1 Case study 01 - meteocean data in hurricane season

The chosen set is a public meteocean data available online in the National Data Buoy Center of the National Oceanic and Atmospheric Administration's (NOAA). The dataset was collected in the Atlantic Ocean off the Bahamas coast (23,838 N; 68,333 W). The data were structured in hourly frequency and it begins in June 2012 until November 2012 (213 days and 22 hours), comprising 15,315 data points. This period corresponds to the hurricane season in the Ocean Atlantic, which that year was especially active with 19 tropical cyclones (winds above 52 km/h), which 10 cyclones became hurricanes (winds above 64 km/h).

Hurricanes can be detected by several meteorological variables that consequently are directly impacted. In this case study, the following analysis variables were considered: (a) significant wave height (WVHT); (b) sea level pressure (PRES); and (c) wind speed (WSPD). Within the period of the dataset, three hurricanes transited through the Bahamas coast region: (i) Isaac; (ii) Rafael and (iii) Sandy.

Isaac had his first alert issued on August 21 by the National Hurricane Center. Several islands in the Lesser Antilles have been placed under hurricane surveillance or tropical storm warnings. Isaac was tracked between Guadalupe and Dominica on August 22, it passed over Haiti and Cuba with a strong tropical storm force. On

August 26, the Isaac approaches Florida Keys and the next day entered the eastern Gulf of Mexico causing several economic impacts in the USA. There was a gradual intensification and Isaac reached its peak intensity as a category 1 hurricane, with sustained 1-minute winds of 80 mph (130 km/h) [23].

Hurricane Rafael produced minor damage in the northeastern Caribbean Sea in mid-October 2012. The first alert was issued to Bermuda on October 14, but was canceled on October 17 when the hurricane passed northeast of the island. On October 16, Rafael reached his peak intensity with maximum sustained winds of 90 mph (150 km/h). Rafael intensified in a category 1 hurricane [24].

Hurricane Sandy was the deadliest and most destructive, as well as the strongest, hurricane of the 2012 Atlantic hurricane season. Inflicting nearly $70 billion USD in damage, Sandy was a Category 3 storm at its peak intensity when it made landfall in Cuba. On October 24, Sandy became a hurricane reaching the coast near Kingston and Jamaica. On October 25, it hit his peak intensity in Cuba. On October 31, Sandy was already off the coast of Maine in the United States of America. [25].

The **Figure 1** illustrates the period of hurricanes Isaac, Rafael and Sandy in the multivariate time series data.

In **Figure 1**, it is possible to visualize a behavior similarity between the three hurricanes. During the passage of the hurricanes, the variables WVHT and WSPD presented upward spikes, but on the other hands, PRESS presented downward spikes.

It is worth noting that the period of hurricanes in the **Figure 1** represents the time of its trajectory on the coast of Bahamas, and not its life span throughout its trajectory in the Atlantic Ocean.

3.1.2 Case study 02: monitoring data from dynamic machinery

The public dataset provided by the KNIME was acquired from 28 sensors installed in a dynamic machine. The sensors were installed to collect eight mechanical components parts (1st column of **Table 3**). The data starts on January 1st of 2007 and goes until April 20th of 2009 (838 days), comprising 16,660 data points.

The dataset was composed of 28 time series from 28 sensors. The signals were pre-processed with Fast Fourier Transform (FFT). The **Table 3** shows the groups and description of the sensors.

Each sensor group had at least 3 collections with different frequency bands, except the torque variable (M1), which had only one collection.

Signs of rotor malfunction could be traced back to March 6, 2008. The breakdown event happened on July 21, 2008. The break was visible only to some sensors, especially with low frequency bands.

For a cleaner and clearer view, **Figure 2** illustrates the multivariate time series only for sensors that detected the malfunction zone of the dynamic machine. Therefore, of total of 28 sensors, 18 were chosen to illustrate the multivariate time series. The machinery malfunction was detected in all sensor groups, except for the M1.

In **Figure 2**, it is possible to verify a behavior change in the multiple sensors inside the malfunction zone (begging of March until the end of July). The **Figure 2** also illustrates two alarms in the beginning of 2007 triggered by the KNIME system. However, these two alarms can be considered as pre-mature, as the history of the machinery continued to run normally over one year. Another detail is that, afterwards the breakdown and rotor replacement, the signals were recorded much cleaner.

3.2 Parameterization of the ML algorithms

The parameters of unsupervised ML algorithms were settings to achieve the best possible performance to find the patterns of interest. The parameterization requires

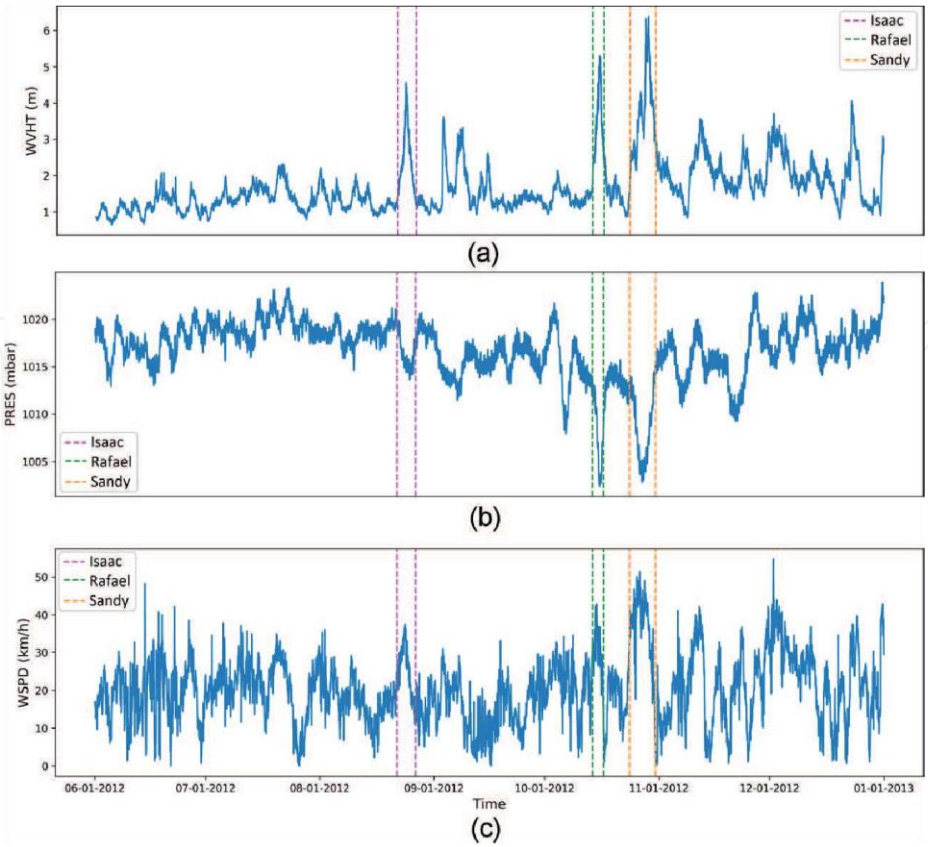


Figure 1.
Visualization of three variables in the same time domain (a) significant wave height, (b) sea level pressure and (c) wind speed. Color boxes represent hurricanes Isaac, Rafael and Sandy. Source: produced by the authors.

Sensor Group	Sensor Description
A1	Input shaft vertical
A2	Second shaft horizontal upper bearing
A3	Third shaft horizontal lower bearing
A4	Internal gear 275 degrees
A5	Internal gear 190.5 degree
A6	Input shaft bearing 150
A7	Input shaft bearing 151
M1	Torque kNm

Table 3.
The eight parts of the rotor monitored through groups of sensors.

several attempts of success and error to achieve the best possible result. SAX-REPEAT was the most difficult method of setting the parameters due the high sensitivity of the variables. Whereas the Luminol Bitmap revealed not too sensitive to parameter choices, where the Bitmap Detector Score demonstrated the most determinant parameter for the algorithm. The Bootstrap showed a similar result for iterations above 200 and confidence level above 95%. C-AMDATS, RRCF and k-NN are easy algorithms to set the parameter due the small number they have. As an example of experiment case 2, it was necessary to run SAX-REPEAT with 76 different combinations of parameters to identify the best configuration, k-NN was necessary to run 21 times, C-AMDATS 20 times, RRCF 11 times, Luminol 12 times, and Bootstrap 10 times.

The **Table 4** summarizes the parameter settings of the presented algorithms for the two real cases applied in this chapter.

All ML algorithms in this paper were implemented in Python 3.6 programming language and executed on a high performance computing named AIRIS (Artificial Intelligence RSB Integrates System) at the Supercomputing Center for Industrial Innovation at SENAI CIMATEC. The AIRIS processor model is the Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz and has 376 GB RAM memory.

3.3 Case study experiment 01 - meteocean data in hurricane season

All monitoring variables at the meteocean data were processed in ML algorithms using the settings presented in the **Table 4**. The results were compared to the period of hurricanes life as shows in **Figure 1**. The hurricanes behaviors are

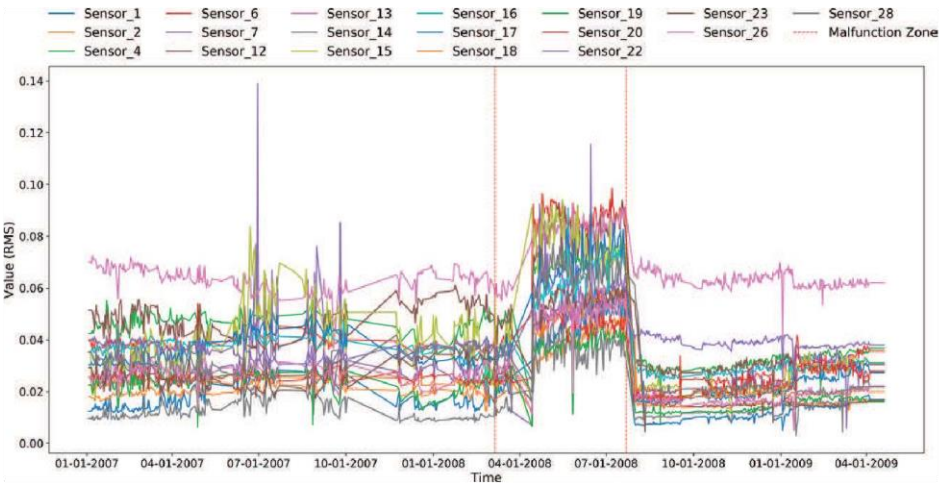


Figure 2.
Multivariate time series of 18 sensors that detected the malfunction zone of the machine.

Algorithm	Parameter Setting	
	Case 01	**Case 02**
C-AMDATS	CF = 3.0 ICS = 24	CF = 1.8 ICS = 30
Luminol Bitmap	Bitmap Detector Score = 1.0 Precision = 40 Lag Window Size = 10 Future Window Size = 10 Chuck Size = 24	Bitmap Detector Score = 0.35 Precision = 40 Lag Window Size = 10 Future Window Size = 10 Chuck Size = 30
SAX-REPEAT	Window size = 1155 PAA size = 3 Alphabet length = 2	Window size = 120 PAA size = 3 Alphabet length = 3
k-NN	k = 5 Metric = Euclidean Distance	k = 150 Metric = Euclidean Distance
Bootstrap	Confidence = 0.95 Number of Iterations = 200	Confidence = 0.95 Number of Iterations = 200
RRCF	Number of Trees = 10 Tree Size = 5105	Number of Trees = 20 Tree Size = 595

Table 4.
Parameter setting of the Unsupervised ML algorithms. CF: Cluster Factor - ICS: Initial Cluster Size - PAA: Piecewise Aggregate Approximations - k: Neighbors number.

more visually clear through the WVHT variable. Thereby, for the better understanding of the reader, we only illustrated the ML results in the WVHT variable, even though was made a multivariate analyzed. **Figure 3** shows the detection of the six algorithms.

In **Figure 3**, the C-AMDATS algorithm detected three distinct behavior patterns in the multivariate time series. Patterns 0 and 1 had the highest anomaly score and are well situated in hurricanes regions, so these patterns where considered as anomaly behavior. However, pattern 1 also appear in November and December, which had no records of hurricane or tropical depression or tropical storm in the Bahamas cost, revealing to be a false positive signal. The Luminol bitmap and RRCF algorithms failed to isolate the patterns of interest. Luminol demonstrated a little sensitivity for detecting anomalies in this experiment, because it was possible to verify a few data points detected as anomaly. SAX-REPEAT returned 6 distinct patterns, which patterns 4 and 5 were the top 2 of the anomaly score. These two patterns are precisely in the regions of interest, however, it is possible to verify these patterns also in other regions, also indicating false positive signals. Bootstrap and k-NN had similar results, both algorithms detected spikes caused by hurricanes, but with many false positives, especially Bootstrap.

Therefore, it is possible to ascertain that the algorithms with the best performance in detecting the patterns of interest in case 01 were C-AMDATS and SAX-REPEAT. But a quantitative analysis will still be performed.

3.4 Case study experiment 02 - monitoring data from dynamic machinery

Analogous to the experiment performed in case 1, the experiment case 2 brings the results of the patterns and anomalies detection of unsupervised learning algorithms in the KNIME dataset.

All 28-monitoring data were processed using the settings in **Table 4**. The results were compared to the period of malfunction of the machine as show in **Figure 2**.

The machine malfunction is more visually clear through the sensor 1. Therefore, **Figure 4** only illustrated the results of Sensor 1, although the analysis was performed in a multivariable way.

In **Figure 4**, the C-AMDATS algorithm detected three distinct behavior patterns in the multivariate time series. Patterns 0 had the highest anomaly score and is well situated in the interest region, so this pattern was assumed to be anomalous. Luminol and RRCf again failed to isolate the fault, both algorithms had many false positives and false negatives. SAX-REPEAT detected 15 different patterns, which is not desired as it makes difficult for the specialist to analyze many patterns. Nevertheless, patterns 0 and 4 had the lowest punctuation in the anomaly score ranking, so these patterns were assumed to be normal and the others as anomaly. The k-NN and Bootstrap methods also demonstrated a good performance in isolating the period of interest, with few false positives and false negatives.

Therefore, the algorithms were able to isolate the anomalous region well in case 02, with exception of Luminol Bitmap and RRCF.

3.5 Performance evaluation

The performance evaluation of the algorithms - in their ability to identify the same anomalous patterns - was performed through the calculation of seven metrics: accuracy (ACC), precision (PR), recall (REC), specificity (SP), F1-score (F1), area under the curve (AUC) of receiver operating characteristics (AUC-ROC), and AUC of precision and recall curve (AUC-PRC).

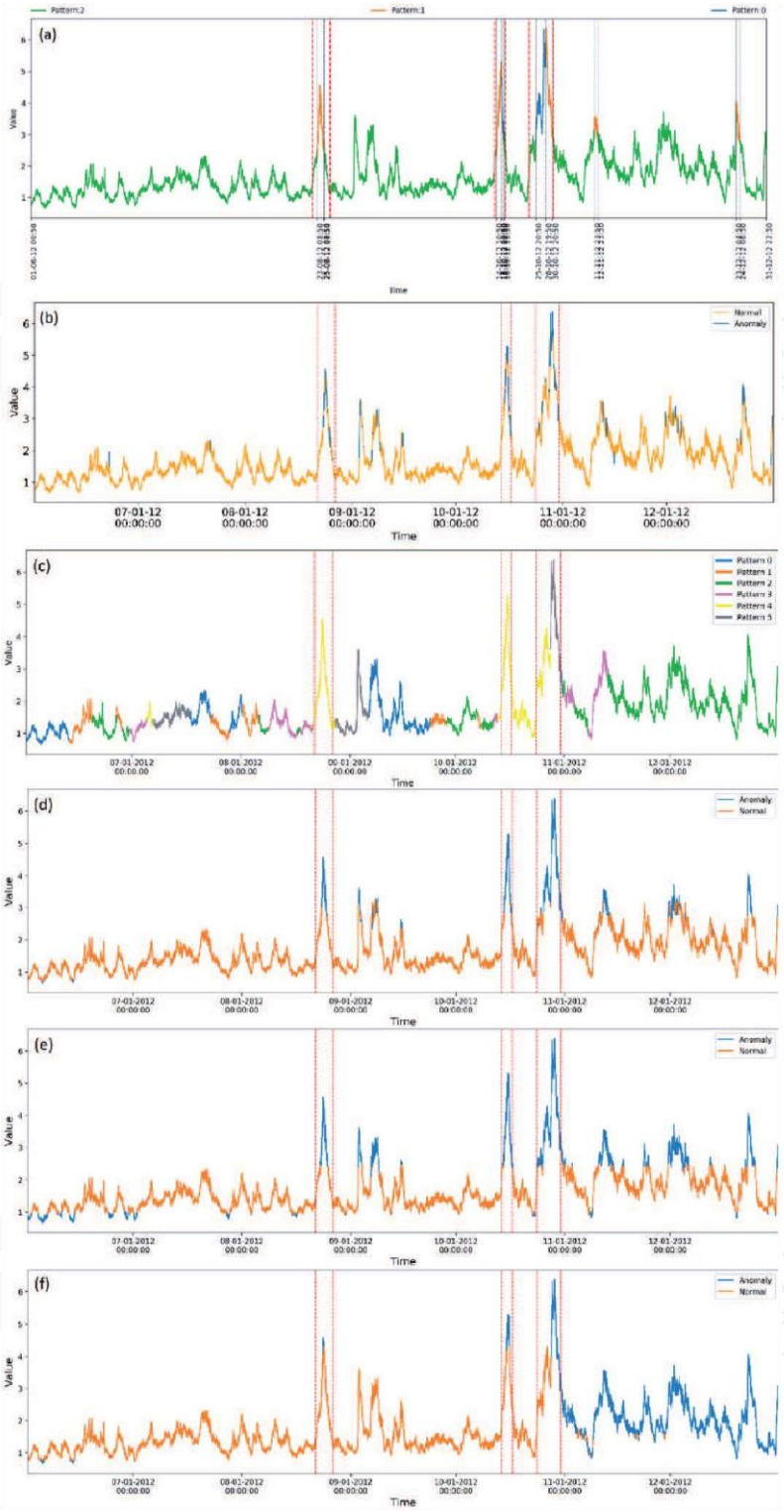


Figure 3.
Results of the unsupervised algorithms of Case 1. (a): C-AMDATS, (b): Luminol Bitmap, (c): SAX-REPEAT, (d): K-NN, (e): Bootstrap, and (f): RRCF.

However, the performance evaluation would not be properly fair, as the Luminol, k-NN, Bootstrap and RCCF algorithms made the analysis univariably (different from C-AMDATS and SAX-REPEAT). Thereby, in an attempt to obtain a more appropriate analysis, the threshold metrics was calculated for all proposed variables and then extracted an average evaluation, except C-AMDATS and SAX-REPEAT.

All the evaluation metrics are calculated by comparing the real data points (classified by experts) with the predicted data points (predicted by ML algorithms). So, the ACC reveals the correct prediction in a general approach, but it may hide the error rate of the model, that is why it is prudent to measure the performance jointly with other metrics. PR indicates the true positive value compared to the false negative. REC reveals out the true positive value with the false positive. Both

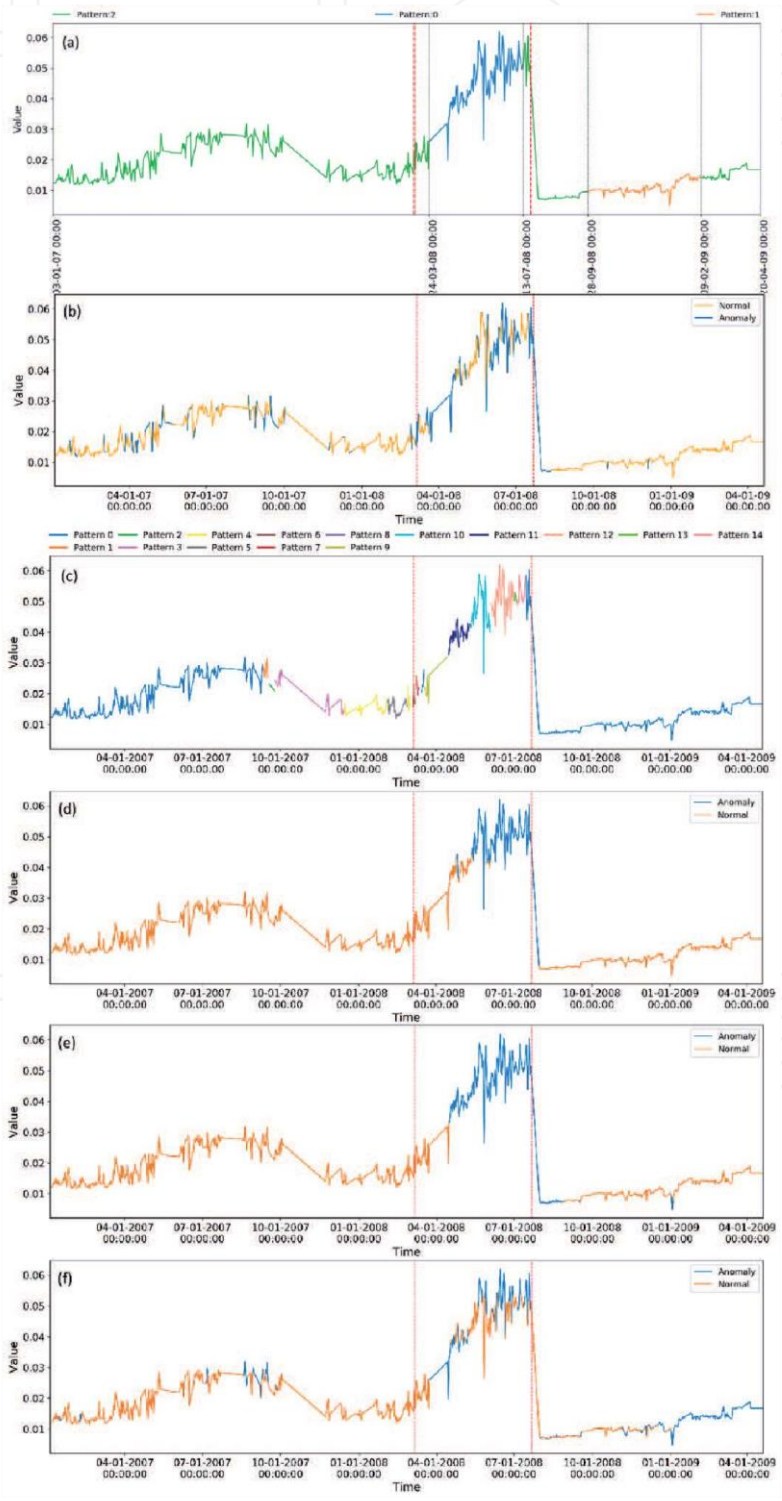


Figure 4.
Results of the unsupervised algorithms of Case 2. (A): C-AMDATS, (B): Luminol Bitmap, (C): SAX-REPEAT, (D): K-NN, (E) Bootstrap, and (F): RRCF.

ML Algorithms	Metrics	Case #1	Case #2	Average
C-AMDATS	ACC	96%	96%	96%
	PR	90%	98%	94%
	REC	80%	89%	85%
	SP	80%	89%	85%
	F1	84%	92%	88%
	AUC-ROC	81%	89%	85%
	AUC-PRC	76%	88%	82%
Luminol Bitmap	ACC	86%	72%	79%
	PR	55%	58%	57%
	REC	55%	61%	58%
	SP	55%	61%	58%
	F1	55%	58%	57%
	AUC-ROC	56%	61%	58%
	AUC-PRC	52%	56%	54%
SAX-REPEAT	ACC	86%	89%	88%
	PR	66%	82%	74%
	REC	92%	90%	91%
	SP	92%	90%	91%
	F1	70%	85%	78%
	AUC-ROC	92%	90%	91%
	AUC-PRC	66%	79%	73%
k-NN	ACC	94%	86%	90%
	PR	80%	81%	80%
	REC	67%	68%	67%
	SP	67%	68%	67%
	F1	69%	71%	70%
	AUC-ROC	66%	68%	67%
	AUC-PRC	61%	67%	64%
Bootstrap	ACC	79%	85%	82%
	PR	59%	72%	65%
	REC	73%	75%	74%
	SP	73%	75%	74%
	F1	59%	73%	66%
	AUC-ROC	73%	75%	74%
	AUC-PRC	56%	70%	63%
RRCF	ACC	80%	66%	73%
	PR	55%	44%	50%
	REC	55%	45%	50%
	SP	55%	45%	50%
	F1	54%	44%	49%
	AUC-ROC	55%	45%	50%
	AUC-PRC	53%	49%	51%

Table 5.
Performance Evaluation of unsupervised ML algorithms to detect interesting/anomalous patterns in multivariate time series data.

metrics (PR and REC) reveal the model's ability to predict positive values, but with different perspectives. SP demonstrated the capacity of the model to predict the true negative over false positives perspective. The F1 is a harmonic average between REC and PR. AUC-ROC is the area under the curve on the true positive (REC) and false positive (1- SP) rates. The AUC-PRC is the area below the curve between PR and REC. AUC-PRC is an important metric for assessing unbalanced datasets, being a great advantage over the others, since in the vast majority of cases, especially real data, have a higher volume of normal than abnormal data.

The seven performance assessment metrics for all proposed variables of case 1 and case 2 experiments are listed in the **Table 5**.

The performance evaluation presented in the **Table 5** revealed that the C-AMDATS was the one that stood out amongst the other algorithms. C-AMDATS was superior in ACC, PR, AUC-PRC and F1 metrics against SAX-REPEAT, which was the second algorithm that stood out. Nevertheless, it is relevant to note that C-AMDATS was 10% superiority in AUC-PRC of SAX-REPEAT. Then, in decreasing order of algorithm position in the performance evaluation would be: (i) C-AMDATS, (ii) SAX-REPEAT, (iii) k-NN, (iv) Bootstrap, (v) Luminol and (vi) RRCF. Both algorithms that have a multivariate analysis intrinsically were superior. However, more case studies must be carried out to affirm the superiority of the algorithms studied here.

Therefore, the results presented in this study strengthens the idea that unsupervised machine learning algorithms can assist the data annotation and labeling process. This approach can optimize much of the specialists' time and leverage the supervised AI models.

4. Conclusions and future work recommendations

This work demonstrated the effectiveness of a multivariate analysis using six different unsupervised ML algorithms for time series. To verify the performance of the unsupervised ML algorithms to detect interesting/anomalous patterns in real time series data, the six algorithms were applied in two different real cases: (i) meteocean data in hurricane season and (ii) monitoring data from dynamic industrial machinery. The experimental results showed that clustering methods as C-AMDATS have higher capacity to recognize and isolate the anomaly region, revealing the ability to assist experts to label raw data with unsupervised ML algorithms with great performance.

Future works include the extension of this analysis to more real cases aiming to develop a broader analysis, as well as an extensive study and investigation of approaches of semi-supervised learning to train deep learning algorithms to predict and classify unknown data in different dataset.

Acknowledgements

The authors thank the NOAA and KNIME for providing to the scientific community publics datasets, leveraging researches and technology development. The authors also thank the research group CS2i and the Reference Center for Artificial Intelligence at SENAI CIMATEC.

Author details

Ilan Figueirêdo, Lílian Lefol Nani Guarieiro
and Erick Giovani Sperandio Nascimento*
SENAI CIMATEC, Salvador, Brazil

*Address all correspondence to: erick.sperandio@fieb.org.br

References

[1] Rodner E, Barz B, Guanche Y, Flach M, Mahecha M, Bodesheim P, et al. Maximally Divergent Intervals for Anomaly Detection. In: ICML Workshop on Anomaly Detection [Internet]. New York, NY, USA; 2016. Available from: http://arxiv.org/abs/1610.06761%0A

[2] Sperandio Nascimento EG, Tavares O, De Souza A. A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance. In: ICAI'2015 - International Conference on Artificial Intelligence [Internet]. Las Vegas, Nevada, USA: ICAI'15 - The 17th International Conference on Artificial Intelligence; 2015. p. 622-8. Available from: https://www.researchgate.net/publication/282330724_A_Cluster-based_Algorithm_for_Anomaly_Detection_in_Time_Series_Using_Mahalanobis_Distance

[3] Wei L, Kumar N, Lolla VN, Keogh EJ, Lonardi S, Ratanamahatana C (Ann). Assumption-Free Anomaly Detection in Time Series. In: 17th International Conference on Scientific and Statistical Database Management, SSDBM [Internet]. Berkeley, CA, USA; 2005. p. 237-42. Available from: https://pdfs.semanticscholar.org/909b/8226968d41f76cc14d0eef2d365572e7a37b.pdf?_ga=2.68813208.804195361.1594613685-826585445.1591805583

[4] Liu H, Zhou J, Xu Y, Zheng Y, Peng X, Jiang W. Unsupervised fault diagnosis of rolling bearings using a deep neural network based on generative adversarial networks. Neurocomputing [Internet]. 2018 Nov;315:412-24. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0925231218308695

[5] Figueiredo IS, Guarieiro LLN, Santos AAB, Nascimento EGS. Algoritmo de aprendizagem de máquina não supervisionado para detecção de anomalias em séries temporais multivariadas aplicadas a temporadas de furacões. In: V Seminário de Avaliação de Pesquisa Científica e Tecnológica [Internet]. Salvador, Bahia: SENAI CIMATEC; 2020. p. 3. Available from: https://www.researchgate.net/publication/343252454_ALGORITMO_DE_APRENDIZAGEM_DE_MAQUINA_NAO_SUPERVISIONADO_PARA_DETECCAO_DE_ANOMALIAS_EM_SERIES_TEMPORAIS_MULTIVARIADAS_APLICADAS_A_TEMPORADAS_DE_FURACOES

[6] Kim M, Ou E, Loh P-L, Allen T, Agasie R, Liu K. RNN-Based online anomaly detection in nuclear reactors for highly imbalanced datasets with uncertainty. Nucl Eng Des [Internet]. 2020 Aug;364(April):110699. Available from: https://doi.org/10.1016/j.nucengdes.2020.110699

[7] Schwabacher M, Oza N, Matthews B. Unsupervised Anomaly Detection for Liquid-Fueled Rocket Propulsion Health Monitoring. J Aerosp Comput Information, Commun [Internet]. 2009 Jul;6(7):464-82. Available from: https://arc.aiaa.org/doi/10.2514/1.42783

[8] Elavarasan D, Vincent DR, Sharma V, Zomaya AY, Srinivasan K. Forecasting yield by integrating agrarian factors and machine learning models: A survey. Comput Electron Agric [Internet]. 2018;155(October):257-82. Available from: https://doi.org/10.1016/j.compag.2018.10.024

[9] Khan S, Liew CF, Yairi T, McWilliam R. Unsupervised anomaly detection in unmanned aerial vehicles. Applied Soft Computing [Internet]. 2019 Oct;83:105650. Available from: https://doi.org/10.1016/j.asoc.2019.105650

[10] Ward CP, Weston PF, Stewart EJC, Li H, Goodall RM, Roberts C, et al.

Condition monitoring opportunities using vehicle-based sensors. In: Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit. 2011.

[11] Yin S, Ding SX, Xie X, Luo H. A review on basic data-driven approaches for industrial process monitoring. IEEE Transactions on Industrial Electronics. 2014.

[12] Yiakopoulos CT, Gryllias KC, Antoniadis IA. Rolling element bearing fault detection in industrial environments based on a K-means clustering approach. Expert Systems with Applications [Internet]. 2011 Mar;38(3):2888-911. Available from: http://dx.doi.org/10.1016/j.eswa.2010.08.083

[13] Wang X-B, Zhang X, Li Z, Wu J. Ensemble extreme learning machines for compound-fault diagnosis of rotating machinery. Knowledge-Based Syst [Internet]. 2019; Available from: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85071672820&doi=10.1016%2Fj.knosys.2019.105012&partnerID=40&md5=a47687f0f999c07d4b3a5a2ed5a8eb2b

[14] Nguyen VH, Cheng JS, Yu Y, Thai VT. An architecture of deep learning network based on ensemble empirical mode decomposition in precise identification of bearing vibration signal. Journal of Mechanical Science and Technology 2019;33(1):41-50.

[15] Abid A, Khan MT, Khan MS. Multidomain Features-Based GA Optimized Artificial Immune System for Bearing Fault Detection. IEEE Trans Syst Man, Cybern Syst [Internet]. 2020 Jan;50(1):348-59. Available from: https://ieeexplore.ieee.org/document/8031077/

[16] Lin J, Keogh E, Lonardi S, Chiu B. A symbolic representation of time series, with implications for streaming algorithms. In: Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery - DMKD '03 [Internet]. New York, New York, USA: ACM Press; 2003. p. 2-11. Available from: http://portal.acm.org/citation.cfm?doid=882082.882086

[17] Mohammad Y, Nishida T. Robust learning from demonstrations using multidimensional SAX. In: 14th International Conference on Control, Automation and Systems (ICCAS) [Internet]. Gyeonggi: IEEE; 2014. p. 64-71. Available from: http://ieeexplore.ieee.org/document/6987960/

[18] Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory [Internet]. 1967 Jan;13(1):21-7. Available from: http://ieeexplore.ieee.org/document/1053964/

[19] Gou J, Ma H, Ou W, Zeng S, Rao Y, Yang H. A generalized mean distance-based k-nearest neighbor classifier. Expert Syst Appl [Internet]. 2019 Jan;115:356-72. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0957417418305293

[20] Efron B. Bootstrap Methods: Another Look at the Jackknife. In: Breakthroughs in Statistics [Internet]. 1992. p. 569-93. Available from: http://link.springer.com/10.1007/978-1-4612-4380-9_41

[21] Efron B, Rogosa D, Tibshirani R. Resampling Methods of Estimation. In: Wright JD, editor. International Encyclopedia of the Social & Behavioral Sciences [Internet]. Second Edi. Oxford: Elsevier; 2015. p. 492-5. Available from: http://www.sciencedirect.com/science/article/pii/B9780080970868421653

[22] Guha S, Mishra N, Roy G, Schrijvers O. Robust random cut forest based anomaly detection on streams. In: 33rd International Conference

on Machine Learning, ICML 2016 [Internet]. 2016. p. 3987-99. Available from: http://proceedings.mlr.press/v48/guha16.pdf

[23] National Hurricane Center Administration. Tropical Depression Nine Public Advisory One [Internet]. NINTH DEPRESSION OF THE SEASON FORMS EAST OF THE LESSER ANTILLES. Miami; 2012. Available from: https://www.nhc.noaa.gov/archive/2012/al09/al092012.public.001.shtml

[24] Avila L. Hurricane Rafael Tropical Cyclone Report [Internet]. Miami; 2013. Available from: http://www.nhc.noaa.gov/data/tcr/AL172012_Rafael.pdf

[25] Blake ES, Kimberlain TB, Berg RJ, Cangia, Losi JP, Beven II JL. Tropical cyclone report Hurricane Sandy [Internet]. National Weather Service, National Hurricane Center. Miami; 2013. Available from: https://www.nhc.noaa.gov/data/tcr/AL182012_Sandy.pdf

3.1.2 Manuscrito 2 - Unsupervised Machine Learning for Anomaly Detection in Multivariate Time Series Data of a Rotating Machine from an Oil and Gas Platform

O presente manuscrito, publicado na revista Journal on Systemics, Cybernetics and Informatics (JSCI), continuou o estudo de investigação dos seis algoritmos identificados no Manuscrito 1. A fundamentação teórica reforçou a lógica de programação dos algoritmos em análise. O experimento utilizou dados de séries temporais multivariadas de alta dimensionalidade, com 28 atributos (sensores). Os dados foram extraídos de turbogeradores da Unidade Flutuante de Armazenamento e Transferência, do inglês *Floating, Production, Storage and Offloading* (FPSO) P-50 da Petrobras. Quatro estudos de casos foram selecionados para os algoritmos detectarem os padrões de mau funcionamento da máquina. A avaliação de desempenho reforçou os resultados obtidos no Manuscrito 1, com destaque para o algoritmo de aprendizagem de máquina não supervisionada C-AMDATS. Além disso, também foi avaliado o tempo de resposta de cada algoritmo, demonstrando que o C-AMDATS tinha o maior tempo de resposta entre todos.

Os resultados parciais deste manuscrito também foram publicados e apresentados na Rio Oil & Gas Expo and Conference 2020 (FIGUEIRÊDO, et al., 2020).

Este artigo foi publicado na revista Journal on Systemics, Cybernetics and Informatics (ISSN 1690-4524). Qualis A2 na área Interdisciplinar.

Link: http://www.iiisci.org/Journal/PDV/sci/pdfs/ZA422HO21.pdf

Unsupervised Machine Learning for Anomaly Detection in Multivariate Time Series Data of a Rotating Machine from an Oil and Gas Platform

[1]Ilan Sousa Figueirêdo[1], Tássio Farias Carvalho[1], Wenisten Dandas da Silva[1], Lílian Lefol Nani Guarieiro[1], Alex Alisson Bandeira Santos[1], Leonildes Soares de Melo Filho[2], Ricardo Emmanuel Vaz Vargas[3], and Erick Giovani Sperandio Nascimento[*1]

[1]*Manufacturing and Technology Integrated Campus – SENAI CIMATEC Salvador, Bahia, Brazil.*
[2]*Repsol Sinopec Brazil – Rio de Janeiro, Rio de Janeiro, Brazil*
[3]*Petróleo Brasileiro S.A. – Vitória, Espiríto Santo, Brazil*

***Abstract*[2]**

Deep Learning (DP) models have been successfully applied to detect and predict failures in rotating machines. However, these models are often based on the supervised learning paradigm and require annotated data with operational status labels (e.g. normal or failure). Furthermore, machine measurement data is not commonly labeled by industry because of the manual and specialized effort that they require. In situations where labels are nonexistent or cannot be developed, unsupervised machine learning has been successfully applied for pattern recognition in large and multivariate datasets. Thus, instead of experts labeling a large amount of structured and/or non-structured data instances (also referred to as Big Data), unsupervised machine learning allows the annotation of the dataset from the few underlying interesting patterns detected. Therefore, we evaluate the performance of six unsupervised learning algorithms for the identification of anomalous patterns from a turbogenerator installed and operating in an oil and gas platform. The algorithms were C-AMDATS, Luminol Bitmap, SAX-REPEAT, k-NN, Bootstrap, and Robust Random Cut Forest. The evaluation performance was quantitatively calculated with seven classification metrics. The C-AMDATS algorithm was able to effectively and better detect the anomalous patterns, and it presented an accuracy of 99%, which leverages the further development of supervised DL models.

Keywords: *Multivariate Time Series, Anomaly Detection, Pattern Recognition, Unsupervised Machine Learning, Rotating Machinery.*

[1] *Corresponding Author. E-mail address: ericksperandio@gmail.com.

[2] We would like to express our deeply felt gratefulness to Professor Dr. Davidson Moreira and Dr. Leandro Andrade for the comprehensive and detailed peer-editing of this document, as well as for the gentle alerts with regards to important issues.

1. Introduction

Over the past decade, advances in technologies and engineering expertise have extended the lifespan of machines and devices of the oil and gas industry. However, machine malfunction and unexpected breakdown are still a reality for many oil companies. For example, in September 2019, Petrobras reported that the P-50 floating production storage and offloading (FPSO) had its production preventively interrupted because of a rupture of the mooring system. Fortunately, there was no record of labor accident or environmental disaster, but it caused an average loss of 20,000 bbl/d (barrels of oil per day), which means U$1,200,000 bbl/d considering an average value of U$60/bbl at the time (PETROBRAS, 2019). Furthermore, Petrobras has estimated a production loss from failures events in the Operational Unit located in the Brazilian state of Espírito Santo during 2016 of 1,514,000 bbl, which corresponds to US$75.7 million considering an average value of US$50/bbl in this period. Despite production losses and the company's financial damage, failures in the oil industry can lead to serious catastrophes, such as the Macondo Incident in 2010 that unfortunately caused the deaths of 11 workers, the sinking of the Deepwater Horizon rig, and massive marine and coastal damage, which marked this incident as one of the largest environmental disasters in the US history (Vargas et al., 2019).

Nevertheless, oil and gas production systems are designed to operate uninterruptedly, as the operating cost of an FPSO is high (approximately U$ 250,000/month) (Perera et al., 2019). Thus, an intelligent maintenance management is crucial for the maintenance of operational activities under required levels of structural integrity and system security, as well for the prevention production losses, environmental accidents, and human casualties and for reduction maintenance costs. However, the maintenance of FPSO machinery can be challenging, since the entire system is in a highly hostile environment.

Predictive Maintenance (PdM) is a specific method of maintainability that provides the integrity of the machinery and establishes the main necessary intervention activities. It uses historical measurement data for early detection of machine deterioration, *i.e.*, long before the malfunction exceeds the acceptable operating limits designed by the project. The method has gained prominence for providing great efficiency in the durability of the system and reducing maintenance costs and logistical footprints (Gombé et al., 2019).

The measurement data of the machine are acquired in real time by multisensor or periodically by inspection; both ways may indicate the health status of the machinery, but normally critical assets are monitored in real time. Thus, computation models designed to implement PdM capabilities can diagnose or predict failures based on a long record of historical data of machine health status. Therefore, with the diagnostic or prognostic information of failures inferred by these models, maintenance managers would have a greater basis in decision making to improve operational performance for scheduling maintenance, reduce unplanned repairs, and minimize downtime.

In theory, the shutdown of machinery only occurs when there is an evidence of deterioration; however, the volume of data generated is large, and conventional diagnostic methods rely mainly on specialized knowledge, which hindes artificial characteristics and other factors that seriously restrict the intelligent and automated development of predictive models (Zhu et al., 2019).

Artificial Intelligence (AI) is an area of research that is currently on the rise and has been used to develop computational models for the detection, diagnosis, and prognosis of failures in dynamic machinery. The main tasks are: (i) to detect the normal or abnormal condition of the machine operation, (ii) to detect incipient failure and identify its cause, and (iii) to predict the development trend of the failures. The goal is to increase the reliability and safety of maintenance in dynamic and complex systems (Jardine et al., 2006; Liu et al., 2018).

Machine learning (within the AI field) has promising tools for pattern recognition of historical data from machine measurement (Caggiano et al., 2019). The tools are conceptualized in two types of learning: (i) supervised and (ii) unsupervised, both being a viable paradigm for nonlinear data analysis (Wachowiak et al., 2019).

In supervised classification learning, the learner is given a stimulus, classifies it, and then it is provided with corrective feedback (Love, 2002). Nevertheless, supervised models are limited by the need to have an annotated dataset for the corrective feedback during the training process, *i.e.*, each data sample must have its corresponding target or label, which can be discrete or continuous and with single or multiple values. However, all of this generated data from machine monitoring sensors turns the annotation into an increasingly complex, harder and thus unfeasible task . Because it usually demands highly specialized human workforce, which is normally overloaded with demand and does not have time for this relevant activity. Thereby, despite all promising publications with supervised DP models for failure classification in rotating machinery (Li et al., 2020; Souza et al., 2021), there is a gap in supervised machine learning models,which points to a desire and opportunity to employ unlabeled data in unsupervised machine learning algorithms (Jati & Georgiou, 2019).

Unsupervised learning is a branch of machine learning that consists of finding interesting information from the raw data without the help of any targets or labels. For instance, dimensionality reduction and clustering are well-known categories of unsupervised learning (Chollet, 2018). These approaches can extract useful features to handle unlabeled data. Dimensionality reduction consists of reducing the high dimensional data, as it captures the essence of the data by projecting the data to a lower dimensional subspace. Clustering algorithms can be used to find natural groupings or patterns. Clustering approaches can also serve as an advantageous data-preprocessing step to identify homogeneous groups on which to build supervised models (Kakarla et al., 2021).

There is an expectation that unsupervised machine learning will prove to be similarly effective in situations where labels are expensive, impractical to collect, or where the prediction target is unknown during the training process (Zhu et al., 2019). Thus, unsupervised algorithms can be implemented directly on raw data with no need of previous training. They only depend on internal clustering of the data without prior knowledge (Torabi Jahromi et al., 2016). This feature is one of the main advantages of unsupervised methods, especially in industrial environments (Yiakopoulos et al., 2011).

Unsupervised learning models have already shown good results for failure recognition in rotating machinery. However, most researches studies in the literature have been

limited to the analysis of only vibration data, *i.e.*, an univariate approach (Abid et al., 2020; Ben Ali et al., 2018; Durbhaka & Barani, 2016; Nguyen et al., 2019; Soualhi et al., 2014; Wang et al., 2019; Yiakopoulos et al., 2011), but there are other numerous useful data for the failure diagnosis of rotating machinery, including for example oil analysis, acoustic emission, pressure and temperature measurement, and microwave energy.

Critical rotating machinery is generally monitored by multi-sensors because of the probability of a more robust representation of the phenomena implicated. However, multivariate data presents a bigger challenge for the use of machine learning algorithms, as they must be able to process and correlate attributes in a greater amount of data (Figueirêdo et al., 2020).

In this paper, we explored a turbogenerator, which is a critical rotating machine for the power generation system of an FPSO and other systems of oil and gas industry, for instance: Liquefied Natural Gas (LNG), pipeline, refinery, and petrochemical (Parrella et al., 2019). It operates on a wide variety of gaseous and liquid fuels, being extremely useful for offshore electricity generation, since oil and gas platforms are usually located hundreds of miles from the coast (Goldmeer et al., 2018).

In this context, this work proposes a comparative performance assessment between six unsupervised machine learning algorithms for pattern recognition and anomaly detection in multivariate time-series data from the oil and gas industry. The goal is to reveal the capacity of unsupervised algorithms to assist experts in the task of data annotation and to leverage the supervised models. We use the labels of the data only for evaluation purposes. The data was acquired from a turbogenerator of an FPSO power generation system.

2. Unsupervised Learning Algorithms

As our goal is to compare the performance of six unsupervised machine learning algorithms, understanding the logic of each one of them can contribute to a correct parameterization of the algorithm and consequently its performance. Thus, we review the concepts and applications of welldeveloped unsupervised algorithms.

2.1. C-AMDATS

The Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance (C-AMDATS) is an unsupervised clustering algorithm designed for pattern recognition or anomaly detection in multivariable time series.

The algorithm starts by dividing the time-series T into a set of groups C of the same size τ, where each subset $C_k = \{t_a, t_{a+1}, \dots, t_b\}$ is sized $|C_k| = \tau$, *i.e.*, $b - a = \tau$ (except the last set, in cases where $|T|$ is not divisible by τ). Afterwards, the algorithm reconstructs C iteratively using its copy C'. For this, the algorithm uses $f(C')$ to determine which set in C is closest to t_i by Mahalanobis distance.

The function $f(C')$ computes the average of the time-series segments within each C'_k group or its centroid, $m_k = (t_a + t_{a+1}, \ldots, t_b)/(b-a)$, and the distance, $d(t_i, m_k)$, from t_i for each m_k centroid, for $1 \leq k \leq |C'|$.. Using these distances, the algorithm moves t_i from its current group in C to the group C_k, where k is the index of the group C'_k whose centroid m_k is the closest to t_i according to $d(t_i, m_k)$.

After the iterative reconstruction of C, the set of groups C is definitely formed, which consists of groups of samples C_k of different sizes and better groups according to their values and distributed along the time axis, which is due to the Mahalanobis function.

In general, clustering algorithms use the Euclidean distance, which tends to form groups with a circular shape (since it does not consider the variance of each dimension of the dataset). Thus, if the shape of the group is more elongated on the x-axis or the y-axis, the general shape of the group will always be circular. However, this circular shape may not be adequate to represent the real shape of the group. Therefore, the C-AMDATS apply another distance to solve this problem. Eq. (1) presents the Mahalanobis distance formula.

$$d_m(x, \mu) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)} \tag{1}$$

Where: $d_m(x, \mu)$ is the Mahalanobis distance between a specific point in the time series and its respective centroid; $x = (x_1, x_2, \ldots, x_n)^T$ is a specific variable in the time-series data, where n is the number of variables; $\mu = (\mu_1, \mu_2, \ldots, \mu_n)^T$ is a certain cluster centroid; and S is the covariance matrix relative to that cluster.

The next step implements the logic of finding the hidden patterns P in the time series T. After all the groups have been discovered, the algorithm compares which groups are similar to each other. This similarity is computed applying the standard deviation σy of the actual values of the T samples, the y coordinate of each centroid, and the clustering factor φ. Thus, if the modulus of the difference between the y coordinate of the centroids of two groups is less than or equal to the product of φ with σy, then these groups can be merged, which means that they will represent the same pattern in P. This task is carried out until every group has been analyzed.

In conclusion, the algorithm computes the probability of each pattern P being interesting in terms of being an anomaly in the time-series T. This is done by computing the anomaly index for each pattern P, which is calculated as the ratio between the total size of the time-series and the size of the pattern present in P. The higher the anomaly score is, the greater the probability of an anomalous behavior in T (Nascimento et al., 2015). Eq. (2) presents the Anomaly Score formula.

$$Anomaly\ Score_{P_i} = \frac{|T|}{|P_i|} \tag{2}$$

Where $Anomaly\ Score_{P_i}$ is the anomaly score of the pattern P_i, $|P_i|$ is the size of the pattern P_i, and $|T|$ is the size of the time series T.

2.2. Luminol Bitmap

Bitmap is an unsupervised learning algorithm in the Luminol library for anomaly detection or time-series correlation. The background of the Bitmap algorithm is based on the approaches of dimensionality reduction by symbolic aggregate approximation (SAX) and time-series bitmaps.

Initially, the algorithm normalizes each time series for a mean of zero and a standard deviation of one, since it is well understood that it is meaningless to compare time series with different offsets and amplitudes. Eq. (3) presents the Z-Score formula:

$$Z_{\mathrm{i}} = \frac{C_i - \bar{C}}{\sigma} \tag{3}$$

Where: Z_i is the Z-Score value of C_i, C_i is a specific sample in the time series data, $\bar{C}$ is the mean of the time-series, and σ is the standard deviation of the time series.

In the next step, the algorithm makes a feature extraction of the raw time series data by converting them into a Piecewise Aggregate Approximation (PAA). The PAA takes the real valued signal, divided into equal sized sections, and calculates the mean value of each section. Thus, by replacing each section with its mean, a reduced dimensionality piecewise constant approximation of the data is already obtained.

A time-series C of length n can be represented in PAA number segments w by a vector $\bar{C} = \bar{C}_1, \ldots, \bar{C}_w$. The ith element of $\bar{C}$ is calculated in Eq. (4):

$$\bar{C}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} C_j \tag{4}$$

After transforming a time-series dataset into PAA, it is applied for a further transformation to obtain a discrete representation. Once the normalized time series has a Gaussian distribution, a discretization technique can be achieved which will produce symbols with equiprobability. The conversion of the PAA number into SAX words is made by the Breakpoints β. Breakpoints are a threshold that will produce a equal-sized areas under the Gaussian curve, i.e., the Gaussian distribution is divided into a bins with equal probability at every bin. A set of break points corresponding to every symbol in SAX alphabet and a correspond to the alphabet size. SAX can produce strings on any alphabet size, but Luminol Bitmaps are defined for sequences with an alphabet size of four.

For instance, Figure 1 shows an example of Gaussian distribution with two areas of equiprobability ($a = 2$), *i.e.*, the probability of a PAA number being in the left side is the same as the right side. We can clearly see β as the threshold between the areas, thus for example, if the PAA value is less than or greater than β, the value will be converted

into A and B SAX symbol, respectively. That is how SAX transformation works (Lin et al., 2003).

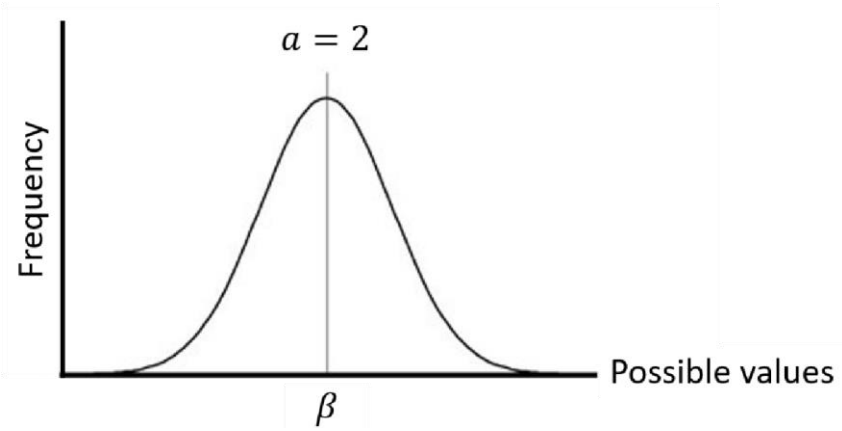


Figure 1. Gaussian curve with two areas of equiprobability.

The conversion into the SAX symbol is made by two slides of short windows. The lead window shows how far to look ahead for anomalous patterns and the lag window shows the size of the memory of the past to memorize.

After converting the original time series into SAX representation, the algorithm will count the frequencies of SAX subwords at the desired level of resolution to color the corresponding pixel of the bitmap grid in a principled way. The frequencies of SAX subwords are normalized [0,1] by the largest value and encode the pixels values to be in the RGB color space.

At this point, the distance between the two bitmap windows is measured and reported as an anomaly score at each time instance. The distance between them is defined as the summation of the square of the distance between each pair of pixels. More formally, for two nxn bitmaps BA and BB, the distance between them are represented in Eq. (5):

$$dist(BA, BB) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(BA_{ij} - BB_{ij}\right)^2 \tag{5}$$

The user must choose both lengths of the sliding window feature N, the number w of equal-sized sections in which to divide the time series N, and the detector score for the anomaly data computed. There is no choice to be made for alphabet size, because a simple alphabetical order of size 4 (*e.g.*: a, b, c, d) was fixedly defined in the Bitmap approach (Wei et al., 2005).

2.3 SAX-REPEAT

This approach consists of expanding the original SAX technique to handle multivariable data. The algorithm apply SAX to every dimension of the data separately and then combine the resulting string by assigning every possible combination of symbols in the resulting dimensionality D strings a unique identifier. The symbol combination leads to a string of length N but with an extended alphabet of length M^D. So, to remain the

requirement that the final string must have an M-symbols alphabet, the algorithm use k-means in the multivariable SAX subwords to perform a clustering with M clusters and replace it with the centroid of its cluster (Mohammad & Nishida, 2014).

K-means is an unsupervised algorithm for pattern recognition based on the distanceof a data point to its cluster centoride in a Euclidean space. It aims to partition n data points into k clusters in which each data point belongs to the cluster with the nearest mean (also called cluster centers or cluster centroid) (Figueirêdo et al., 2020).

So, the SAX-REPEAT algorithm returned a set of groups or patterns from the multivariable dataset by applying SAX and k-means. Moreover, in order to extend this approach for an anomaly detection, we propose the Eq. (2) to compute the anomaly score for each cluster (the same approach used by the C-AMDATS algorithm). Therefore, SAX-REPEAT is now able to recognize interesting patterns in a multivariable dataset with their respective anomaly score, *i.e.*, their probability of being an anomalous pattern.

2.4 k-NN

The k-Nearest Neighbors (k-NN) algorithm is a popular method for classification and regression applications. The principle behind it is to predict the new data from the closest k training samples. The k is the number of neighbors that is predefined by the user as a constant and the distance is calculated in an Euclidean space. In general, the distance applied is an Euclidean function, which is the most common choice, but it can be calculated using other distance functions (Cover & Hart, 1967).

The Euclidean distance between the points $p = (p_1, p_2, \dots, p_n)$ and $q = (q_1, q_2, \dots, q_n)$, in a n-dimensional Euclidean space is defined in Eq. (6):

$$d_e(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{6}$$

Where: $P = (p_1, p_2, \dots, p_n)$ and $q = (q_1, q_2, \dots, q_n)$ are two points in Euclidean n-space. If we compare Eq. (6) to Eq. (1), a small difference can be noted, where Eq. (1) only includes the inverse of the covariance matrix in the calculation.

Ramaswamy et al. (2000) have extended k-NN to an unsupervised approach using the distance to the k^{th} nearest neighbor as an anomaly score. The anomaly score for each data point is the sum of the distance from its k nearest neighbors, called weight (Angiulli & Pizzuti, 2002). Each data point is ranked by its anomaly score and the top n points of the rank are considered anomalous. The proportion of anomaly data is calculated based on contamination parameter times sample number.

The k-NN algorithm depends on three parameters, a metric used to compute the distance between two points (the Euclidean function applied in this study), a k value of the number of neighbors to consider, and contamination to define the threshold on the decision function (Cover & Hart, 1967; Gou et al., 2019).

2.5 Bootstrap

Bootstrap is a particularly useful algorithm to estimate any summary statistics, for example: variance, mean, standard error, and confidence intervals. The name bootstrap concerns the use of the original dataset to generate new datasets Z^*. The algorithm is statistical based and consists in generating samples of size B (called bootstrap samples) from an initial dataset of size n by randomly drawing with replacement B observations. There are two hypotheses that have to be verified to make this approach valid, which are: (i) the size N of the initial dataset should be large enough to capture most of the complexity of the underlying distribution (representativity), and (ii) the size N of the dataset should be large enough compared to the size B of the bootstrap samples so that samples are not too much correlated (independence).

The algorithm uses computational power to generate the bootstrap sample, which is a resampling of size n drawn to replace the original dataset $Z = (Z_1, Z_2, \dots, Z_n)$. The bootstrap sample is represented $Z^* = (Z_1^*, Z_2^*, \dots, Z_n^*)$. Each Z_i^* is one of the original Z values randomly chosen; the chosen probability for each Z value is equipollent, for instance: $Z_1^* = Z_7, Z_2^* = Z_5, Z_3^* = Z_9, Z_4^* = Z_7$, *etc*. Observe that the same original value can be chosen zero, one, or more times; in the example, Z_7 appeared twice, *i.e.*, the selection of Z value is not exclusive. The algorithm generates a larger number of bootstrap samples B of each size n using a random number device to perform the training. The number of bootstrap samples defines the variance of the estimate, *i.e.*, the higher the number is, the better the variance, but, on the other hand, the computational cost increases withincreasing B value (Efron, 1992; Efron et al., 2015).

In this sense, we used Bootstrap to calculate a confidence interval of the time series to get the potential anomaly data. The algorithm initially requests the statistics stored during the training and selects values in the chosen percentile for the confidence interval. The chosen percentile is denoted as δ (Alpha or Significance Level). Eq. (7) and Eq. (8) define the formula to estimate the distribution of δ* for each Bootstrap sample.

$$\delta^* = \bar{x}^* - \bar{x} \tag{7}$$

Where: $\bar{x}^*$ is the mean of an empirical bootstrap sample and $\bar{x}$ is the mean of the original data.

After computing the distribution of the Bootstrap Sample, the confidence interval of the bootstrap distribution can be found. For instance, δ.05 and δ.95 are the 0.05 and 0.95 critical values of δ, which gives a confidence interval of 90%; that confidence interval is defined in Eq. (8)

$$\mathit{C}\text{onfidence interval} = [\bar{x} - \delta_{.05}^*, \bar{x} - \delta_{.95}^*] \tag{8}$$

Where: $\bar{x}$ is the mean of the original data, $\delta_{.05}^*$ is significance level at the 5th percentile, and $\delta_{.95}^*$ is the significance level at the 95th percentile.

Thus, in order to obtain a very accurate estimate of $\delta^*_{.05}$ and $\delta^*_{.95}$, it is important to generate a large number of bootstrap samples.

2.6 Robust Random Cut Forest (RRCF)

RRCF is an ensemble algorithm for detecting anomaly data points. The algorithm takes a set of random data points (Random), cuts them to the same number of points, and creates trees (Cut). Then it looks at all trees together (Forest) to determine whether a particular data point is an anomaly.

The user can choose the number of trees and the size of each tree. The size corresponds to the number of data points that each tree has which is randomly sampled from the original dataset. Once the forest is developed, the algorithm injects the new data point p inside the trees and start to cut to compute the average depth of the point across the collection of trees. Point p is labeled as an anomaly if the score overtakes the threshold, which corresponds to the average depth across the trees (Guha et al., 2016).

The comparative performance evaluation was done using multivariate time series data from the oil and gas industry. The data comes from a rotating machine of the P-50 FPSO. The platform is located in the Albacoara Leste field (latitude 22º05'04 " S and longitude 039º49'45 " W) in the Campos Basin, 120 km from the coast of the state of Rio de Janeiro, Brazil. The P50 is operated by Petrobras and Repsol Sinopec Brazil.

The platform has many critical machines that are constantly monitored for maintenance purposes. In this sense, multivariate time series data was collected from a turbogenerator of the power generation system. The datasets refer to the turbogenerator health status. The machine consists of a multi-stage gearbox, a generator, and a turbine with an axis rotation speed of 6,000 rpm.

The measurement data come from 28 different sensors installed in the gearbox, generator, and electrical parts of the turbogenerator, as given in Table 1. The variables collected are diverse, such as vibration, temperature, pressure, rotation, current, and reactive power. The high number of sensors installed on the machine demonstrates how important it is to the system.

Our goal was to evaluate the performance of unsupervised algorithms on multivariate time series data, as usually several parameters are collected from critical machines for a better representation of phenomena. However, the greater the amount of data, the more difficult human analysis is, especially in real time. However, it also presents a challenge for machine learning algorithms because of the increased volume of data and attributes to be correlated.

Table 1. List of variables collected, including sensor number (N), description, and measuring units.

N	Description	Unit
1	Lube oil supply pressure	kPa
2	Input shaft speed	RPM
3	Generator reactive power	VAr
4	Generator radial bearing temperature 1 DE	°C
5	Generator radial bearing temperature 2 DE	°C
6	Generator radial bearing NDE temperature 1	°C
7	Generator radial bearing NDE temperature 2	°C
8	Gearbox radial bearing temperature 1 DE	°C
9	Gearbox radial bearing temperature 2 DE	°C
10	Gearbox high shaft bearing DE X temperature	°C
11	Gearbox high shaft bearing DE Y temperature	°C
12	Gearbox high shaft bearing NDE X temperature	°C
13	Gearbox high shaft bearing NDE Y temperature	°C
14	Gearbox shaft bearing NDE X temperature	°C
15	Gearbox shaft Bearing NDE Y Temperature	°C
16	Gearbox thrust bearing temperature 1	°C
17	Gearbox thrust bearing temperature 1	°C
18	Gearbox thrust bearing temperature 2	°C
19	Gearbox thrust bearing temperature 2	°C
20	Lube oil supply temperature	°C
21	Gearbox frame accelerometer 1	g
22	Gearbox frame accelerometer 2	g
23	Generator DE frame accelerometer	g
24	Generator NDE frame accelerometer	g
25	Gearbox X DE radial vibration of high shaft	µm P-P
26	Gearbox Y DE radial vibration of high shaft	µm P-P
27	Gearbox X DE radial vibration of low shaft	µm P-P
28	Gearbox Y DE radial vibration of low shaft	µm P-P

Four cases of multivariable time series data was acquired from the turbogenerator. All of them were collected at a rate of 1 to 30 samples per second. In summary, four datasets of real cases were structured for this study:

Dataset I. Started on 07/17/2018 at 12:20:00 PM and ended on 07/19/2018 at 02:21:00 PM (2 days, 2 hours, and 1 minute) and comprising 82,174 data points;
Dataset II. Started on 06/25/2018 at 03:26:00 PM and ended on 27/06/2018 at 05:22:00 PM (2 days, 1 hour, and 56 minutes) and comprising 79,411 data points;
Dataset III. Started on 06/29/2018 at 07:50:00 AM and ended on 07/01/2018 at 04:05:00 PM (2 days, 8 hours, and 15 minutes) and comprising 91,116 data points;

Dataset IV. Started on 08/01/2018 at 04:00:00 AM and ended on 08/03/2018 at 07:39:00 AM (2 days, 3 hours, and 39 minutes) and comprising 83,297 data points.

As our focus is to evaluate the performance of unsupervised machine learning algorithms to assist experts on labeling the normal/anormal data, all datasets contain normal data and abnormal events that were indicated in operational reports. However, we have noted that the operator recorded the time after they observed the abnormal event and performed immediate follow-up actions, and thus some inaccurate records may be present. In addition, usually the control room clock does not display seconds (*i.e.*, one-minute resolution). Depending on the complexity of the abnormal event and the operator's expertise, the time difference between the exact abnormal event and the recorded time could be several seconds to several minutes.

As a data pre-processing step, a minute-to-minute resample was made to the interval of data points. We assume that all data points within the same one-minute interval would be equivalent to the average of the value. Eq. (9) displays the preprocessing formula.

$$\{t_0 \leq t(x_i) < t_0 + 1min\} \rightarrow \bar{x} = \sum_{i=1}^{n} \left(\frac{x_i}{n}\right) \tag{9}$$

Where t_0 is the initial time of 1 minute interval, $t(x_i)$ is the time of the data point i, n is the amount number of data points between t_0 and $t_0 + 1min$, and $\bar{x}$ is the average of the value of data points between t_0 and $t_0 + 1min$.

On July 18, 2018, the turbogenerator tripped because of the high vibration signal measured by sensor 28. Figure 2 shows the multi-sensor signals of the machine collected during this anomalous behavior (Dataset I). The vertical red lines indicate the beginning of the malfunction until the return of the operation, which lasted at total of 2 hours and 1 minute. This event was recorded in the operational reports as two types of failures that occurred sequentially, which were: trip by vibration and trip by flame loss. Nevertheless, we can note that there are different sensors that show significantly distinct patterns and trends. Moreover, a comparison between this operation record and other records – as shown in Figure 3, Figure 4, and Figure 5 –from the same sensors show that they may behave quite differently under different operating circumstances. This complex data structure motivates us to explore the performance of unsupervised machine learning algorithms.

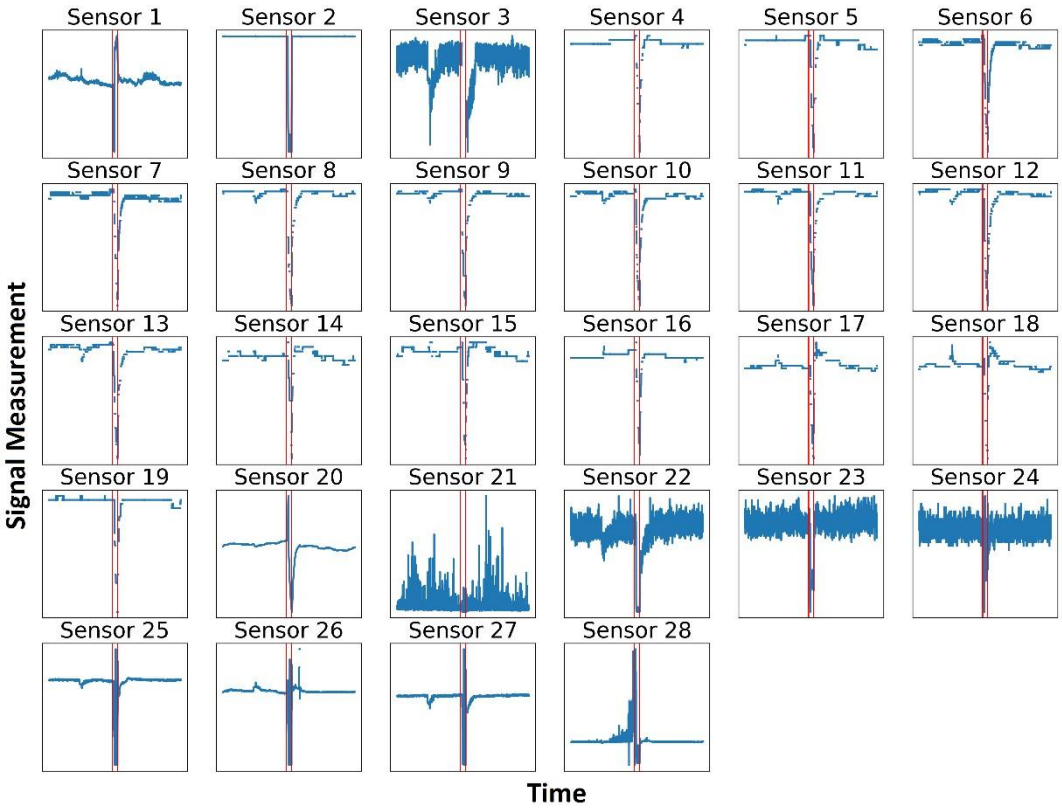


Figure 2. Multi-sensor signals of Dataset I with a recorded anomaly marked between the vertical red lines. Each sensor description can be found in Table 1 according to the sensor number.

The failure in Figure 2 is a typical case of machine degradation that occurred gradually, *i.e.*, in this case, predictive models can be used to detect the imminence of the failure to anticipate it. Thus, unsupervised algorithms can recognize interesting\anomalous patterns to assist specialists in data labeling for further training of DP models for maintenance purposes. All six algorithms were applied in the same data to verify their ability to detect the same failure signal.

In Dataset II, on June 26, 2018, the turbogenerator had another trip because of the low lube supply pressure measured by Sensor 1. Figure 3 shows the multi-sensor signals collected from the machine operation with this anomaly data. The operational reports described three different types of failure that occurred sequentially: trip during alignment of the exchanger, trip by gearbox vibration, and trip by flame loss.

The beginning of the anomaly function until the return of operation of the machine is indicated between the vertical red lines in Figure 3, which lasted 2 hours.

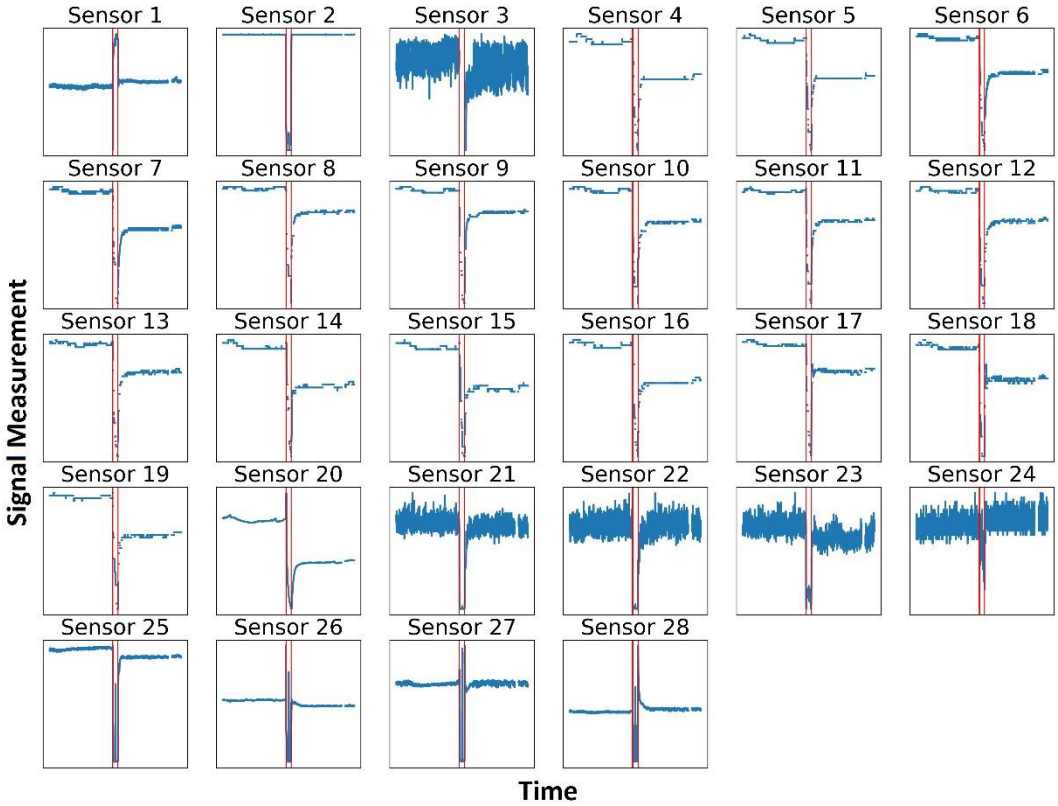


Figure 3. Multi-sensor signals of Dataset II with a recorded anomaly marked between the vertical red lines. Each sensor description can be found in Table 1 according to the sensor number.

In Dataset III, on June 30, 2018, the machine had another unscheduled stop. Figure 4 shows the multi-sensor signals collected before, during, and after the machine operation at this specific unexpected stop. The operational reports described two different types of failure that occurred sequentially: trip by pressure differential in the pump filter and trip by vibration. In Figure 4, the red vertical lines represent the beginning of the anomalous behavior recorded in the operational reports until the return of the machine, which lasted 8 hours and 15 minutes.

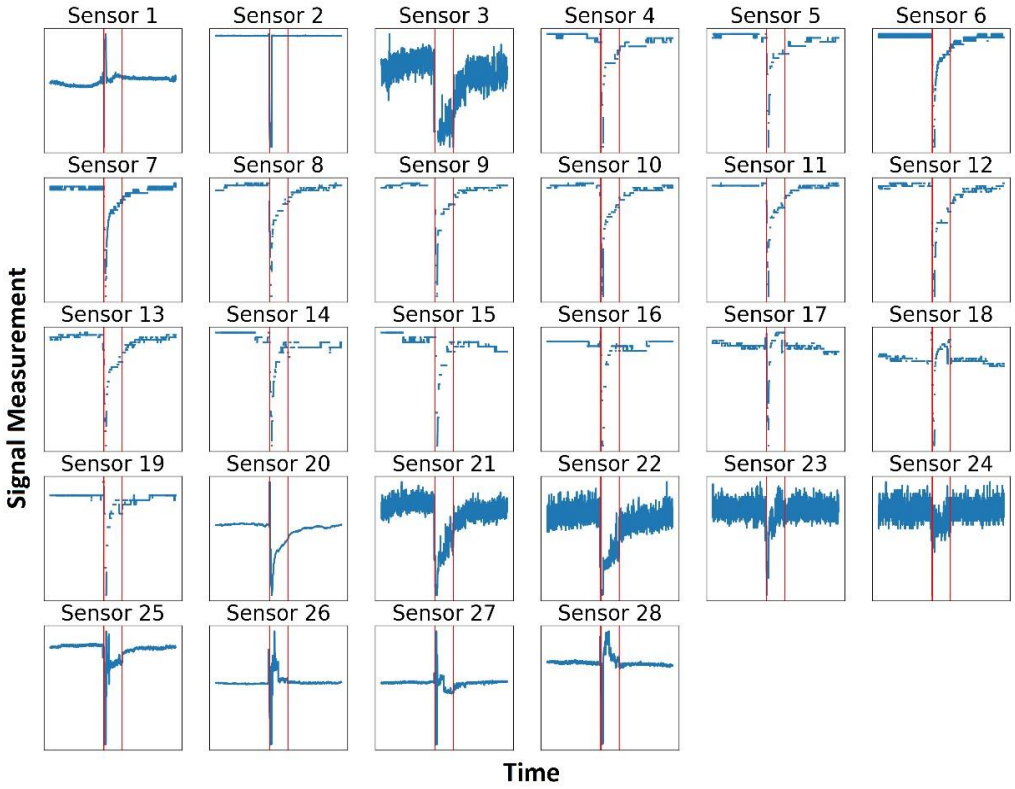


Figure 4. Multi-sensor signals of Dataset III with a recorded anomaly marked between the vertical red lines. Each sensor description can be found in Table 1 according to the sensor number.

Lastly, in Dataset IV the machine shuts down unexpectedly because of a trip by Vacuum Solenoid Valve (VSV) failure on August 2, 2018. Figure 5 shows the multi-sensor signals that contain the anomaly data that caused the machine shutdown, which lasted 3 hours and 35 minutes.

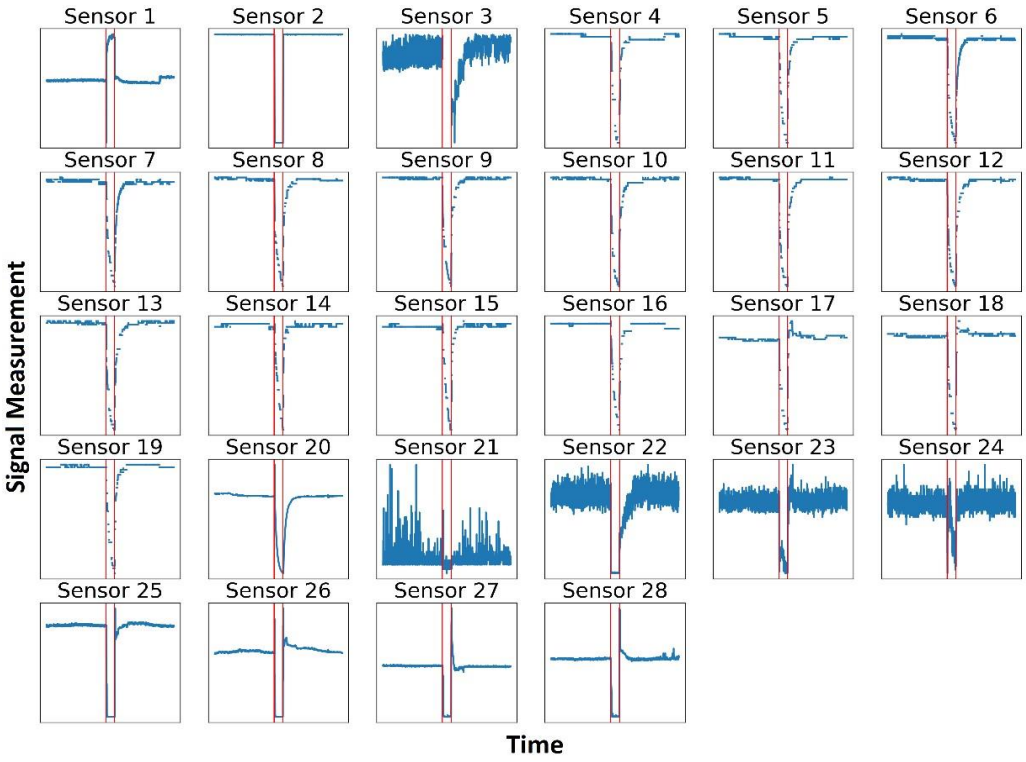


Figure 5. Multi-sensor signals of Dataset IV with a recorded anomaly marked between the vertical red lines. Each sensor description can be found in Table 1 according to the sensor number.

Table 2 shows the statistics of each dataset and their respective failure signals. We can note that the datasets are highly unbalanced with 3.86 to 14.65% of anomaly data. Unbalanced data is quite common in real machine measurement data, as the machine is expected to operate most of the time in good condition. Furthermore, the data has few Not a Number[3] (NaN) records; Dataset II is the highest one, with 5.81%. However, the amount of NaN in the failure zone is quite significant: Dataset III has the highest proportion with 15.83%, and Dataset II has the lowest one with 6.10%.

Table 2. Summary of the dataset statistics.

Dataset Id	Data	Data Points (#)	NaN (#)	Variables (#)	Rows (#)
I	Total	82,174	1,882	28	3,002
	Failure	3,175	241	28	122
II	Total	79,411	4,617	28	3,002
	Failure	3,107	281	28	121
III	Total	91,116	3,412	28	3,376
	Failure	13,348	540	28	496
IV	Total	83,297	3,391	28	3,096
	Failure	5,615	433	28	216

NaN: Not-a-Number.

[3] Not a Number means missing data points.

A linear interpolation was performed to fill the NaN because most of the algorithms are sensitive to missing data points. The reason for existing NaN in the dataset is because the 28 sensors are not synchronized to collect data at the same sample rate.

4. Experimental Setup

Experimental setup in the field of machine learning is an optimization of parameters or adjustment of the algorithm. In summary, it concerns in choosing a set of ideal parameters to optimize the performance of the algorithm.

Each algorithm has hyperparameters that need to be adjusted to each case study. Thus, expertise in the algorithm logic and some attempts at success and error are necessary to find good values among them.

Figueirêdo et al. (2020) have reported difficulty in fine tuning the algorithms manually. Thus, we have implemented a loop in the execution of algorithms that varies the parameters in each iteration, which generates different combinations of parameters. The following combinations are defined in Table 3.

Table 3. Parameter tuning.

Algorithm	Parameter combination
C-AMDATS	• Initial Cluster Size (minutes): [15, 20, 25, 30] • Clustering Factor: [0.5, 1, 1.4, 1.6, 1.8, 2.0]
Luminol Bitmap	• Detector Score: [0.1,0.5,1.0] • Lead / Lag Size: [1, 10, 100, 1000] • Precision: [1, 10, 100] • Chunk Size: [1, 10, 100]
SAX-REPEAT	• Window Size: [1, 10, 100, 500, 1000] • PAA Size: [1, 2, 3, 4] • Alphabet Size: [1, 2, 3, 4]
k-NN	• K: [3, 5, 10, 50, 100, 150, 300, 500] • Contamination: [0.1, 0.2, 0.3]
Bootstrap	• Confidence Interval: [0.90, 0.95, 0.99] • Iteractions: [100, 200, 400, 500, 600, 1000, 2000, 5000]
RRCF	• Number of Trees: [55, 110, 220] • Tree Size: [128, 256] • Shingle Size: [4]

The RRCF had the lowest number of iterations because of the small number of parameters and the little variation in results during the loop. The metric score to define the best model setting was Area Under the Precision-Recall Curve (AUC-PRC). Section 5 describes AUC-PRC in more detail. Table 4 summarizes the best parameter settings of the presented algorithms in the four case studies.

We cannot say that our proposed parameterization is the best solution to the problem because it would be necessary to test all existing possibilities of hyperparameters, which would make this study exhaustive and with a high level of computational cost.

As the different cases analyzed are data from the same machine and the same multiple sensors, but for different moments of time, the parameters values in Table 4 did not vary much between them. This suggests that for the same applications the parameters may vary little, which allows the skipping of the step of finding the best parameters and the reduction of the computational power once the application is already known.

Table 4. Parameter settings of the unsupervised algorithms.

Algorithm	Parameter	Cases			
		#1	#2	#3	#4
C-AMDATS	ICS (min)	30	30	30	30
	Cluster Factor	2.0	2.0	1.8	2.0
Luminol Bitmap	Detector Score	0.1	0.1	0.1	0.1
	Precision	10	10	10	10
	Window Size(1)	100	100	100	100
	Chunk Size	1	21	10	1
SAX-REPEAT	Window Size	1	1	1	1
	PAA Size	1	1	1	1
	Alphabet Size	4	4	5	4
k-NN(2)	k	300	300	500	500
	Contamination	0.1	0.1	0.2	0.1
Bootstrap	CI	0.99	0.99	0.99	0.99
	Iteractions	400	200	200	400
RRCF(3)	Number of Trees	110	110	110	220
	Tree Size	256	256	128	256

ICS: Initial Cluster Size, PAA: Piecewise Aggregate Approximations, k: Number of Neighbors, CI: Confidence Interval, (1) same value applied for Lag window and Lead window, (2) Euclidean distance applied for every case, (3) Shingle Size = 4 for every cases.

The algorithms were executed on the supercomputer named AIRIS (Artificial Intelligence Repsol Sinopec Brazil Integrated System) at the Supercomputing Center for Industrial Innovation (CS2I) at SENAI CIMATEC. The AIRIS processor model is the Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz and has 376 GB RAM memory. All algorithms have been implemented with Python 3.6.

5. Metrics Employed for Evaluation Performance

In this paper we evaluate the performance of the unsupervised algorithms for identifying the same anomaly patterns by seven metrics. It is prudent to measure the performance with a set of metrics to avoid any bias deviation. The evaluation compares the real data points (classified by experts) against the predicted data points (predicted by the unsupervised algorithms). The seven metric are:

- Accuracy (ACC): considering all normal and faulty samples, ACC can be computed as:

$$ACC = \frac{TP + TN}{n_{total}} \tag{10}$$

Where: TP and TN are respectively the number of true positives and true negatives in samples, and n_{total} is the overall number of samples.

- Precision (PR): indicates the true positive value compared to the false negative; PR can be calculate as:

$$PR = \frac{TP}{TP + FP} \tag{11}$$

Where: FP is the number of false positives.

- Recall (REC): or True Positive Ratio (TPR) indicates the proportion of anomalies that are correctly detected out of all anomalies. Normally, it is a high prioritized metric since the failure of anomaly detection (false negatives) leads to much more deleterious results than false alarms (false positives) in industrial applications. REC can be calculated as:

$$REC = \frac{TP}{TP + FN} \tag{12}$$

Where FN is the number of false negatives.

- Specificity (SP): demonstrates the capacity of the model to predict the true negative over false positives, and it can be computed as:

$$SP = \frac{TN}{TN + FP} \tag{13}$$

- F1-Score (F1): is a harmonic average between REC and PR, and it can be calculated as:

$$F1 = \frac{2 \times (PR \times REC)}{PR + REC} \quad (14)$$

- Area Under the Receiver Operating Characteristics Curve (AUCROC): provides a relative tradeoff between the False Positive Rate (FPR) and the TPR. FPR is the equivalent of $1 - SP$;

- AUC-PRC: provides a relative tradeoff between the PR and the REC. It is an important metric for assessing unbalanced datasets, being a great advantage over the others metrics, because the vast majority of real data has a higher volume of normal data in relation to abnormal data.

All metrics above take a value between 0 and 1. A perfectly inaccurate anomaly detection model has a value of 0; a model that makes random guesses has a value of 0.5; and a perfectly accurate model has a value of 1, *i.e.*, the higher the metric value, the better is the anomaly detection performance.

The performance evaluation would not be properly fair as the Luminol, kNN, Bootstrap, and RCCF algorithms made an univariable analysis, unlike C-AMDATS and SAX-REPEAT that have a multivariate approach. Thus, to obtain a more appropriate analysis, the metrics were calculated for all proposed variables and then we extracted an average evaluation, except for C-AMDATS and SAX-REPEAT.

6. Results and Discussion

In this section, we present the results for all six unsupervised machine learning algorithms described in Section 2. The algorithms were applied in four different cases studies with multivariate time series data, described in Section 3. Therefore, the goal is to investigate how each unsupervised algorithm can work to detect anomalies.

The C-AMDATS and SAX-REPEAT algorithms are designed to find interesting/anomalous patterns in a multivariate time series data, and each pattern of interest is assigned an anomaly index. Thus, patterns with the highest indexes are selected as anomalous patterns, unlike the other algorithms that are designed to find anomaly and normal data.

The evaluation performance is summarized in Table 5. We can see that the C-AMDATS was the one that stood out among the other algorithms. The algorithm achieved the best performance in all seven metrics. SAXREPEAT had the second best performance. A rank in decreasing order of performance the algorithms would be: (i) C-AMDATS, (ii) SAX-REPEAT, (iii) Bootstrap, (iv) k-NN, (v) Luminol Bitmap, and (vi) RRCF. Both CAMDATS and SAX-REPEAT algorithms with a multivariate analysis were intrinsically superior. However, more case studies must be carried out to affirm the superiority of the algorithms studied here.

Table 5. Unsupervised machine learning performance in four real cases (the best performance is highlighted in bold).

Algorithm	Metric	Cases (%)				Mean (%)	SD (%)
		#1	#2	#3	#4		
C-AMDATS	ACC	**99**	**100**	**99**	**99**	**99**	**1**
	PR	**80**	**100**	**94**	**90**	**91**	**8**
	REC	**98**	**99**	**97**	**100**	**99**	**1**
	F1	**88**	**100**	**95**	**95**	**94**	**5**
	SP	**98**	**99**	**97**	**100**	**99**	**1**
	AUC-ROC	**99**	**100**	**98**	**100**	**99**	**1**
	AUC-PRC	**79**	**99**	**92**	**90**	**90**	**8**
Luminol Bitmap	ACC	61	68	70	75	69	6
	PR	14	19	43	33	27	13
	REC	100	100	91	99	98	4
	F1	24	31	55	48	39	14
	SP	100	100	91	99	98	4
	AUC-ROC	80	84	79	86	82	3
	AUC-PRC	14	19	39	33	26	12
SAX-REPEAT	ACC	96	100	98	98	98	2
	PR	47	92	95	82	79	22
	REC	68	99	90	96	88	14
	F1	56	95	92	89	83	18
	SP	68	99	90	96	88	14
	AUC-ROC	82	99	95	97	93	8
	AUC-PRC	34	91	87	79	73	26
k-NN	ACC	93	95	92	96	94	2
	PR	34	46	72	65	54	17
	REC	65	91	85	90	83	12
	F1	44	60	77	75	64	15
	SP	65	91	85	90	83	12
	AUC-ROC	79	93	89	93	89	6
	AUC-PRC	25	43	65	60	48	18
Bootstrap	ACC	95	94	94	97	95	2
	PR	51	55	89	78	68	18
	REC	62	87	66	88	76	14
	SP	54	65	73	82	69	12
	F1	62	87	66	88	76	14
	AUC-ROC	79	91	82	93	86	7
	AUC-PRC	35	52	65	71	56	16
RRCF	ACC	91	91	82	89	88	4
	PR	17	16	27	21	20	5

Algorithm	Metric	Cases (%)				Mean (%)	SD (%)
		#1	#2	#3	#4		
	REC	27	27	13	20	22	7
	SP	20	20	17	20	19	1
	F1	27	27	13	20	22	7
	AUC-ROC	61	60	53	57	58	3
	AUC-PRC	8	8	16	10	11	4

SD: Standard Deviation

With these outcomes, C-AMDATS demonstrated the highest capacity to detect the anomalous and normal patterns present in the multivariate time series. The algorithm revealed the highest and lowest value in all metrics for the mean and standard deviation, respectively. Therefore, with these automatically detected patterns, experts would be able to label a large mass of data in a few moments and with a high accuracy, as well as reduce the many hours of manual and tedious efforts of data annotation tasks.

The SAX-REPEAT and the Bootstrap algorithms performed well for some case studies. The SAX-REPEAT detected the anomalous data in case 2, case 3, and case 4; however, it was unsuccessful in case 1. Bootstrap detected only anomalous data from case 4. The remaining algorithms scored low for all cases, as shown in Table 5.

In addition to evaluating the performance of each algorithm, it is also essential to analyze processing time, which is directly proportional to computational cost. Thus, Table 6 shows this comparison:

Table 6. Processing time of each algorithm (the fastest algorithm is highlighted in bold).

Algorithm	Time of each case study (s)				Mean (s)	SD (s)
	#1	#2	#3	#4		
C-AMDATS	429.2	73.9	703.5	8,439.0	2,411.4	4,026.7
Bitmap	10.3	10.2	20.4	9.8	12.7	5.1
SAX-REPEAT	18.3	201.2	184.8	221.7	156.5	93.3
k-NN	11.5	8.6	14.8	14.3	12.3	2.9
Bootstrap	**3.0**	**3.5**	**3.3**	**3.7**	**3.4**	**0.3**
RRCF	1,625.7	1,519.1	2,449.7	5,849.1	2,860.9	2,035.1

SD: standard deviation

Bootstrap was the fastest algorithm with 3.4 and 0.3 seconds of mean and standard deviation, respectively. A fast algorithm enables real-time analysis. However, the processing time for RRCF and C-AMDATS were very high with an average of 2,860 seconds and 2,411 seconds, respectively. as CAMDATS showed the best performance (Table 5), it would be interesting to optimize its structure to speed up its processing time, such as parallelization of the calculation of the inverse of the covariance matrix and use of Multiple Graphics Processing Units (GPU).

7. Conclusion

In this work, we have demonstrated the effectiveness of a multivariate analysis using six different unsupervised machine learning algorithms in a real system from an FPSO. Our goal was to compare the performance of the algorithms to automatically identify anomalous patterns in multivariate time series data. The experimental results showed that unsupervised algorithms, such as C-AMDATS, have high ability to recognize and isolate abnormal events of rotating machinery, thus showing their great capacity to automatically label raw data with unsupervised learning.

Therefore, instead of experts spending several hours of hard and tedious work labeling a large amount of data− as shown in Figure 2, Figure 3, Figure 4, and Figure 5 −, they would just have to label a few patterns recognized by unsupervised learning with demonstrated high performance, such as C-AMDATS which had 99% of ACC. However, C-AMDATS revealed the need to speed up its execution time.

In conclusion, unsupervised learning can leverage the development of DP models for industrial applications, which normally only have unlabeled data. This can be especially applied in the oil and gas industry as it has a large number of multivariate datasets.

Future works include the extension of this analysis to more real cases in the oil and gas industry aiming to develop a broader assesment, as well as an extensive study and investigation of approaches of semi-supervised learning to train DP algorithms to predict and classify unknown data in different mechanical failure modes of rotating machinery for PdM tasks.

8. Acknowledgment

The authors would like to thank the partner companies Repsol Sinopec Brasil and Petrobras for providing the data and all the support given. The authors also thank the CS2I for providing the computational resources needed to develop this work and access to the AIRIS supercomputer, and the Reference Center for Artificial Intelligence, both at SENAI CIMATEC. This work was partially supported by the Brazilian National Agency of Petroleum, Natural Gas and Biofuels (ANP). We would like to express our deeply felt gratefulness to Professor Dr. Davidson Moreira from SENAI CIMATEC and Dr. Leandro Andrade from SENAI CIMATEC for the comprehensive and detailed peer-editing of this document.

References

Abid, A., Khan, M. T., & Khan, M. S. (2020). Multidomain Features-Based GA Optimized Artificial Immune System for Bearing Fault Detection. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(1), 348–359. https://doi.org/10.1109/TSMC.2017.2746762

Angiulli, F., & Pizzuti, C. (2002). Fast Outlier Detection in High Dimensional Spaces. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (pp. 15–27). https://doi.org/10.1007/3-540-45681-3_2

Ben Ali, J., Saidi, L., Harrath, S., Bechhoefer, E., & Benbouzid, M. (2018). Online automatic diagnosis of wind turbine bearings progressive degradations under real experimental conditions based on unsupervised machine learning. *Applied Acoustics*, *132*, 167–181. https://doi.org/10.1016/j.apacoust.2017.11.021

Caggiano, A., Zhang, J., Alfieri, V., Caiazzo, F., Gao, R., & Teti, R. (2019). Machine learningbased image processing for on-line defect recognition in additive manufacturing. *CIRP Annals*, *68*(1), 451–454. https://doi.org/10.1016/j.cirp.2019.03.021 Chollet, F. (2018). Deep Learning with Phyton. In *Manning*.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

Durbhaka, G. K., & Barani, S. (2016). Fault behaviour pattern analysis and recognition. *2016 International Conference on Information Science (ICIS)*, 191–193. https://doi.org/10.1109/INFOSCI.2016.7845325

Efron, B. (1992). Bootstrap Methods: Another Look at the Jackknife. In *Breakthroughs in Statistics* (Vol. 7, Issue 1, pp. 569–593). https://doi.org/10.1007/978-1-4612-4380-9_41 Efron, B., Rogosa, D., & Tibshirani, R. (2015). Resampling Methods of Estimation. In J. D. Wright (Ed.), *International Encyclopedia of the Social & Behavioral Sciences* (Second Edi, pp. 492–495). Elsevier. https://doi.org/10.1016/B978-0-08-097086-8.42165-3

Figueirêdo, I. S., Guarieiro, L. L. N., & Nascimento, E. G. S. (2020). Multivariate Real Time Series Data Using Six Unsupervised Machine Learning Algorithms. In *Anomaly Detection - Recent Advances, Issues and Challenges [Working Title]* (Issue tourism, p. 13). IntechOpen. https://doi.org/10.5772/intechopen.94944

Goldmeer, J., Sanz, A., Adhikari, M., & Hundal, A. (2018). *Gas to Power: The art of the Possible The Fuel Flexibility of GE Power Aeroderivative Gas Turbines*. https://www.ge.com/content/dam/gepower-pw/global/en_US/documents/GasPower/gasturbines/GEA34108_Aero_Fuel_Flexibility_Whitepaper_Final.pdf

Gombé, B. O., Mérou, G. G., Breschi, K., Guyennet, H., Friedt, J. M., Felea, V., & Medjaher, K. (2019). A SAW wireless sensor network platform for industrial predictive maintenance. *Journal of Intelligent Manufacturing*, *30*(4), 1617–1628. https://doi.org/10.1007/s10845017-1344-0

Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., & Yang, H. (2019). A generalized mean distance-based k-nearest neighbor classifier. *Expert Systems with Applications*, *115*, 356–372. https://doi.org/10.1016/j.eswa.2018.08.021

Guha, S., Mishra, N., Roy, G., & Schrijvers, O. (2016). Robust random cut forest based anomaly detection on streams. *33rd International Conference on Machine Learning, ICML 2016*, *48*, 3987–3999. http://proceedings.mlr.press/v48/guha16.pdf

Jardine, A. K. S., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

Jati, A., & Georgiou, P. (2019). Neural Predictive Coding Using Convolutional Neural Networks Toward Unsupervised Learning of Speaker Characteristics. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *27*(10), 1577–1589. https://doi.org/10.1109/TASLP.2019.2921890

Kakarla, R., Krishnan, S., & Alla, S. (2021). Unsupervised Learning and Recommendation Algorithms. In *Applied Data Science Using PySpark* (pp. 251–298). Apress. https://doi.org/10.1007/978-1-4842-6500-0_7

Li, Y., Du, X., Wan, F., Wang, X., & Yu, H. (2020). Rotating machinery fault diagnosis based on convolutional neural network and infrared thermal imaging. *Chinese Journal of Aeronautics*, *33*(2), 427–438. https://doi.org/10.1016/j.cja.2019.08.014

Lin, J., Keogh, E., Lonardi, S., & Chiu, B. (2003). A symbolic representation of time series, with implications for streaming algorithms. *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery - DMKD '03*, 2–11. https://doi.org/10.1145/882082.882086

Liu, R., Yang, B., Zio, E., & Chen, X. (2018). Artificial intelligence for fault diagnosis of rotating machinery: A review. In *Mechanical Systems and Signal Processing* (Vol. 108). https://doi.org/10.1016/j.ymssp.2018.02.016

Love, B. C. (2002). Comparing supervised and unsupervised category learning. *Psychonomic Bulletin & Review*, *9*(4), 829–835. https://doi.org/10.3758/BF03196342

Mohammad, Y., & Nishida, T. (2014). Robust learning from demonstrations using multidimensional SAX. *14th International Conference on Control, Automation and Systems (ICCAS)*, 64–71. https://doi.org/10.1109/ICCAS.2014.6987960

Nascimento, E. G. S., Tavares, O., & De Souza, A. (2015). A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance. *ICAI'2015 - International Conference on Artificial Intelligence*, 622–628. https://www.researchgate.net/publication/282330724_A_Cluster-based_Algorithm_for_Anomaly_Detection_in_Time_Series_Using_Mahalanobis_Distance

Nguyen, V. H., Cheng, J. S., Yu, Y., & Thai, V. T. (2019). An architecture of deep learning network based on ensemble empirical mode decomposition in precise identification of bearing vibration signal. *Journal of Mechanical Science and Technology*, *33*(1), 41–50. https://doi.org/10.1007/s12206-018-1205-6

Parrella, I., Bardi, F., Salerno, G., Gronchi, D., Cannavò, M., & Sparacino, E. (2019, November 11). Using Analytics to Assess Health Status of DLE Combustion Gas Turbines. *Abu Dhabi International Petroleum Exhibition & Conference*. https://doi.org/10.2118/197679-MS

Perera, L. P., Machado, M. M., Valland, A., & Manguinho, D. A. P. (2019). Failure intensity of offshore power plants under varying maintenance policies. *Engineering Failure Analysis*, *97*(February 2018), 434–453. https://doi.org/10.1016/j.engfailanal.2019.01.011

PETROBRAS. (2019). *Petrobras informa sobre ocorrência na P-50*. PETROBRAS; Wiley Online Library. https://www.agenciapetrobras.com.br/Materia/ExibirMateria?p_materia=981253&p_editoria=8

Ramaswamy, S., Rastogi, R., & Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. *SIGMOD Record (ACM Special Interest Group on Management of Data)*. https://doi.org/10.1145/335191.335437

Soualhi, A., Razik, H., Clerc, G., & Doan, D. D. (2014). Prognosis of bearing failures using hidden markov models and the adaptive neuro-fuzzy inference system. *IEEE Transactions on Industrial Electronics*, *61*(6), 2864–2874. https://doi.org/10.1109/TIE.2013.2274415

Souza, R. M., Nascimento, E. G. S., Miranda, U. A., Silva, W. J. D., & Lepikson, H. A. (2021). Deep learning for diagnosis and classification of faults in industrial rotating machinery. *Computers & Industrial Engineering*, *153*(N/A), 107060. https://doi.org/10.1016/j.cie.2020.107060

Torabi Jahromi, A., Er, M. J., Li, X., & Lim, B. S. (2016). Sequential fuzzy clustering based dynamic fuzzy neural network for fault diagnosis and prognosis. *Neurocomputing*, *196*(N/A), 31–41. https://doi.org/10.1016/j.neucom.2016.02.036

Vargas, R. E. V., Munaro, C. J., Ciarelli, P. M., Medeiros, A. G., Amaral, B. G. do, Barrionuevo, D. C., Araújo, J. C. D. de, Ribeiro, J. L., & Magalhães, L. P. (2019). A realistic and public dataset with rare undesirable real events in oil wells. *Journal of Petroleum Science and Engineering*, *181*, 106223. https://doi.org/10.1016/j.petrol.2019.106223

Wachowiak, M. P., Moggridge, J. J., & Wachowiak-Smolikova, R. (2019). Clustering Continuous Wavelet Transform Characteristics of Heart Rate Variability through Unsupervised Learning. *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 4584–4587. https://doi.org/10.1109/EMBC.2019.8857515

Wang, X.-B., Zhang, X., Li, Z., & Wu, J. (2019). Ensemble extreme learning machines for compound-fault diagnosis of rotating machinery. *Knowledge-Based Systems*. https://doi.org/10.1016/j.knosys.2019.105012

Wei, L., Kumar, N., Lolla, V. N., Keogh, E. J., Lonardi, S., & Ratanamahatana, C. (Ann). (2005). Assumption-Free Anomaly Detection in Time Series. *17th International Conference on Scientific and Statistical Database Management, SSDBM*, *5*, 237–242. https://pdfs.semanticscholar.org/909b/8226968d41f76cc14d0eef2d365572e7a37b.pdf?_ga=2.68813208.804195361.1594613685-826585445.1591805583

Yiakopoulos, C. T., Gryllias, K. C., & Antoniadis, I. A. (2011). Rolling element bearing fault detection in industrial environments based on a K-means clustering approach. *Expert Systems with Applications*, *38*(3), 2888–2911. https://doi.org/10.1016/j.eswa.2010.08.083

Zhu, A. Z., Yuan, L., Chaney, K., & Daniilidis, K. (2019). Unsupervised Event-Based Learning of Optical Flow, Depth, and Egomotion. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 989–997. https://doi.org/10.1109/CVPR.2019.00108

3.1.3. Manuscrito 3 - Detecting Interesting and Anomalous Patterns In Multivariate Time Series Data in an Offshore Platform Using Unsupervised Learning

O presente manuscrito, publicado na Offshore Technology Conference 2021 e anexado na biblioteca OnePetro, consistiu em fazer mais uma análise comparativa entre os modelos de aprendizagem de máquina não supervisionada para reconhecimento de padrões anômalos. O experimento utilizou a base de dados 3W. Cinco casos de estudo foram selecionados para a análise. Os dados foram coletados na linha de produção de O&G de três poços offshore de fluxo natural. Portanto, como resultado, novamente o algoritmo C-AMDATS revelou uma superioridade significativa em relação aos outros algoritmos para reconhecimento de padrões anômalos. O tempo de resposta do C-AMDATS também foi o maior de todos. Portanto, os resultados obtidos reforçam o desfecho do Manuscrito 1 e Manuscrito 2.

Este artigo foi publicado na Offshore Technology Conference 2021 (OTC 2021), realizado em Houston, Texas, EUA, e anexado na biblioteca online de literatura técnica para a indústria de exploração e produção de petróleo e gás (E&P) OnePetro (ISBN: 978-1-61399-787-1)
DOI: https://doi.org/10.4043/31297-MS

Detecting Interesting and Anomalous Patterns In Multivariate Time-Series Data in an Offshore Platform Using Unsupervised Learning

Ilan Sousa Figueirêdo; Tássio Farias Carvalho; Wenisten José Dantas Silva; Lílian Lefol Nani Guarieiro; Erick Giovani Sperandio Nascimento

Paper presented at the Offshore Technology Conference, Virtual and Houston, Texas, August 2021.
Paper Number: OTC-31297-MS
https://doi.org/10.4043/31297-MS
Published: August 09 2021

Abstract

Detection of anomalous events in practical operation of oil and gas (O&G) wells and lines can help to prevent production losses, environmental accidents, and human casualties, as well as reducing maintenance costs. Supervised machine learning models have been successful to detect, diagnose, and predict anomalous events in O&G industry. However, these models need a large quantity of annotated dataset, and labelling data in real world scenarios is typically unfeasible due to exhaustive work of experts. Therefore, once unsupervised machine learning does not require an annotated dataset, this paper proposes a comparative evaluation of unsupervised learning algorithms to assist experts for pattern recognition and anomaly detection in multivariate time-series data. So, the goal is to allow experts to analyze a small set of patterns and label them, instead of analyzing big datasets. This paper used the public 3W database of three offshore naturally flowing wells. The experiment used real data of production of O&G from underground reservoirs with the following anomalous events: (i) spurious closure of Downhole Safety Valve (DHSV) and (ii) quick restriction in Production Choke (PCK). Six unsupervised machine learning algorithms were evaluated: Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance (C-AMDATS), Luminol Bitmap, SAX-REPEAT, k-NN, Bootstrap, and Robust Random Cut Forest (RRCF). The comparison evaluation of unsupervised learning algorithms was performed using a set of metrics: accuracy, precision, recall, specificity, F1-Score, Area Under the Receiver Operating Characteristic Curve (AUC-ROC), and Area Under the Precision-Recall Curve (AUCPRC). The experiment only used the data labels for assessment purposes. The results revealed that unsupervised learning successfully detected the patterns of interest in multivariate data without prior annotation, with emphasis on the C-AMDATS algorithm. So unsupervised learning can leverage supervised models through the support given to data annotation.

1. Introduction

Detecting the early stage of anomalous events of operation of O&G wells and lines is incredibly important for improving environment and community safety. In addition, it can considerably improve the operational performance, such as by adjusting operating parameters to forestall failures or by scheduling maintenance to reduce unplanned repairs and to minimize downtime (Yintao Liu et al., 2011). However, these events are complex and hard to detect, especially in the production of O&G from underground reservoirs that have many mechanical and chemical processes that may cause serious problems, such as the total loss of the well (Matheus A. Marins et al., 2020).

Advances in data acquisition, transmission, and storage have led oil companies to develop large databases associated with their entire production and management chains. Such databases can integrate several kinds of data from all sorts of sources, as example: machinery sensors, economic series, human resources, production rates, fluids, and mechanical condition of the well. These data are useful to develop robust algorithms to prevent failures, such as: flow influx detection during drilling (Tang et al., 2019), leak detection and location in water and oil pipelines (C. Liu et al., 2019), hydrate in production line (Matheus Araújo Marins & Lima Netto, 2018), and artificial lift failure (Y Liu et al., 2010; Yintao Liu et al., 2011). The Predictive Maintenance (PdM) is a well-known method that uses historical data to draw up a maintenance schedule based on the condition of the machine's health. (Jardine et al., 2006). The method gained notoriety for supporting great proficiency in the durability of the system and reducing maintenance costs and logistical footprints (Gombé et al., 2019). Nevertheless, conventional PdM methods rely mainly on physical phenomena that generally require precise and specific knowledge about the system degeneration. In addition, they further weaken by preventing an analysis that cannot be carried out through new online measurements.

The development of prediction models based on Artificial Intelligence (AI) has been a hotspot among scientists in the new era of Industry 4.0. Through the improvement of the multi-sensor, signal transmission and computational power technologies, AI can use high-dimensional data (*e.g.*: multivariate time-series) to create robust algorithms for support decision-making. These models have advantages over physicsbased models because they can be updated and self-adapt in real time through new online signal measurements. However, AI models are strongly dependent on the availability of a big mass of data (Yu et al., 2019).

Recently, AI, machine learning, and deep learning have been published in the literature with promising results in O&G sector, such as: Marins et al., (2020) obtained an overall accuracy (ACC) above 94% by applying the well-developed Random Forest method for detecting and classifying faults in oil wells and production/service lines; Zhumekeshov and Bogdanchikov (2020) revealed a root mean square error (RMSE) of 0.53 using deep learning with Long-Short Term Memory (LSTM) layers to predict oil production; and Aung et al., (2020) provided an overview of the Artificial Neural Network (ANN) and the Support Vector Machine (SVM) analyzing geological data, price forecasting, and flow regime forecasting; and they concluded that AI-based methods increase the efficiency of the work performed both in exploration and production, making it possible to achieve better results at a lower cost. Nonetheless, these models are supervised learning and require tens or hundreds of thousands of label data, *i.e.*, in supervised learning a prior knowledge of the expected output values exists, which are expensive to obtain in an industrial environment (Wu et al., 2021).

Unsupervised machine learning algorithms contrarily to the supervised algorithms enables to create decisions from the unlabeled data. So, unsupervised learning is a promising method for situations wherein labels are nonexistent or impracticable to

develop. The main goal is to determine the underlying hidden patterns and results from the unlabeled data, *i.e.*, trying to learn the intrinsic structure of the data without explicit labels (Elavarasan et al., 2018). Therefore, instead of selecting features by a human operator, the unsupervised learning is quite intelligent and independent of specific knowledge of processing techniques and field expertise in a data-driven way. Therefore, there is a desire that unsupervised learning can solve the problem of labeling big data to leverage supervised machine learning models. (H. Liu et al., 2018). As example, Lim et al., (2020) employed Gaussian Mixture Model (GMM) to pattern recognition with purpose of identifying oil adulteration with no need of a human labeling; Alatrach et al., (2020) used Autoencoders (AE) to learn normal behavior to reconstruct the input data and fed a supervised model to predict well production event and; Zhang et al., (2020) applied a locality preserving projection based on unsupervised learning to provide a prediction production in low-permeability reservoirs.

Therefore, this paper proposes a comparison evaluation between six unsupervised machine learning algorithms for anomaly detection and pattern recognition to assist data annotation and labeling activity. The experiment used the public 3W database which consists of multivariate time-series data of O&G production from underground reservoirs. The proposed is to investigate the technological maturity of the unsupervised learning algorithms applied in the O&G industry.

2. Unsupervised Machine Learning

This paper implemented six consolidated unsupervised machine learning algorithms for evaluation comparison. In theory, the algorithms should be able to detect and isolate the anomalous pattern directly from the raw time-series data. So, the well-known unsupervised learning algorithms were: C-AMDATS, Luminol Bitmap, SAX-REPEAT, k-NN, Bootstrap and RRCF.

C-AMDATS is a clustering algorithm designed for pattern recognition or anomaly detection in multivariate time-series data. The algorithm uses the Mahalanobis distance to find hidden patterns and afterward calculate their respective anomaly scores in the time-series. Therefore, the higher the anomaly score, the more likely it is to be an anomaly pattern (Nascimento et al., 2015).

Bitmap is an available unsupervised learning algorithm in Luminol library for anomaly detection. The background of Bitmap is based on dimensionality reduction through the Symbolic Aggregate Approximation (SAX) and bitmap time-series representation to identify portions of data that can be anomaly (Wei et al., 2005).

SAX-REPEAT algorithm uses the original SAX approach extending to handle multivariate time-series data. The algorithm uses SAX individually for each dimension of the data and then combines the resulting string by applying k-means to generate a unique identifier (Mohammad & Nishida, 2014).

The k-NN method was implemented by Ramaswamy et al. (2000) to become an anomaly detection approach as unsupervised learning. The unsupervised k-NN uses the distance to k^{th} nearest neighbor as anomaly score. The anomaly score for each data point is the sum of the distance from its k nearest neighbors (Angiulli & Pizzuti, 2002). Every Single data point are ranked by its anomaly score and the highest n points of the rank are classified as anomalous.

Bootstrap is an algorithm for estimating any summary statistics, such as the time-series confidence interval for finding potential anomaly data. The algorithm uses the original dataset to generate randomly a new data (called bootstrap samples) with substitution observations. Therefore, in general, the higher the bootstrap samples, the better the accuracy estimate (Efron, 1992; Efron et al., 2015).

RRCF is an ensemble algorithm for anomalous data detection. The algorithm uses a set of random data points, cuts them to equal numbers of points and builds trees. Thereby it verifies at all trees together to detect anomaly data (Guha et al., 2016).
As the unsupervised machine learning algorithms are well-understood in the literature, we did not explain in detail their logical functioning, such as mathematical equations, hyperparameters, limitations, advantages etc. We strongly recommend reading the book chapter of Figueirêdo et al., (2020) for further information or the papers produced by the developers (reference cited in the text above).

3. The 3W Database and preparation

In mid-2017, a Brazilian oil company developed a project to encourage the creation of machine learning algorithms that automate the detection and diagnosis of anomalous events in offshore naturally flowing wells. Naturally flowing wells are reservoirs that have enough pressure to produce hydrocarbons at a commercial rate without requiring any additional energy (*e.g*: beam pumps, gas lift, and electric submersible pumps). This scenario can occur in both offshore and onshore wells. This kind of well has less machineries and consequently less instrumentation, control loops, and automation. However, it is not uncommon for a well to be operated in an intercalated fashion between an artificial lift technique and the natural method (Vargas et al., 2019).
The project selected eight specific types of anomalous events in offshore naturally flowing wells to create the 3W database with annotation data.[4] Further this data would be published for the scientific community. The data labeling was done for each anomalous event to have up to three periods: normal, faulty transient and faulty steady state. Vargas et al. (2019) reported that many hours of expert work were required to validate the historical anomalous event.
In faulty transient periods the undesirable events are still ongoing. When this period ceases, the faulty steady state period begins. In other words, the faulty transient period can be interpreted as an undesirable pre-event period.
In this paper, we merge the fault transient and faulty steady states and consider it as a failure, *i.e.*, we perform a binary classification (normal and failures). Two specifically anomalous events were chosen as experiments for the unsupervised learning algorithms: (i) spurious closure of DHSV and (ii) quick restriction in PCK.
The DHSV was installed in the production pipeline of O&G well for safety issues. The objective was to safeguard the closure of the well in emergency scenarios or physical disconnection in the event of an emergency or catastrophic failure of surface machinery. Eventually, the closure function fails in a spurious manner without any warning on the surface (*e.g.*, pressure drop in the hydraulic actuator). Therefore, with the advent of machine learning algorithms to detect automatically spurious DHSV closure, it will be possible to reopen the valve through corrective operational procedures, avoiding production losses and additional costs.
The PCK is a valve installed at the beginning of the production unit responsible for controlling the well from the surface. When this kind of valve is operated manually, undesired quick restrictions may eventually occur affecting the O&G production directly. Therefore, detecting this event automatically is very desirable to reverse the fault more quickly.
Each event in the 3W database is a time-series data composed by 8 tags acquired by 8 different sensors, chosen according to their availability and relevance to the faults at

[4] https://github.com/ricardovvargas/3w_dataset

hand. For the events that we chose, only five monitored variables of the offshore naturally flowing wells were available, as given in Table 1.

Table 1. List of available tags, including tag names, descriptions, and measuring units.

Name	Description	Unit
P-TPT	Pressure at temperature/pressure transducer (TPT)	Pa
T-TPT	Temperature at temperature/pressure transducer (TPT))	◦ C
P-MON-CKP	Pressure upstream of production choke (CKP)	Pa
T-JUS-CKP	Temperature downstream of production choke (CKP)	◦ C
P-JUS-CKGL	Pressure downstream of gas lift choke (CKGL)	Pa

Five real operational events representing different states of the well were chose, varying from normal operation (Class 0) to the faulty (spurious closure of DHSV or quick restriction in PCK - [5]Class 1). All

real data from the 3W database were extracted from the plant information (PI) system used to track the industrial processes of the Operational Unit located in the Brazilian state of Espírito Santo (UO-ES). This extraction was done without preprocessing, such as filling Not-a-Number (NaN) values, freezing variables (due to sensor or network communication issues), instances with different sizes, and outliers removed. All collected at a rate of 1 sample per second. In summary, five datasets of real cases were structured for this study:

Dataset 1. Started on 03/01/2014 at 15:17:00 and ended on 03/01/2014 at 18:28:17 (3 hours, 11 minutes and 17 seconds);
Dataset 2. Started on 02/12/2014 at 17:03:33 and ended on 12/02/2014 at 23:49:42 (6 hours, 46 minutes and 9 seconds);
Dataset 3. Started on 10/31/2017 at 18:15:09 and ended on 10/31/2017 at 19:14:54 (59 minutes and 45 seconds);
Dataset 4. Started on 03/13/2017 at 16:08:04 and ended on 03/13/2017 at 18:00:21 (1 hour, 52 minutes and 17 seconds);
Dataset 5. Started on 10/31/2017 at 20:00:59 and ended on 10/31/2017 at 20:28:56 (27 minutes and 57 seconds).

Each dataset has up to 5 monitoring variables as given in Table 1. The unsupervised machine learning algorithms should be able to detect and isolate the anomalous period to support the experts in data labeling. There is no NaN value in the monitoring variables data. However, there are few data points without labels (NaN values in the class column).

[5] In Vargas et al. (2019), the authors classified spurious closure of DHSV as class 2 and quick restriction in PCK as class 6.

For these samples, the data that is located between the faulty transient and faulty steady state periods were considered as failure and the rest as normal. Table 2 shows the statistics of each dataset and their respective balancing.

Table 2. Summary of dataset statistics.

Dataset ID	Filename	Failure Type	Data Balancing (%)	Data Points (#)	Variables (#)	Rows (#)
1	WELL-00002_20140301151700	2	13.6	57,390	5	11,478
2	WELL-00002_20140212170333	2	38.1	121,850	5	24,370
3	WELL-00004_20171031181509	2	24.5	14,344	4	3,586
4	WELL-00009_20170313160804	1	47.4	33,690	5	6,738
5	WELL-00004_20171031200059	2	49.3	6,712	4	1,678

Failure type 1 = spurious closure of Downhole Safety Valve. Failure type 2 = quick restriction in Production Choke

As shown in Table 2, there is balanced and unbalanced data. For example, Dataset 5 is almost symmetric with 49.3% of anomalous data. However, in real scenarios are more common unbalanced data, as it is expected the industrial process to operate most of the time in good condition, as Dataset 1 is. Figure 1 shows the multivariate time-series data of the five datasets. The vertical red lines indicate the anomalous period state.

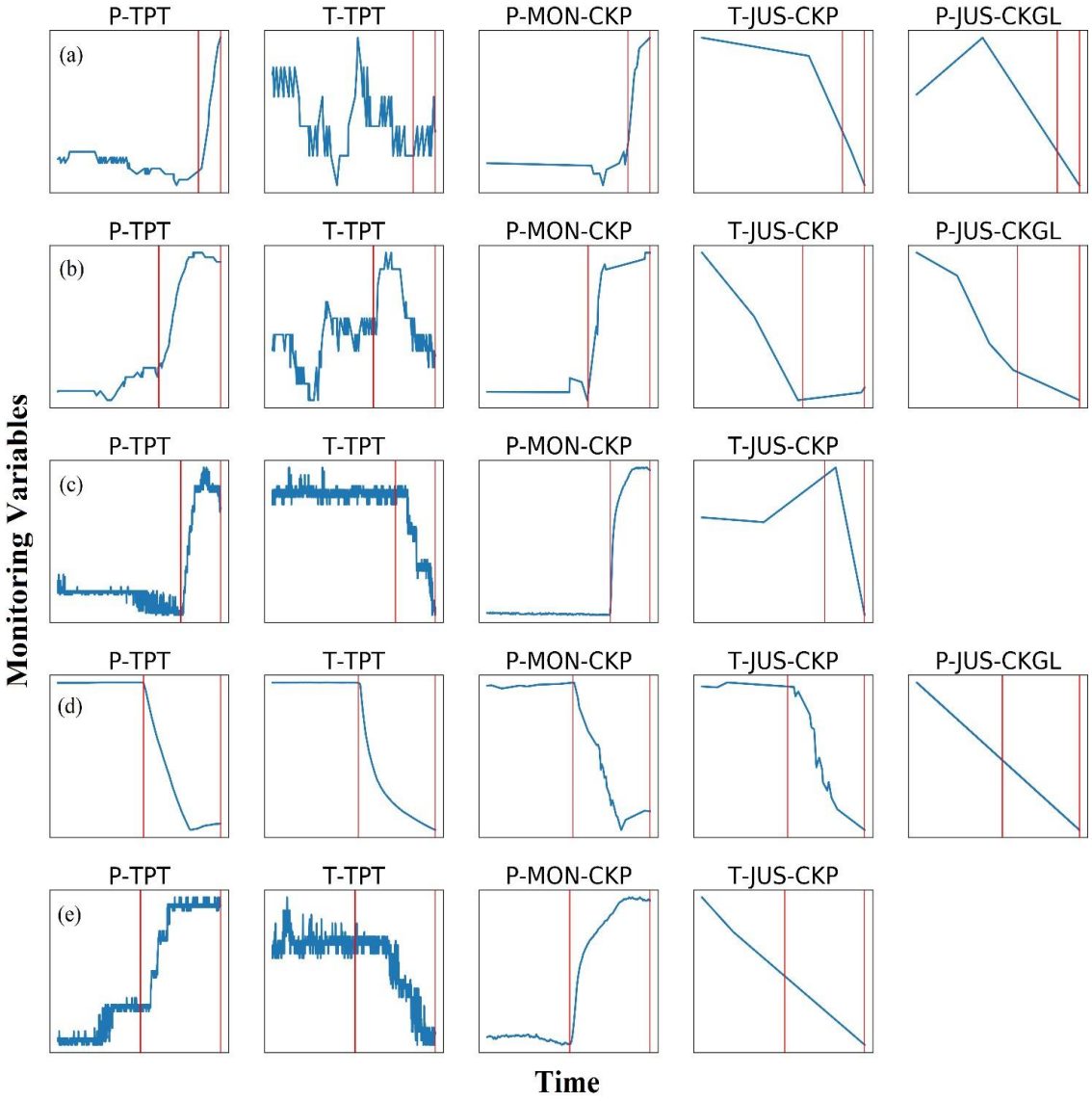


Figure 1. Multivariate time-series data with a recorded anomalous marked between the vertical red line (anomalous period state): (a) Dataset 1, (b) Dataset 2;(c) Dataset 3, (d) Dataset 4, and (e) Dataset 5.

The monitoring variable P-JUS-CKGL is not available for Dataset 3 and Dataset 5. In most of the graphs in Figure 1 there is a change in the behavior of the time-series in the first vertical red line (when failures begin). These changes presented upward spikes or presented downward spikes.

4. Parametrization Tuning

A learning model that summarizes data with a set of tuning parameters is called a parametric machine learning algorithm. The parametric algorithm may work well if the assumptions turn out to be correct, but it may perform badly if the assumptions are wrong. To find a good parametrization requires expertise in algorithm logic and some attempts at success and error. We implemented a loop in the execution of algorithms that vary the parameters of each interaction, generating different combinations of parameters each interaction. The following values are showed in Table 3:

Table 3. Parameter tunning.

Algorithm	Parameters Tunning
C-AMDATS	• Initial Cluster Size (in seconds): [60] • Clustering Factor: [0.5, 0.8, 1.0, 1.4, 1.6, 1.8, 2.0, 2.2]
Luminol Bitmap	• Detector Score: [0.1,0.5,1.0] • Future / Lag Size: [1, 10, 100, 1000] • Precision: [1, 10, 100] • Chunk Size: [1, 10, 100]
SAX-REPEAT	• Window Size: [1, 10, 100, 500, 1000] • PAA Size: [1, 2, 3, 4] • Alphabet Size: [1, 2, 3, 4]
k-NN	• K: [3, 5, 10, 50, 100, 150, 300, 500] • Contamination: [0.1, 0.2, 0.3]
Bootstrap	• Confidence Interval: [0.90, 0.95, 0.99] • Iteractions: [100, 200, 400, 500, 600, 1000, 2000, 5000]
RRCF	• Trees Number: [55,110, 220] • Tree Size: [128, 256] • Shingle Size: [4]

The metric score to choose the best model setting parameters was Area Under the Curve of Precision and Recall Curve (AUC-PRC). Next section will describe AUC-PRC in more detail. Table 4 summarizes the best parameter settings of the presented algorithms in the five case studies.

Table 4. Parameter settings of the unsupervised algorithms.

Algorithm	Parameter	Cases				
		#1	#2	#3	#4	#5
C-AMDATS	ICS (s)	60	60	60	60	60
	Cluster Factor	2.2	1.8	0.8	1.8	1.8
Luminol Bitmap	Detector Score	0.1	0.1	0.1	0.1	0.1
	Precision	10	100	10	10	10
	Window Size(1)	100	100	100	1000	100
	Chunk Size	1	100	1	100	1
SAX-REPEAT	Window Size	100	100	250	120	125
	PAA Size	2	2	2	2	2
	Alphabet Size	2	2	4	4	3
k-NN(2)	k	500	100	500	500	500
	Contamination	0.2	0.2	0.2	0.3	0.3

Algorithm	Parameter	Cases				
		#1	#2	#3	#4	#5
Bootstrap	CI	0.95	0.90	0.99	0.90	0.90
	Iteractions	100	200	400	200	100
RRCF[3]	Trees Number	55	220	220	55	55
	Tree Size	128	128	256	128	256

ICS: Initial Cluster Size, PAA: Piecewise Aggregate Approximations, k: Number of Neighbors, CI: Confidence Interval, (1) applied same value for Lag window and future window, (2) applied Euclidean distance for every cases, (3) Shingle Size = 4 for every cases.

We cannot ensure that our proposed parameterization is the best solution to our goal. Because it would be needed to test all existing possibilities of parameters, which would make this study exhaustive and with a high level of computational cost.
As the different cases analyzed are data from the same monitoring variable and type of failure (except Dataset 4), the parameters values in Table 4 did not vary much between them. This propose that for similar scenarios the parameters of the unsupervised algorithms may varies little, which is good to skip the step of finding the best parameters or to give a lead where to start, once the scenario is already known and validated.

5. Evaluation Performance

Threshold metrics are highlighted and used as a benchmark to evaluate the performance of classification algorithms. In this paper, we use a set of metrics to compare and evaluate the tuned unsupervised machine learning algorithms.

- Accuracy (ACC): considering all normal and faulty samples, the ACC can be computed as:

$$ACC = \frac{TP + TN}{n_{total}} \quad (1)$$

  where TP and TN are respectively the number of true positives and true negatives, in samples, and n_{total} is the overall number of samples;

- Precision (PR): indicates the true positive value compared to the false negative, the PR can be calculated as:

$$PR = \frac{TP}{TP + FP} \quad (2)$$

  where FP is the number of false positives;
- Recall (REC): or True Positive Ratio (TPR) indicates the proportion of anomalies that are correctly detected out of all anomalies. Normally, it is a high prioritized metric since the failure of anomaly detection (false negatives) leads to much more

deleterious results than false alarms (false positives) in industrial applications. REC can be calculated as:

$$REC = \frac{TP}{TP + FN} \quad (3)$$

where FN is the number of false negative;

- Specificity (SP): demonstrates the capacity of the model to predict the true negative over false positives, can be computes as:

$$SP = \frac{TN}{TN + FP} \quad (4)$$

- F1-Score (F1): is a harmonic average between REC and PR, and can be calculated as:

$$F1 = \frac{2 \times (PR \times REC)}{PR + REC} \quad (5)$$

- Area Under the Curve of Receiver Operating Characteristics (AUC-ROC): provides a relative tradeoff between the False Positive Rate (FPR) and the TPR. FPR is the equivalent of $1 - SP$;
- AUC-PRC: provides a relative tradeoff between the PR and the REC. It is an important metric for assessing unbalanced datasets, being a great advantage over the others metrics, because the vast majority of real data has a higher volume of normal than abnormal data.

All metrics above yield a value between 0 and 1. A perfectly inaccurate anomaly detection model has a value of 0; a model that makes random guesses has a value of 0.5; and a perfectly accurate model has a value of 1. So, higher the metric value, better anomaly detection performance.

The performance evaluation would not be properly fair as the Luminol Bitmap, k-NN, Bootstrap and RCCF algorithms make an univariable approach, different from C-AMDATS and SAX-REPEAT that have a multivariate approach. Thereby, to achieve a more proper investigation, the threshold metrics were calculated for all available variables and then computed an average evaluation, except C-AMDATS and SAX-REPEAT.

6. Results and Discussion

In this section, we present the results for the six unsupervised machine learning algorithms described in Section 2. This provides a comprehensive assessment of the unsupervised algorithms performance, under different fault-detection scenarios.

All algorithms were implemented with Python 3.6 programming language and executed on a highperformance computing named AIRIS (Artificial Intelligence RSB Integrates System) at the Supercomputing Center for Industrial Innovation at SENAI CIMATEC.

AIRIS processor model is an Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz and has 376 GB RAM memory.
After evaluating the different unsupervised algorithms for binary classification, the best model according to the seven distinct evaluation metrics are shown in Table 5, along their average and standard deviation performance.

Table 5. Unsupervised machine learning performance in four real cases (the best performance is highlighted in bold).

Algorithm	Metric	Cases (%)					Average (%)	SD (%)
		#1	#2	#3	#4	#5		
C-AMDATS	ACC	**99.5**	**97.5**	**99.9**	**100.0**	**99.4**	**99.3**	**1.0**
	PR	**100.0**	**93.7**	**100.0**	**100.0**	**100.0**	**98.7**	**2.8**
	REC	**96.7**	**100.0**	**99.7**	**100.0**	**98.8**	**99.0**	**1.4**
	F1	**98.3**	**96.8**	**99.8**	**100.0**	**99.4**	**98.9**	**1.3**
	SP	**96.7**	**100.0**	**99.7**	**100.0**	**98.8**	**99.0**	**1.4**
	AUC-ROC	**98.3**	**97.9**	**99.8**	**100.0**	**99.4**	**99.1**	**0.9**
	AUC-PRC	**97.1**	**93.7**	**99.7**	**100.0**	**99.4**	**98.0**	**2.7**
Luminol Bitmap	ACC	71.2	66.0	66.5	75.6	55.9	67.0	7.3
	PR	32.1	57.9	51.4	88.1	58.2	57.5	20.1
	REC	57.7	76.3	69.8	61.3	72.2	67.4	7.7
	F1	38.3	64.0	54.1	71.6	62.1	58.0	12.7
	SP	57.7	76.3	69.8	61.3	72.2	67.4	7.7
	AUC-ROC	65.5	68.0	67.6	74.9	56.1	66.4	6.8
	AUC-PRC	24.7	52.7	40.8	73.6	55.4	49.4	18.1
SAX-REPEAT	ACC	82.3	97.8	76.4	73.9	71.8	80.4	10.5
	PR	43.5	98.1	51.6	74.1	73.4	68.1	21.5
	REC	100.0	96.0	61.3	69.2	67.0	78.7	17.9
	F1	60.7	97.1	56.0	71.6	70.0	71.1	15.9
	SP	100.0	96.0	61.3	69.2	67.0	78.7	17.9
	AUC-ROC	89.8	97.4	71.3	73.7	71.7	80.8	12.0
	AUC-PRC	43.5	95.7	41.1	65.9	65.4	62.3	22.0
k-NN	ACC	87.9	60.7	93.1	76.6	64.2	76.5	14.2
	PR	53.6	44.6	96.2	90.0	72.8	71.4	22.3
	REC	61.3	18.9	74.9	52.2	42.9	50.0	21.0
	F1	56.1	26.4	84.1	64.7	53.9	57.1	20.9
	SP	61.3	18.9	74.9	52.2	42.9	50.0	21.0
	AUC-ROC	76.7	52.6	86.9	75.4	63.9	71.1	13.2
	AUC-PRC	47.7	45.4	78.3	74.1	62.9	61.7	14.9
Bootstrap	ACC	83.3	67.1	91.9	74.6	63.1	76.0	11.8
	PR	62.2	58.2	93.7	90.1	73.4	75.5	16.0
	REC	71.6	47.9	72.8	55.0	48.7	59.2	12.2

Algorithm	Metric	Cases (%)					Average (%)	SD (%)
		#1	#2	#3	#4	#5		
	SP	62.4	52.2	81.2	68.0	56.3	64.0	11.3
	F1	71.6	47.9	72.8	55.0	48.7	59.2	12.2
	AUC-ROC	78.4	63.4	85.5	73.6	62.9	72.8	9.7
	AUC-PRC	54.5	58.0	75.0	72.0	60.7	64.1	9.0
RRCF	ACC	79.5	60.1	74.0	54.6	51.6	64.0	12.2
	PR	17.2	41.3	41.6	61.5	54.2	43.1	16.9
	REC	12.7	10.7	12.6	13.5	12.2	12.4	1.0
	SP	14.4	16.8	19.2	21.9	19.8	18.4	2.9
	F1	12.7	10.7	12.6	13.5	12.2	12.4	1.0
	AUC-ROC	51.4	50.6	53.3	52.6	51.0	51.8	1.1
	AUC-PRC	14.1	39.3	26.6	49.4	50.1	35.9	15.5

SD: standard deviation.

From a practical perspective, analyzing the algorithm performance in the real data is far more critical than analyzing its performance in the simulated ones. Furthermore, the algorithms must avoid false positives and false negative, as each time this go on, the expert loses confidence in using the model. For these reasons, C-AMDATS revealed a high confidence model.

These results in Table 5 show that the C-AMDATS algorithm can properly detect faults without further annotation dataset. The mean metrics reached a level of 98% and 99%. Even Dataset 1, which is highly unbalanced. The other algorithms as Bootstrap and SAX-REPEAT were able to detect anomalous data for some cases study, for instance: the k-NN succeeded in isolating case 3 with 93% of ACC, and SAXREPEAT detected the anomalous signal of case 2 with 97% of ACC. However, they did not have a good average score for the 5 cases study. The remind algorithms presented a below score for all cases. Additionally, it is crucial to evaluate the processing time of each algorithm, which is directly proportional to the computational cost. Table 6 shows this evaluation comparison.

Table 6. Processing time of each algorithm (the fastest algorithm is highlighted in bold).

Algorithm	Time of each study Case (s)					Mean (s)	STD (s)
	#1	#2	#3	#4	#5		
C-AMDATS	2159.5	98.9	297.7	739.8	58.2	670.8	782.6
Bitmap	66.2	134.7	42.7	46.0	16.5	61.2	40.0
SAX-REPEAT	5.5	169.7	1.5	6.8	1.4	37.0	66.4
k-NN	4.8	5.2	1.6	3.0	1.1	3.1	1.7

Algorithm	Time of each study Case (s)					Mean (s)	STD (s)
	#1	#2	#3	#4	#5		
Bootstrap	**0.6**	**0.9**	**0.4**	**0.4**	**0.3**	**0.5**	**0.2**
RRCF	1,018.1	7,988.0	830.5	677.5	120.6	2,127.0	2,945.8

Bootstrap was the fastest algorithm with 0.5 and 0.2 seconds of mean and standard deviation, respectively. The processing time of RRCF and C-AMDATS were extremely high. C-AMDATS presented an average of 782 seconds (approximately 13 minutes) and, once C-AMDATS revealed the best performance to detect de anomalous period, as given in Table 6, it must be implemented to speed up its processing time, such as parallelization of the calculation of the inverse of the covariance matrix and use of Multiple Graphics Processing Units (GPU).

7. Conclusion and Future Works

In this paper, we have demonstrated the usefulness of unsupervised machine learning to leverage supervised models for unlabeled data scenarios. The unsupervised algorithms should be able to detect and isolate de anomalous data to aid data labeling.

Our goal was to compare the performance of the algorithms to identify automatically anomalous patterns in multivariate time-series data. The experimental results demonstrated that unsupervised algorithms as C-AMDATS had high performance (with ACC of 99%) to detect and isolate the anomalous events of production of O&G from underground reservoirs, revealing the capacity to automatically label raw data with excellent performance.

Therefore, instead the experts spend many hours labeling a big volume of data, as shown in Figure 1, they would just have to label a few recognized patterns.

In conclusion, unsupervised machine learning can leverage the development of supervised learning models for industrial applications, which is common to have unlabeled dataset. Especially in the O&G industry that have many multivariate datasets.

Future works include an extensive study and investigation of approaches of semi-supervised learning to train deep learning algorithms to predict and classify unknown data in different failure modes for PdM tasks.

Acknowledgment

The authors also thank the Supercomputing Center for Industrial Innovation (CS2I) for providing the computational resources needed to develop this work and access to the AIRIS supercomputer, and the Reference Center for Artificial Intelligence, both at SENAI CIMATEC. This work was partially supported by the Brazilian Industrial Research and Innovation Company (EMBRAPII).

References

Alatrach, Y., Mata, C., Shoeibi Omrani, P., Saputelli, L., Narayanan, R., Hamdan, M. 2020. Prediction of Well Production Event Using Machine Learning Algorithms. *Abu Dhabi International Petroleum Exhibition & Conference*, 15. https://doi.org/10.2118/202961-MS

Angiulli, F., Pizzuti, C. 2002. Fast Outlier Detection in High Dimensional Spaces. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (pp. 15–27). https://doi.org/10.1007/3-540-45681-3_2

Aung, Z., Mikhaylov, I. S., Aung, Y. T. 2020. Artificial Intelligence Methods Application in Oil Industry. *2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 563–567. https://doi.org/10.1109/EIConRus49466.2020.9039330

Efron, B. 1992. Bootstrap Methods: Another Look at the Jackknife. In *Breakthroughs in Statistics* (Vol. 7, Issue 1, pp. 569– 593). https://doi.org/10.1007/978-1-4612-4380-9_41

Efron, B., Rogosa, D., Tibshirani, R. 2015. Resampling Methods of Estimation. In J. D. Wright (Ed.), *International Encyclopedia of the Social & Behavioral Sciences* (Second Edi, pp. 492–495). Elsevier. https://doi.org/10.1016/B978-008-097086-8.42165-3

Elavarasan, D., Vincent, D. R., Sharma, V., Zomaya, A. Y., Srinivasan, K. 2018. Forecasting yield by integrating agrarian factors and machine learning models: A survey. *Computers and Electronics in Agriculture*, *155*(October), 257–282. https://doi.org/10.1016/j.compag.2018.10.024

Figueirêdo, I. S., Guarieiro, L. L. N., Nascimento, E. G. S. 2020. Multivariate Real Time Series Data Using Six Unsupervised Machine Learning Algorithms. In *Anomaly Detection - Recent Advances, Issues and Challenges [Working Title]* (Issue tourism, p. 13). IntechOpen. https://doi.org/10.5772/intechopen.94944

Gombé, B. O., Mérou, G. G., Breschi, K., Guyennet, H., Friedt, J. M., Felea, V., Medjaher, K. 2019. A SAW wireless sensor network platform for industrial predictive maintenance. *Journal of Intelligent Manufacturing*, *30*(4), 1617–1628. https://doi.org/10.1007/s10845-017-1344-0

Guha, S., Mishra, N., Roy, G., Schrijvers, O. 2016. Robust random cut forest based anomaly detection on streams. *33rd International Conference on Machine Learning, ICML 2016*, *48*, 3987–3999. http://proceedings.mlr.press/v48/guha16.pdf

Jardine, A. K. S., Lin, D., Banjevic, D. 2006. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. https://doi.org/10.1016/j.ymssp.2005.09.012

Lim, K., Pan, K., Yu, Z., Xiao, R. H. 2020. Pattern recognition based on machine learning identifies oil adulteration and edible oil mixtures. *Nature Communications*, *11*(1), 5353. https://doi.org/10.1038/s41467-020-19137-6

Liu, C., Li, Y., Xu, M. 2019. An integrated detection and location model for leakages in liquid pipelines. *Journal of Petroleum Science and Engineering*. https://doi.org/10.1016/j.petrol.2018.12.078

Liu, H., Zhou, J., Xu, Y., Zheng, Y., Peng, X., Jiang, W. 2018. Unsupervised fault diagnosis of rolling bearings using a deep neural network based on generative adversarial networks. *Neurocomputing*, *315*, 412–424. https://doi.org/10.1016/j.neucom.2018.07.034

Liu, Y, Yao, K.-T., Liu, S., Raghavendra, C., Lenz, T., Olabinjo, L., Seren, F. B., Seddighrad, S., Dinesh Babu, C. G. 2010. Failure Prediction for Rod Pump Artificial Lift Systems. *Proceedings of SPE Western Regional Meeting*, *2*, 845–852. https://doi.org/10.2523/133545-MS

Liu, Yintao, Yao, K. T., Liu, S., Raghavendra, C. S., Balogun, O., Olabinjo, L. 2011. Semi-supervised failure prediction for oil production wells. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 434–441. https://doi.org/10.1109/ICDMW.2011.151

Marins, Matheus A., Barros, B. D., Santos, I. H., Barrionuevo, D. C., Vargas, R. E. V., de M. Prego, T., de Lima, A. A., de Campos, M. L. R., da Silva, E. A. B., Netto, S. L. 2020. Fault detection and classification in oil wells and production/service lines using random forest. *Journal of Petroleum Science and Engineering*, *September*, 107879. https://doi.org/10.1016/j.petrol.2020.107879

Marins, Matheus Araújo, Lima Netto, S. 2018. Machine learning tecniques applied to hydrate failure detection on production lines. In *UFRJ Institutional Repository* (Vol. 1, Issue 1). file:///C:/Users/lanla/Google Drive/1.DOUTORADO/Journal-Congress/OTC 2021/REF/MACHINE LEARNING TECHNIQUES APPLIED TO HYDRATE FAILURE DETECTION ON PRODUCTION LINES.pdf

Mohammad, Y., Nishida, T. 2014. Robust learning from demonstrations using multidimensional SAX. *14th International Conference on Control, Automation and Systems (ICCAS)*, 64–71. https://doi.org/10.1109/ICCAS.2014.6987960

Nascimento, E. G. S., Tavares, O., De Souza, A. 2015. A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance. *ICAI'2015 - International Conference on Artificial Intelligence*, 622–628. https://www.researchgate.net/publication/282330724_A_Cluster-based_Algorithm_for_Anomaly_Detection_in_Time_Series_Using_Mahalanobis_Distance

Ramaswamy, S., Rastogi, R., Shim, K. 2000. Efficient algorithms for mining outliers from large data sets. *SIGMOD Record (ACM Special Interest Group on Management of Data)*. https://doi.org/10.1145/335191.335437

Tang, H., Zhang, S., Zhang, F., Venugopal, S. 2019. Time series data analysis for automatic flow influx detection during drilling. *Journal of Petroleum Science and Engineering*. https://doi.org/10.1016/j.petrol.2018.09.018

Vargas, R. E. V., Munaro, C. J., Ciarelli, P. M., Medeiros, A. G., Amaral, B. G. do, Barrionuevo, D. C., Araújo, J. C. D. de, Ribeiro, J. L., Magalhães, L. P. 2019. A realistic and public dataset with rare undesirable real events in oil wells. *Journal of Petroleum Science and Engineering*, *181*, 106223. https://doi.org/10.1016/j.petrol.2019.106223

Wei, L., Kumar, N., Lolla, V. N., Keogh, E. J., Lonardi, S., Ratanamahatana, C. (Ann). 2005. Assumption-Free Anomaly Detection in Time Series. *17th International Conference on Scientific and Statistical Database Management, SSDBM*, *5*, 237–242. https://pdfs.semanticscholar.org/909b/8226968d41f76cc14d0eef2d365572e7a37b.pdf?_ga=2.68813208.804195361.159 4613685-826585445.1591805583

Wu, X., Zhang, Y., Cheng, C., Peng, Z. 2021. A hybrid classification autoencoder for semi-supervised fault diagnosis in rotating machinery. *Mechanical Systems and Signal Processing*, *149*, 107327. https://doi.org/10.1016/j.ymssp.2020.107327

Yu, W., Kim, I. I. Y., Mechefske, C. 2019. Remaining useful life estimation using a bidirectional recurrent neural network based autoencoder scheme. *Mechanical Systems and Signal Processing*, *129*, 764–780. https://doi.org/https://doi.org/10.1016/j.ymssp.2019.05.005

Zhang, Y., Hu, J., Zhang, Q. 2020. Application of Locality Preserving Projection-Based Unsupervised Learning in Predicting the Oil Production for Low-Permeability Reservoirs. *SPE Journal*, *August 2019*, 1–12. https://doi.org/10.2118/201231PA

Zhumekeshov, A., Bogdanchikov, A. 2020. Forecasting Oil Production Using LSTM Networks Confined to Decline. *Natural and Technical Sciences*, *1*(52), 6–10. https://doi.org/https://doi.org/10.47344/sdubnts.v52i1.51

3.2 Aprendizagem de Máquina Semi-Supervisionada

A partir do desfecho dos estudos investigativos da aprendizagem de máquina não supervisionada para reconhecimento de padrões de anomalia ou de interesse, iniciou-se a pesquisa da aprendizagem de máquina semi-supervisionada. O objetivo foi viabilizar a aprendizagem de máquina profunda para situações que não há dados rotulados. Portanto, este subcapítulo apresenta uma novo modelo semi-supervisionado de autoaprendizagem de máquina profunda aplicada a séries temporais multivariadas.

3.2.1. Manuscrito 4 - A Novel Self Deep Learning Semi-Supervised Model to Classify Unlabeled Multivariate Time-Series from offshore naturally flowing wells of O&G Industry

O manuscrito 4, submetido para a revista IEEE Transactions on Patterns Analysis and Machine Intelligence (TPAMI), apresentou uma revisão de literatura sobre a autoaprendizagem de máquina e introduziu uma nova abordagem semi-supervisionada de autoaprendizagem de máquina profunda aplicada a séries temporais multivariadas. O objetivo foi desenvolver um modelo, sem dados rotulados, para classificar as principais falhas temporais de O&G em linhas de produção de poços surgentes offshore. Para fins de avaliação, os resultados do modelo proposto foram comparados com a aprendizagem de máquina supervisionada. Dessa forma, o modelo semi-supervisionado de autoaprendizagem de máquina profunda apresentou um desempenho equiparável à aprendizagem profunda supervisionada. Portanto, o modelo proposto apresentou recursos para alavancar a aprendizagem de máquina tradicional em um ambiente operacional industrial.

Os resultados parciais deste manuscrito também foram publicados e apresentados na GPU Technology Conference (GTC) 2022 da NVIDIA (NASCIMENTO; GUARIEIRO; FIGUEIRÊDO, 2022).

O Apêndice A desta tese apresenta uma reprodução do experimento deste manuscrito para a topologia de uma MLP.

O presente artigo foi submetido para a revista IEEE Transactions on Pattern Analysis and Machine Intelligence. (ISSN 0162-8828). Qualis A1 na área Ciência da Computação.

A Novel Self-Learning Model to Classify Unlabeled Multivariate Time-Series Applied to Fault Diagnosis

Ilan Sousa Figueirêdo, Lílian Lefol Nani Guarieiro, and Erick Giovani Sperandio Nascimento

Abstract—The traditional supervised learning paradigm requires a large amount of annotated data, which is often costly to get, becoming a bottleneck to develop deep learning solutions. This is evident in operational environments, such as in industry, as it requires specialized personnel to engage in the task of labeling data. Semi-supervised learning can overcome this bottleneck by combining unsupervised machine learning and supervised machine learning. Therefore, we designed a novel self-learning model to classify failures in multivariate time-series. We first used unsupervised learning to recognize faulty and normal patterns for pseudo-labeling a small data set. Then, secondly, we used this pseudo-labeled data to train a deep supervised learning. Finally, we designed a confidence layer to calculate a confidence value in predictions to iteratively generate pseudo-labeled data, creating a new annotated dataset. Hence, AI can be improved by training with more unlabeled data. As a result, our model can match the equivalent deep learning performance that were trained with real labels, but without the need of having any labeled data, for multivariate time series classification tasks. Therefore, this approach has the potential to underpin and improve the adoption and development of AI-based smart predictive solutions when there is a lack annotate data.

Index Terms—Deep Learning, Multivariate Time Series, Fault Diagnosis, self-learning, semi-supervised learning.

1 Introduction

In recent years, deep learning has achieved superior performance across numerous tasks and domains, such as computer vision, natural language processing (NLP), time series analysis, autonomous vehicles, predictive maintenance etc. [1], [2]. However, it requires large amounts of annotated data, which are usually costly to obtain [3]. This becomes even more challenging in industrial environments because of the complexity of allocating specialized personnel to annotate data without leaving their operational position, impacting its core business. For example, ResNet-152 achieves 77.8% Top-1 accuracy but needs 1.28 million labeled training instances. Turan and Jaschke (2021) [4] and Marins et al., (2020) [5] revealed an accuracy of 88% and 94%, respectively, and an F1-Score of 85% to classify undesirable events in the 3W database with supervised machine learning algorithms. However, the database has approximately 2,000 MTS annotated.

Many good classifier models for oil and gas (O&G) are supervised methods that require a large set of labeled data to achieve excellent performance. In many cases, in the real world, large datasets such as multivariate time-series (MTS) are available but unlabeled for a specific behavior or anomaly event. Labeled data can be expensive to acquire or at least represent a bottleneck to develop machine learning models, as resource is allocated to apply labels.

In fact, annotated datasets are rare and hard to obtain [6], so machine learning models to classify MTS data without human supervision have become highly desirable by many companies and industries [7].

Therefore, this work focuses to design a Deep Learning model without labels to diagnosis faults of MTS data. Fault diagnosis aims on detecting, isolating, and identifying faults when they occur. Diagnosis assists in discovering if a machinery is suffering from a specific irregular condition [8].

Unsupervised machine learning is a hot spot technique for clustering, pattern recognition and dimensionality reduction in unlabeled data, however it cannot diagnose faults by itself. For instance, previous works on auto-encoders [9], [10], anomaly detection [11], [12], [13], [14], and Generative Adversarial Network (GANs) [15], [16] are proposed for general unsupervised learning. Nevertheless, they are not specifically designed for classification tasks.

A promising method that may overcome this bottleneck is self-learning. The self-learning method has been introduced for computer vision [3] and language understanding [17] fields to minimize the need of large labeled data. The method is a specific type of semi-supervised learning. More precisely, commonly self-learning first designs training with few labeled data and secondly assigns pseudo-labels to new unlabeled data instances. In this technique, instead of manually labeling unlabeled data, approximate labels are given based on the labeled data. However, the pseudo-labeling method generally uses a small set of labeled data along with a large set of unlabeled data to improve the performance of

- *Ilan Sousa Figueirêdo, Lílian Lefol Nani Guareiro and Erick Giovani Sperandio Nascimento are with the University Center SENAI CIMATEC, Salvador, BA 41650-010, Brazil.*
  E-mail: erick.sperandio@surrey.ac.uk.
- *Erick Giovani Sperandio Nascimento is also with the Surrey Institute for People-Centred Artificial Intelligence, Faculty of Engineering and Physical Sciences, University of Surrey, Guildford GU2 7XH, United Kingdom.*

Manuscript received XX YY, 202X; revised AA BB, 202X.

a model. Thereby, the model can be improved due to the inclusion of unlabeled samples in the training step [18]. Still, self-learning is an understudied and long-standing area of knowledge and requires more research [19].

The self-learning is a method that can take a small set of labeled data and automatically label more data to increase the training dataset and, consequently, increase model performance. However, the challenge of our work is to develop a Deep Learning model without any label.

Our main goal is to create a deep self-learning model without the need of a single label to classify failures in MTS of O&G and, then, compare it with supervised learning approach. The greatest benefit of our model is bypassing the data annotation step. No labels would be needed to develop the model, which means that there would be no need for previous efforts by experts to label a large volume of data, which normally requires several hours of manual, tedious and specialized work. Therefore, we designed a novel self-learning model to study the ability and performance of training an artificial neural network (ANN). To validate and test our model, the public 3W dataset of an O&G offshore platform with undesirable events was used. The labels were only used for performance evaluation purposes. Specifically, the work of this paper is as follows: 1. design a self-learning model that combines unsupervised and supervised deep learning tasks; 2. train the self-learning model to classify failures of multivariate time series without labels; 3. Compare it with supervised machine learning methods. The proposed model reached a performance equivalent to supervised methods.

2 Related Research

The traditional self-learning method adds approximate labels (pseudo-labels) to unlabeled data one by one until a stopping criterion is met. Typically, it starts with a small labeled dataset and then iteratively augments or iteratively labels the initially unlabeled observations, resulting in a fully labeled dataset. The method is similar to supervised learning, however it develops machine learning algorithms to accomplish tasks where both the inputs and labels are derived from an unlabeled dataset [20]. Many self-learning approaches have been proposed in the literature, such as data augmentations [21], [22], human action recognition [23], object detection and classification [24], image colorization [25], predictive maintenance [26], and time-series classification [27]. Although these learned models may not fully match the performance of supervised learning representations, they have proved to be better than purely unsupervised representation learning approaches. However, all models proposed required at least some amount of labeled data.

The first self-learning method for time-series classification was introduced by Wei and Keogh (2006) [**?**]. It applied the one-nearest-neighbor (1NN) with Euclidean distance as a semi-supervised time-series classification algorithm. The method used a stopping criterion based on the minimum distance between classified instances, that would become known in the literature as Wei-Keogh (WK) criterion [28]. However, WK criterion proved to have the characteristic of belatedly stopping the learning. On the other hand, an early stop is also not desirable.

Therefore, an important issue in self-learning is knowing when to stop learning, and this issue led the scientific community to search for other more reliable stopping criteria, such as Ratanamahatana-Wanichsan (RW) criterion [29], Minimum description length based stopping criterion (MDL) [30], Class boundary detection using graphical analysis (CBD-GA) [31], Learning from common local clusters (LCLC) [32], [33], and Peak evaluation using perceptually important points (PE-PIP) [27]. However, all these criteria are based on minimum distance series and are not able to correct the erroneous labels applying the complete results of self-learning process.

In addition, self-learning has been a very successful method in the field of NLP, for instance Pavlinek and Podgorelec (2017) [34] developed their own semantic measure based on similarity and began to iteratively label the unlabeled observations based on distance to a labelled class. They achieved an accuracy of 86% on the "Reuters R8" dataset eight category dataset using just 0.1% of labelled data. Dorado and Ratté (2016) [35] trained a Naive Bayes (NB) and used it to augment the dataset by classifying the unlabeled data. Using the "20Newsgroups" dataset, which consists of 20 topics to be predicted, they achieved an accuracy of 80% with only 3% of labelled data. Gowda Harsha S.and Suhil (2017) [36] also used the "20Newsgroups" dataset to train a recursive K-means clustering algorithm to classify unlabeled data. They used the NB technique to reduce the number of dimensions equal to the number of classes and were able to produce a 92% accuracy with just 1% of labelled data. McEntee (2019) [20] used a dataset of 10,000 Wikipedia natural language comments.to evaluate the performance of an iterative self-learning algorithm given small set (0.2%, 1.0%, 2.0%, 4.0%, 10.0%, 20.0%, 30%, 40%) of labelled data. They made a comparative assessment between the Convolution Neural Network (CNN) and Support Vector Machine (SVM) models and achieved an Average Class Accuracy of +87% from 4% of the annotated data.

Through the related works presented, it is possible to notice that there is a gap to classify MTS with self-learning techniques for classification of multiple failures in the industry. Although the above methods revealed excellent results with a small amount of labelled data, however, in MTS field, self-learning has not yet been as much explored. Thus, more research is required (He et al., 2017) [19]. As the main technological gap, we identified that there is no methodology proposed for developing an self-learning model without annotated data for MTS classification.

3 Methodology

3.1 Multivariate Time-Series

In this paper, a time series is a set of observations sampled over time. It is adopted that the observations are made sequentially in a fixed time frequency. The time-series can be represented as a vector:

$$X = (x_1, ..., x_n) \tag{1}$$

Where n is the length of the time-series. If there are simultaneously multiple time-series acquisition to model the same event, the set of these time-series is an MTS as:

$$X = \{X^1, ..., X^m\} \quad (2)$$

Each series X^j s a univariate time-series for all $j = 1, ..., m$. An MTS can be represented as a matrix $X \epsilon R^{n \times m}$, where m is the number of univariate time-series in one MTS. The rows of this matrix are called observations. The columns represent a univariate time-series. These columns are called features of the MTS [27].

3.2 Case Study - The 3W Database

In 2019, Petróleo Brasileiro S.A., also known as Petrobras, published a database of O&G production from oil wells offshore with natural flow. The database 3W came from a project entitled "Monitoring of Specialized Alarms" that began in 2017. The main idea was to upgrade the current system, based on parametric univariate alarms, to a machine learning system, based on MTS. [37].

O&G losses are highly undesirable, not only for economic reasons, but also for environmental and health disasters. Marin, et al., (2020) [5] shows the relative cumulative volume loss and number of failures between 2014 and 2017 of Petrobras OU-Rio, which comprises 298 production/injection wells. The significant amount of loss of 23.8% was caused by problems related to the well (or reservoir) and the most significant cause of failure with 37.8% was hydrate in the injection or production line.

The database was labeled with eight main specific types of anomalous events between 2012 and 2018. Vargas et al. (2019) [37] reported that many hours of expert work were required to validate the historical anomalous event. There are up to three periods of labels for each anomalous event: (i) normal, (ii) faulty transient and (iii) faulty steady state. Table 1 describes the concept of each label.

As our goal is to build a model capable of anticipating a fault to the furthest extent, in order to give as much time as possible for the operator to intervene and minimize the losses. However, with the advantage of not needing any labeled data. Therefore, we merged the labels of the faulty transient and faulty steady states to become a unique class, namely failure.

As a proof of concept, two specifically anomalous events were chosen as experiments: (i) quick restriction in Production Choke (QR-PCK) and (ii) spurious closure of Downhole Safety Valve (SC-DHSV). Future investigations with more data should be carried out in order to validate our model for other scenarios.

The Production Choke (PCK) is a valve installed at the beginning of the production unit, responsible for controlling the well from the surface. Undesired quick restrictions may eventually occur affecting the O&G production directly. The Downhole Safety Valve (DHSV) is installed in the production pipeline of O&G well for safeguard the closure of the well in emergency scenarios or physical disconnection in the event of an emergency or catastrophic failure of surface machinery. Eventually, the closure function fails in a spurious manner without any warning on the surface. Therefore, with the advent of machine learning to detect automatically QR-PCK and SC-DHSV, it will be possible to quickly reverse the faults, avoiding production losses and additional costs.

The database presents three types of data: (i) real, (ii) hand-drawn and (iii) simulation. For a more realistic scenario, we ignored the hand-drawn and simulation data.

All real data was extracted from the plant information (PI) system used to track the industrial processes of the Operational Unit located in the Brazilian state of Espírito Santo (UO-ES). This extraction was done without preprocessing, such as filling Not-a-Number (NaN) values, freezing variables (because of sensor or network communication issues), instances with different sizes, and outliers removed. All collected at a rate of 1 sample per second.

Each MTS sample has up to 8 monitoring variables, but we only use tags that exist among all real sets. Therefore, two tags ($m = 2$) were used in this work, as shown in Table 2.

The Temperature and Pressure Transducer (TPT) is a device installed in the subsea Christmas tree, so once PCK is installed at the surface, the difference between P-TPT and P-MON-CKP can be interpreted as the pressure differential between the surface and the subsea.

Ten real operational events representing different states of the well were chosen, varying from normal operation (class 0) to the anomalous operations: (i) QR-PCK (or class 1) and (ii) SC-DHSV (or class 2). In summary, ten samples of real cases were structured for this study.

There is no missing or invalid value (NaN – Not a Number) in the data. However, there are few data points without labels. In this case, the data point that is located between the faulty transient and faulty steady state periods were considered as failure and the rest as normal. These labels were exclusively used for performance evaluation purposes of the proposed model.

The dataset is highly unbalanced - the amount of data points per class can be seen in the appendix by the pie chart. This is because industrial assets are expected to operate most of the time under normal conditions. Hence, in real scenarios, unbalanced data is more frequent than balanced data. It is worth noting that the samples in the 3W database are a clipping from a long history of data. Therefore, in scenarios where historical data is fully replicated, data imbalance is likely to be more expressive.

3.3 Self-Learning Model for MTS

In this study, we validated the Self-Learning Model for MTS (SL4M) with real data from O&G industry, however, the model was not built with purpose only for O&G field. Thus, the method can be applicable to any MTS dataset for multiclass classification.

The SL4M is capable of learning from unlabeled data. In summary, it is an iterative algorithm that has four phases. The diagram of the proposed model is represented in Figure 1, which shows the data flow from its raw version to the model classification output, passing through the (i) randomly split unlabeled raw data, (ii) pseudo-labeling by unsupervised machine learning, (iii) feature selection and extraction, (iv) and self-learning modeling stages.

In the first phase, it is necessary to separate from the entire unlabeled dataset D_u a small set of data D_u^0. We randomly choose one or two samples of MTS with undesirable events. The remaining unlabeled data will be used later in the third phase.

TABLE 1
Labels description of data.

Labels in 3W	Description	Labels in this paper
Normal	Characterized by operating under expected or projected conditions	Normal
Faulty transient	Undesirable events are still ongoing even in a very early stage. The faulty transient period can be interpreted as an undesirable pre-event period	Failure
Faulty steady	When the faulty transient period ceases, the faulty steady state period begins. The fault steady is a state of inability to perform a normal function	Failure

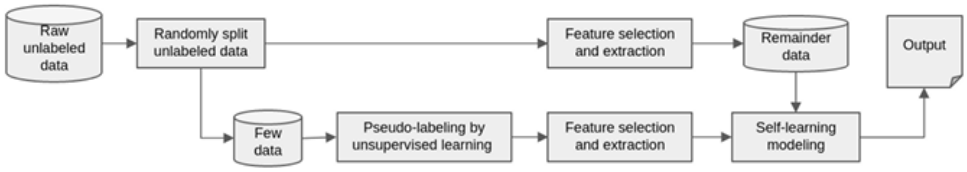


Fig. 1. Block diagram of the Self Learning Model for MTS.

TABLE 2
List of available tags, including tag names, descriptions, and measuring units

Name	Description	Unit
P-TPT	Pressure at temperature and pressure transducer (TPT)	Pa
P-MON-PCK	Pressure upstream of production choke (PCK)	Pa

In the second phase, unsupervised machine learning is applied to the small set D_u^0 for anomalous pattern recognition. It worked as a binary classification task, which classified anomaly and normal classes. Thus instead of manually labeling unlabeled data (as traditional self-learning approaches), we used unsupervised machine learning for pseudo-labeling. Then, we have pseudo-labels D_{pseu}^0 as the output of this phase.

In the third phase, a feature selection and extraction is performed. The tags (or time-series variables) are selected according to their availability among the entire dataset, *i.e.*, all tags must exist among the samples. In addition, frozen time-series variables are discarded because they do not manifest the patterns associated with the undesirable events, which indeed impose additional difficulties to the algorithms learning. The time-series is considered frozen when all observations of a time-series have any single float or integer value. A frozen variable does not always represent a problem, but this characteristic is a symptom of sensor, system configuration, or network communication issues. [37]. Therefore, after these analyses, two tags were selected (Table 2). Then, nine different statistical features are extracted for each tag through a sliding window. Feature extraction allows the reduction of the dimensionality of the data and consequently helps the model for the classification task [38].

In the fourth phase, a supervised deep learning model is trained using pseudo-labels set, D_{pseu}^0 given by the unsupervised algorithm. Following this, the model predicts newer pseudo-label from the remaining unlabeled set D_u^1. The most confident predictions are moved from the unlabeled set to the training set, *i.e.*, so more pseudo-labels are given to increase the training set. This iterative pseudo-labeling will continue until a stopping criterion is met. The stopping criterion prevents the model from pseudo-labeling any more data when the confidence drops too low. Our self-learning approach was implemented on real MTS data.

3.3.1 *Unsupervised Machine Learning*

We used a pre-existing unsupervised machine learning algorithm known as Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance (C-AMDATS) [39]. The C-AMDATS was chosen because of its high performance for recognizing patterns of interest or anomalies in MTS from different fields of science, which has been presented [11], [12], [13], [14].

The C-AMDATS is a clustering algorithm designed for anomalous pattern recognition in raw MTS data. The algorithm uses the Mahalanobis distance to find hidden patterns and afterward calculate their respective anomaly scores in the MTS. Therefore, the higher the anomaly score, the more likely it is to be an anomaly pattern. However, the algorithm has two hyperparameters that users must manipulate, which highly requires empirical knowledge about the physical phenomenon of the data: (i) Initial Cluster Size (ICS) and (ii) Clustering Factor (CF).

These hyperparameters must be correctly defined correctly so the algorithm is able to isolate the anomalous pattern. Hence, this paper also gives a contribution to C-AMDATS to run more independently of human supervision. We developed an analytical equation to automate the CF.

So, initially, the data must be scaled between [0,1] using the minimum and maximum values of each time-series, since it is well understood that it is meaningless to compare time-series with different offsets and amplitudes [40]. Secondly, the kurtosis for each scaled time-series is calculated. Thirdly and finally, the clustering factor will be the ratio

between the mean (μ) and standard deviation (σ) for the scaled time-series with the lowest kurtosis. The equation is formally presented in the appendix through the Algorithm 1.

The ICS hyperparameter continues to require empirical knowledge of the data to be established. In this work, we set it for one minute, *i.e.* sixty time steps for a one-second sampling frequency. One minute-sized clusters appear reasonable sensible for the physical phenomenon inherent in the data. In view of the nature of the event, so if the process fails, one minute of data should be enough for the algorithm to detect it. The automation of this hyperparameter requires further investigation and it is an open point to be improved in the future of this paper. However, the ICS did not appear to be a sensitive hyperparameter.

Since C-AMDATS is designed to run on raw MTS data, all tags available for the particular sample (chosen randomly in the split phase) are input data in the algorithm.

As previously described, the default output of C-AMDATS is a ranking of the probability of the recognized patterns being anomalies. Therefore, we consider the pattern with the lowest anomaly rate as a normal pattern and the remainder patterns as anomalous patterns. The Figure 2 shows the workflow of the unsupervised learning phase of our approach.

The C-AMDATS can recognize multiple patterns in MTS, whose number may vary depending only on the ICS hyperparameter, which is attached to the particular physical behavior of the MTS (input data). The anomaly score is calculated as the ratio between the length of the MTS and the length of each pattern (In Nascimento et al. (2015) [39], the authors referred to length as size). The pattern with the lowest anomaly score is assumed as normal (or negative) class, and the remaining data points are assumed as anomaly (or positive) class. Each MTS observation is given a pseudo-label as follows:

$$D_{pseu} = \{(X_1^m, \hat{y_1}), (X_2^m, \hat{y_2}), ..., (X_n^m, \hat{y_n})\} \quad (3)$$

Where $\hat{y}$ is the pseudo-label, it can be zero for negative and one for positive.

3.3.2 Feature Selection and Extraction

A feature selection and feature extraction are performed as a data pre-processing step to improve the training of the supervised machine learning model, after the unsupervised learning step. The selection of attributes consisted of identifying the available tags among the samples. Data from tags, or sensors, that were not present in all samples, were discarded.

Feature extraction is a trivially known technique for highlighting relevant data to aid models better learning patterns for classification tasks. For the SL4M, we process the raw input data from the available tag, where m_f features were extracted from each N-samples from the data window of each tag. Given an N-samples tag window $X = [x_1, x_2, ..., x_N]^T$. The $m_f = 9$ extracted features are proposed in this paper. The statistical measurement employed was based on Marins et al., (2020) [5]. However, aiming at a more general approach, other studies may choose other feature engineering techniques as data pre-processing.

Nine statistical features were given by the mean, standard deviation, minimum, maximum, median, first quartile, third quartiles, amplitude, and square amplitude values of the data window [41]. The standard deviation provides information on how the data is spread around its mean, minimum and maximum values provide the range of the data distribution within a given window, the median is the center of the distribution, the first and third quartiles reveal how the data is spread within its extremes, amplitude is the difference between the maximum and minimum values, and the square amplitude is the square of the amplitude.

In summary, the input data for the feature extraction can be represent as a matrix of the MTS $X \epsilon R^{n \times m}$,where n is the length of the MTS and m is the number of input tags, and the output data can be represent as a matrix $X \epsilon R^{w \times m_f}$, where $m_f = 9$ is the number of features extracted from each data window and w is the summation number of N-samples data window, as follows in Equation 4:

$$w = \left| \frac{n - N}{s} \right| \quad (4)$$

where s denotes the shift in samples between two consecutive windows. A fixed sliding Window Step of $s = 6$ samples and Window Size of $N = 60$ samples (minute-sized as ICS) was applied.

In order to improve the model capacity to classify the failures, a sliding window of 10 data points associating pseudo-label is used to reshape the entire dataset. This reshape is known as look back technique. The look back defines how many previous timesteps are used to predict the subsequent timestep. A look back of 10 was applied as an input vector for the SL4M. After performing feature extraction and feature selection, the dataset had a shape of (132224, 10, 18). The split of training and testing data is random, so the proportion may vary because the samples have different sizes.

3.3.3 Self-Learning Modeling

Multi-iterative training process is performed using ANN topologies. Initially, supervised deep learning is trained with a small pseudo-labeled dataset D^0_{pseu} (given by the unsupervised machine learning), i.e. $D^1_{train} = D^0_{pseu}$, where the superscript ^{0} denotes the current phase of the multi-iterative process, composed of $T = \{0, 1, ..., t\}$ steps, where $T = 0$ means unsupervised phase and $T > 0$ means supervised phase. D_{train} denotes the pseudo-labeled examples in the training set with size of $|L|$:

$$D^1_{train} = \left\{ \left(X_1^{mf}, \hat{y}_1 \right), ..., \left(X_L^{mf}, \hat{y}_L \right) \right\} \quad (5)$$

And D_u denotes all the unlabeled samples with a size of $|U|$

$$D_u = \left\{ X_{L+1}^{mf}, X_{L+2}^{mf}, ..., X_{L+U}^{mf} \right\} \quad (6)$$

After the first training, the supervised deep learning is employed to predict the confidence value δ in newer unlabeled data D^1_u where δ is the predicted probability

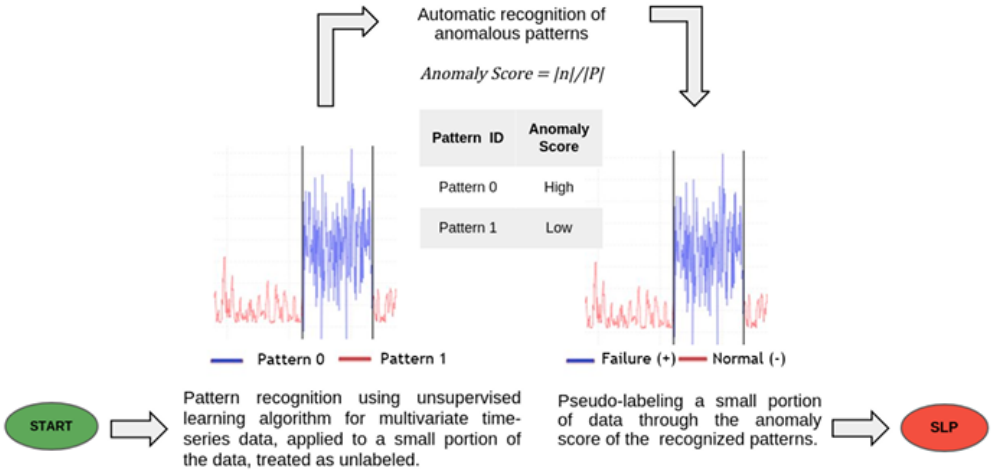


Fig. 2. The workflow of the self-learning modeling representing the unsupervised learning phase. Where n denotes multivariate time-series length, P pattern length and SLP supervised learning phase.

belonging to each class. Then, the output can be expressed as:

$$D'_u = \left\{ \left(X^{mf}_{L+1}, \hat{y}_{L+1} \right), ..., \left(X^{mf}_{L+u}, \hat{y}_{L+u} \right) \right\} \quad (7)$$

Then, if their confidence values δ are higher than the threshold δ_0, our proposed confidence-function layer filters out the confident pseudo-labels, so the confident pseudo-label set of D'_u with of $|P_1|$ can be expressed as:

$$D^1_{pseu} = \left\{ \left(X^{mf}_{L+i}, \hat{y}_{L+i} \right), ..., \left(X^{mf}_{L+P_1}, \hat{y}_{L+P_1} \right) \right\} \quad (8)$$

Afterwards, more confident pseudo-label set $D^1_p seu$ can be merged to generate a new training set D^2_{train}:

$$D^2_{train} = \left\{ D^1_{train} \cup D^1_{pseu} \right\} \quad (9)$$

Therefore, it is possible to retrain the supervised deep learning to generate new confident pseudo-labels set D^2_{pseu} with size of $|P_2|$ and a new training set, as follows:

$$D^3_{train} = \left\{ D^2_{train} \cup D^2_{pseu} \right\} \quad (10)$$

The SL4M will repeat this until it meets the following stopping criterion:

$$D^T_{pseu} = D^{T+1}_{pseu} \quad (11)$$

Once the above criterion is met, it means that the algorithm has converged to the ideal solution, which can happen either when there are no more samples to be labeled, or when the remaining unlabeled samples can not be labeled anymore. The Figure 3 shows the workflow of the supervised learning phase of our approach.

Afterwards, the model reaches the stopping criterion, if there is new data with other types of faults (or classes), it is possible to transfer the knowledge of the model to include a new class, when the whole multi-iterative process restarts from the transfer learning phase, as depicted in Figure 3.

The pseudo-code of the SL4M is presented in the Appendix B through the Algorithm 2.

3.4 Artificial Neural Network

The SL4M was developed based on two architectures of ANN: (i) 1D-CNN and (ii) hybrid 1DCNN-LSTM. The 1D-CNN denotes Convolutional Neural Network and the LSTM means Long Short-term Memory. However, any machine mearning architecture can be implemented in the proposed model.

3.4.1 Convolutional Neural Network (CNN)

The CNN is a special type of Feed-Forward Multi-Layer Neural Network with deep learning capability. The convolution layers have the capacity to create a feature map from the input set through filters with low dimensions, aiming to learn the main features from the data to perform a specific task. So, the greater the number of filters, more features can be extracted. However, there is a considerable increase in memory and processing consumption by increasing the number of filters [42]. The following equations describe mathematically the convolution layer:

$$G[m,n] = (f * k) = \sum_j \sum_i k[j,i] f[m-j, n-i] \quad (12)$$

$$C^l = a^l \left(V^{[l]} \right) \quad (13)$$

$$V^l = K^l \bullet C^{[l-1]} + b^l \quad (14)$$

Where f is the input, G is the feature map, k, n and m represent the kernel, the rows and columns of the result matrix, respectively. The indices j and i are related to the kernel, C is the result of the convolution, V is the

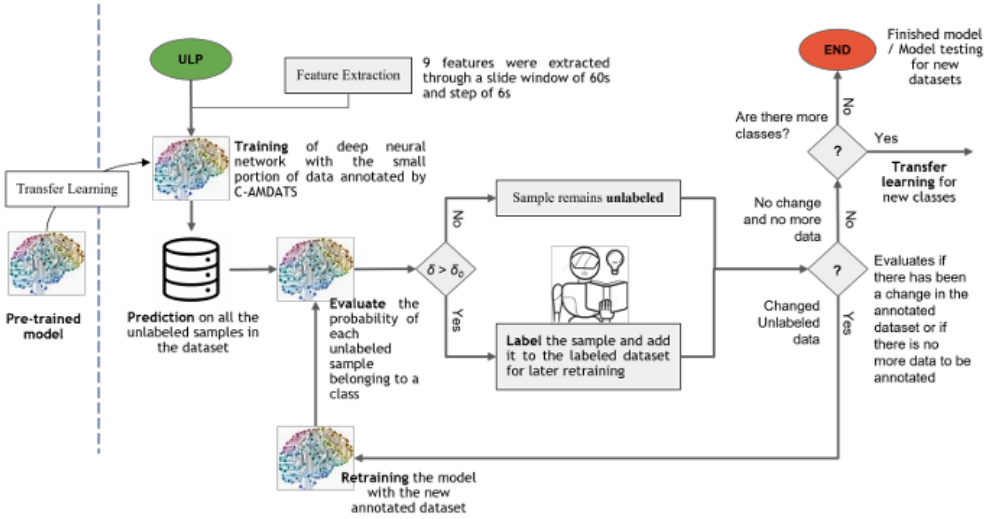


Fig. 3. The workflow of the self-learning modeling representing the multi-iterative supervised learning phase. Where ULP denotes unsupervised learning phase.

intermediate value, l is the layer index, K is the tensor that has filters or kernels, a is the corresponding activation function, and b is the bias. In addition, a Pooling layer can be employed to reduce dimensionality of the size of the output of the convolution step, e.g. extracting the average value (AvgPooling) or the maximum value (Maxpooling) from the learnt and extracted kernels/filters within a fixed-size window, therefore decreasing the required processing power for the network training.

Hybrid models consist of benefiting from the main functionalities of the baseline methods, creating a more robust model to perform more complex tasks. In this sense, CNN-LSTM method can use the CNN advantage, extracting the most important multidimensional attributes from data, resizing them, and sending it as input to the LSTM layer that can still extract more temporal attributes. Therefore, a more reliable result is expected using a combination of both methods.

3.4.2 Long Short-Term Memory (LSTM)

The LSTM layer is a type of Recurrent Neural Network (RNN) and it differs from CNN mainly for the capability to extract the temporal dependencies. The LSTM selectively forgets part of the historical information through three gates (input gate, forget gate and output gate), adds part of the current input information, and integrates it into the current state to generate the output state. [43]. The following equations depict the mathematical procedure of an LSTM cell:

$$f_t = \sigma\left(W_f\left[h_{(t-1)}, x_t\right] + b_f\right) \tag{15}$$

$$i_t = \sigma\left(W_f\left[h_{(t-1)}, x_t\right] + b_i\right) \tag{16}$$

$$C_{S_t} = S\left(W_C\left[h_{(t-1)}, x_t\right] + b_C\right) \tag{17}$$

$$C_t = f_t C_{t-1} + i_t C_S \tag{18}$$

$$o_t = \sigma\left(W_o\left[h_{(t-1)}, x_t\right] + b_o\right) \tag{19}$$

Where i_t is the input gate, f_t is the forget gate, C_t and CS_t are the respective cell state and the candidate for the cell state at a given timestep t, respectively, σ is the sigmoid functions, o_t is the output gate at t, W_x, is the weight matrix of the x neurons, S is the a hyperbolic tangent function, x_t is the input at t, h_t is the cell output at t, and b_x is the bias matrix corresponding to x.

3.4.3 Softmax layer

The Softmax layer calculates the probability distribution of the event over k events. The label prediction layer transforms the output, from the dense layer, to a sequence of vectors with the probability of each label for each corresponding element. In this paper, the Softmax layer is used for multi-classification. The main advantage of using Softmax is the output probability range of 0 to 1 and the sum of all probabilities equals one. For $D_{train} = \left\{\left(X_1^{mf}, \hat{y}_1\right), \left(X_2^{mf}, \hat{y}_2\right), ..., \left(\left(X_L^{mf}, \hat{y}_L\right)\right\}$ with k classification $y^{(i)} \epsilon\, (1, 2, 3, ...k)$ Given a network input parameter encoded as one-hot matrix θ:

$$\theta = W_0 X_0 +, ..., + W_k X_k = \sum_{i=0}^{k} W_i X_i = W^T X \tag{20}$$

Where W is the weight from the ANN and $W_0 X_0$ is the bias that needs to e added on every iteration. Bias can be thought of as analogous to the role of a constant in a linear function, whereby the line is effectively transposed by the constant value. So, for every input X_i^{mf}:

$$p\left(y=j|X_i^{mf}\right)=\begin{bmatrix} p\left(\hat{y}^{(i)}=1|X_i^{mf};\theta\right) \\ p\left(\hat{y}^{(i)}=2|X_i^{mf};\theta\right) \\ \cdot \\ \cdot \\ \cdot \\ p\left(\hat{y}^{(i)}=k|X_i^{mf};\theta\right) \end{bmatrix} \quad (21)$$

$$Eq.(21)=\frac{1}{\sum_{j=1}^{k} e^{\theta_j^T \bullet X_i^{mf}}}\begin{bmatrix} e^{\theta_1^T \bullet X_i^{mf}} \\ e^{\theta_1^T \bullet X_i^{mf}} \\ \cdot \\ \cdot \\ \cdot \\ e^{\theta_k^T \bullet X_i^{mf}} \end{bmatrix} \quad (22)$$

The layer can predict whether the trained set of features X_i^{mf} belongs to class j. Where e denotes the exponential function whose base is the Euler number. In Equations 21 and 22, we compute the exponential of the input parameter and the sum of exponential parameters of all existing values in the inputs. Therefore, our output for the Softmax function is the ratio of the exponential of the parameter and the sum of exponential parameter.

3.4.4 *Confidence-function layer*

The confidence-function layer is designed to calculate the confidence value δ of each element in D_u^t, and generate the pseudo-labels, for every input X_i^{mf}, as described in Equation 23:

$$\delta\left(X_i^{mf}\right)=max\left(0,p\left(y=j|X_i^{mf}\right)\right) \quad (23)$$

Suppose δ_0 is the threshold to filter the unconfident pseudo-labels in D'_u, so the confident pseudo-label of an element X_i^{mf} in D'_u c, if $\delta\left(X_i^{mf}\right) > \delta_0$, then:

$$\hat{y}=\left\{\begin{matrix} 1, if\ j=argmax\ p\left(y=j|X_{mf}^{i}\right) \\ 0, otherwise \end{matrix}\right\} \quad (24)$$

Where δ_0 is a hyperparameter that can be chosen by the user. For this study, after several experiments, we used $\delta_0 = 0.75$, which means that a sample would only be labeled if the maximum argument of the predicted classes is equal to or greater than 0.75.

Consequently, we have the whole confident pseudo-label set of D'_u, D_{pseu}^T with size of $|P_T|$. Then, the new train set takes the following form:

$$\mathrm{D}_{train}^{T+1}=\left|\mathrm{D}_{train}^{T}\cup\mathrm{D}_{pseu}^{T}\right| \quad (25)$$

$$Eq.(25)=\left\{\left(\mathrm{X}_1^{mf},\hat{y}_1\right),...,\left(\mathrm{X}_L^{mf},\hat{y}_L\right),...\left(\mathrm{X}_{L+P_T}^{mf},\hat{y}_{L+P_T}\right)\right\} \quad (26)$$

As a result, the deep learning model is retrained until it reaches the stopping criteria.

3.5 Model Setup

The proposed 1D-CNN model is composed of two convolution layers with Rectified Linear Unit (ReLU) activation function; one pooling layer with MaxPooling activation function, to emphasize the fundamental features of the data; a flatten layer to convert multi-dimensional data into one-dimensional data; a dropout layer to reduce overfitting, and two densely connected layers with Hyperbolic Tangent (Tanh) and softmax function for each layer. The last layer must have the number of neurons equivalent to the number of classes k in training. The 1D-CNN Neural Network Architecture is depicted in appendix. The proposed hybrid 1DCNN-LSTM model is composed of one convolution layer with ReLu activation function; one pooling layer with MaxPooling activation function; a LSTM layer with Tanh activation function; a time distributed dense layer with Tanh activation function, which is used as the wrapper to extract the feature from every temporal slice independently; a flatten layer; a dropout layer; and two densely connected layer with Tanh and softmax function for each layer. The last layer must have the number of neurons equivalent to the number of classes k in training. The hybrid 1DCNN-LSTM Neural Network Architecture is depicted in appendix.

The time distributed layer is known as a layer wrapper, which means all the parameters of layers for each input fragment can be shared. In other words, compared to the normal dense, time distributed dense can share the same weights and biases of each input fragment (*i.e.*, perform the same operation on each timestep) [44].

4 Evaluation Metrics

Threshold metrics are highlighted and used as a benchmark to evaluate the performance of classification algorithms. Therefore, a set of metrics were used to compare and evaluate the SL4M. Accuracy (ACC) that considers all normal and faulty samples. Precision (PR) evaluates the performance considering the true positive value compared to the false negative. Recall (REC) or True Positive Ratio (TPR) indicates the proportion of anomalous data that are rightly detected out of all anomalies. Usually, in industrial applications, REC is a highlighted metric ever since the false negatives guide to much more harmful results than false alarms or false positives. Specificity (SP) estimates the capacity of the algorithm to predict the true negative over false positives. F1-score (F1) is used as the metric to decide the best hyper-parameters and classifiers, with the balanced accuracy also reported for additional information. The F1-score is the harmonic mean of precision and recall. Area Under the Receiver Operating Characteristic Curve (AUC-ROC) grants a relative tradeoff between the False Positive Rate (FPR) and the TPR. FPR is the equivalent of 1 SP. Area Under the Precision-Recall Curve (AUC-PRC) grants a relative trade-off between the PR and the REC. It is an essential metric for evaluating unbalanced datasets, having a fantastic benefit over other metrics, because most real data have a higher volume of normal data in relation to abnormal data.

All metrics above yield a value between 0 and 1. A rubbish anomaly detection model would present a value of 0; a model that performs random predictions would present a value of 0.5; and a perfect model would have a value of

1. Consequently, the higher the metric value, the better its performance. However, the SL4M results are presented in percentages, that is, values [0, 1] are multiplied by 100.

5 Results and Discussion

In this section, we evaluate the performance of the SL4M (presented in Section 3) comparing it with a Supervised Deep Learning model, with the same architecture and data.

The proposed model was primarily developed to learn to classify the QR-PCK and normal classes. After that, we performed a Transfer Learning to teach the SC-DHSV fault class to the model through new data.

The supervised model proved to be slightly superior to the self-learning model. However, the difference is considerably small, especially for the 1DCNN-LSTM model, which did not show a negative difference above 3% in the average column of Table 3. The 1D-CNN model did not show a negative difference greater than 5.3% in the average column. It is also possible to notice that, despite the DHSV class being classified after the Transfer Learning step, the overall classification performance of the model did not drop, presenting in some cases better metrics than the normal and PCK classes. The experiments proved that this algorithm converges to the optimal solution with few $|T|$ iterations.

In descending order, the major limitation of the proposed model was revealed by the following indicators: (i) Recall for the PCK failure class, with a difference of 9.5% for the 1DCNN-LSTM model and 14% for the 1D-CNN model; (ii) Precision for the Normal class, with a difference of 6.8% for the 1DCNN-LSTM model and 9.6% for the 1D-CNN model; and (iii) Specificity for the Normal class, with a difference of 5.8% for the 1DCNN-LSTM model and 8.6% for the 1D-CNN model. The appendix presents a table with the performance difference between supervised learning and self-learning models, where the values of the supervised learning metrics (ACC, PR, REC...) are subtracted by the values of the self-learning metrics.

Both self-learning models showed slight superiority in the following indicators: (i) Precision for the PCK failure class, with +2.5% and +5.2%; (ii) Specificity for the PCK fault class, +1.4% and +2.7%; and (iii) in the Recall for the normal class, +0.8% and +1.8% for the 1DCNN-LSTM and 1D-CNN models, respectively. The 1DCNN-LSTM self-learning model still managed to outperform the supervised model in the DHSV failing class in the indicators of Precision (+0.6%) and Specificity (+0.2%).

This result can be interpreted as a great opportunity to develop deep learning models without any labels provided, once the performance of self-learning showed to be quite equivalent to a supervised model trained with hundreds of thousands of labels.

5.0.1 Evaluating Computational Cost

Additionally, it is crucial to evaluate the processing time for each algorithm to be built/trained and to perform predictions, which are directly proportional to the computational cost. We used NVIDIA GPU Tesla V100-SXM3 to accelerate and develop the self-learning deep learning semi-supervised model. Table 4 shows the processing time comparison between the models.

As previously described, our proposed model was first developed for class 1 (QR-PCK) and secondly, with the employment of transfer learning, to class 2 (SC-DHSV). Furthermore, unlike the supervised model that can be developed in a single training for both classes, our model is incrementally developed over $|T|$ iterations.

In this experiment, our approach presented a processing time ranging from 814% to 1,045% greater than the supervised learning to be built and trained. However, when compared to the time it would take for human experts to manually annotate a such a large set of multivariate data to be latter used for training a supervised model, the processing time of our approach would be negligible, showing its great potential to be used in real world problems for MTS classification in the industry.

Notwithstanding, it is worth mentioning that the main bottleneck in our approach is the unsupervised machine learning stage which can take between 75% to 85% of the development time. Previous works [11], [12], [13], [14] revealed that C-AMDATS has great potential to be computationally optimized, *e.g.* by parallelizing its code for calculation of the inverse of the covariance matrix with the use of multiple GPUs to speed up the processing time. Therefore, we believe that our approach can be improved in future works.

6 Conclusion and Future Works

In this paper, we designed a novel self-deep learning semi-supervised to detect failures of O&G production from unlabeled MTS.

We introduced a confidence network layer that automatically pseudo-labels unlabeled dataset to help the model grow the training set and performance by itself.

We used the 3W database from Petrobras company to demonstrate the equivalence performance of our model over supervised deep learning models. The same conditions of data pre-processing, architecture and hyperparameters of the ANN and data were replicated for both models (semi-supervised and supervised). We used the labels only for evaluation purposes, and the test data was never presented to the models during the training stage.

Transfer learning was applied to increment more classes in the learning of the semi-supervised model. We use three classes to perform a multi classification task.

The performance evaluation of the semi-supervised deep learning self-learning model reached a result close to a supervised model trained with hundreds of annotated data, demonstrating a high performance and ability to develop a Deep Learning model without annotated data. Our proposal models with 1D-CNN and 1DCNN-LSTM architectures reached a AUC-PRC average of 98 and 97%, respectively.

In the future, there is still room for improvement in pattern recognition, through unsupervised learning models, to train a multiclass self-learning model in a single run. We also intend to implement reinforcement learning in the proposed model to improve the stopping criterion. We will also evaluate the self-learning semi-supervised deep learning networks for other databases of MTS and other architectures such as Transformers.

TABLE 3
Performance comparison of Semi-Supervised Self Learning and Supervised Learning

Model	Metric	Normal (%)	QR-PCK (%)	SC-DHSV (%)	AVG (%)	Weighted AVG (%)	SD (%)
1D-CNN self-learning	ACC	94.210	94.210	94.210	94.210	94.210	0.000
	PR	90.027	100.000	96.368	95.465	93.850	5.048
	REC	100.000	84.899	95.824	93.574	95.030	7.798
	F1	94.752	91.833	96.095	94.227	94.116	2.179
	SP	90.958	100.000	98.821	96.593	94.743	4.916
	AUC-ROC	99.748	98.983	99.535	99.422	99.496	0.395
	AUC-PRC	99.653	98.258	97.758	98.556	98.966	0.982
1D-CNN Supervised	ACC	98.319	98.319	98.319	98.319	98.319	0.000
	PR	99.703	94.737	99.206	97.882	98.201	2.735
	REC	97.252	99.566	99.734	98.851	98.292	1.387
	F1	98.462	97.092	99.470	98.341	98.220	1.194
	SP	99.642	98.190	99.796	99.209	99.248	0.886
	AUC-ROC	99.879	99.962	99.996	99.946	99.920	0.060
	AUC-PRC	99.923	99.888	99.984	99.932	99.922	0.048
1DCNN-LSTM self-learning	ACC	96.532	96.532	96.532	96.532	96.532	0.000
	PR	92.778	100.000	100.000	97.593	95.943	4.170
	REC	100.000	89.765	99.097	96.287	96.922	5.667
	F1	96.254	94.607	99.547	96.802	96.277	2.515
	SP	93.744	100.000	100.000	97.915	96.486	3.612
	AUC-ROC	99.907	99.675	99.997	99.860	99.854	0.166
	AUC-PRC	99.883	99.436	99.990	99.770	99.771	0.294
1DCNN-LSTM Supervised	ACC	99.040	99.040	99.040	99.040	99.040	0.000
	PR	99.659	97.447	99.339	98.815	98.975	1.196
	REC	98.602	99.349	99.867	99.273	99.007	0.636
	F1	99.128	98.389	99.602	99.040	98.987	0.611
	SP	99.582	99.149	99.831	99.521	99.495	0.345
	AUC-ROC	99.963	99.971	100.000	99.978	99.971	0.019
	AUC-PRC	99.971	99.911	99.998	99.960	99.958	0.045

SD: Standard Deviation. AVG: Average. ACC: Accuracy. PR: Precision. REC: Recall. F1: F1-Score. SP: Specificity. AUC-ROC: Area Under the Receiver Operating Characteristic Curve. AUC-PRC: Area Under the Precision-Recall Curve. QR-PCK: Quick restriction in Production Choke (PCK). SC-DHSV: Spurious closure of Downhole Safety Valve (DHSV).

TABLE 4
Processing time of self-learning and Supervised Learning models

Model	Normal and QR-PCK (s)	Normal and SC-DHSV (s)	Sum (s)	Prediction Time (s) for the entire test dataset
1D-CNN (self-learning)	1,950.13	1,195.70	3,145.83	00.12
1D-CNN (Supervised)	N/A	N/A	300.80	00.11
1DCNN-LSTM (self-learning)	1,971,59	1,554.81	3,526.40	00.15
1DCNN-LSTM (Supervised)	N/A	N/A	433.27	00.15

QR-PCK: Quick restriction in Production Choke (PCK). SC-DHSV: Spurious closure of Downhole Safety Valve (DHSV).

ACKNOWLEDGMENTS

The authors would like to thank the partner company Petrobras for providing the data. The authors also thank the CS2I for providing the computational resources needed to develop this work and to access to supercomputer, and the Reference Center for Artificial Intelligence, both at SENAI CIMATEC. In addition, we acknowledge the NVIDIA/CIMATEC AI Joint Lab for providing technical support

REFERENCES

[1] R. M. Souza, E. G. S. Nascimento, U. A. Miranda, W. J. D. Silva, and H. A. Lepikson, "Deep learning for diagnosis and classification of faults in industrial rotating machinery," *Computers Industrial Engineering*, vol. 153, no. N/A, p. 107060, mar 2021.

[2] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 2019, pp. 6105–6114. [Online]. Available: https://proceedings.mlr.press/v97/tan19a.html

[3] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting Self-Supervised Visual Representation Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] E. M. Turan and J. Jäschke, "Classification of undesirable events in oil well operation," in *2021 23rd International Conference on Process Control (PC)*, 2021, pp. 157–162.

[5] M. A. Marins, B. D. Barros, I. H. Santos, D. C. Barrionuevo, R. E. V. Vargas, T. de M. Prego, A. A. de Lima, M. L. R. de Campos, E. A. B. da Silva, and S. L. Netto, "Fault detection and classification in oil wells and production/service lines using random forest," *Journal of Petroleum Science and Engineering*, vol. 197, p. 107879, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920410520309372

[6] X. Li, P. Lu, L. Hu, X. Wang, and L. Lu, "A novel self-learning semi-supervised deep learning network to detect fake news on social media," *Multimedia Tools and Applications*, vol. 81, no. 14, pp. 19 341–19 349, 2022. [Online]. Available: https://doi.org/10.1007/s11042-021-11065-x

[7] C. Voß, B. Eiteneuer, and O. Niggemann, "Incorporating Uncertainty into Unsupervised Machine Learning for Cyber-Physical Systems," in *2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS)*, vol. 1, 2020, pp. 475–480.

[8] A. K. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, oct 2006.

[9] K.-P. Kortmann, M. Fehsenfeld, and I. M. Wielitzka, "Feature Extraction from Heterogeneous Multivariate Time Series Data of Mechatronic Systems by Autoencoder Networks," 2021.

[10] J. von Schleinitz, M. Graf, W. Trutschnig, and A. Schröder, "VASP: An autoencoder-based approach for multivariate anomaly detection and robust time series prediction with application in motorsport," *Engineering Applications of Artificial Intelligence*, vol. 104, p. 104354, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197621002025

[11] F. I. sousa, C. T. Farias, S. W. J. Dantas, G. L. Lefol, S. A. A. Bandeira, D. M. F. L. Soares, V. R. E. Vaz, and N. E. G. Sperandio, "Unsupervised machine learning for anomaly detection in multivariate time series data of a rotating machine from an oil and gas platform," 2021. [Online]. Available: https://www.iiisci.org/journal/PDV/sci/pdfs/ZA422HO21.pdf

[12] F. I. Sousa, C. T. Farias, S. W. J. Dantas, G. L. L. Nani, and N. E. G. Sperandio, "Detecting Interesting and Anomalous Patterns In Multivariate Time-Series Data in an Offshore Platform Using Unsupervised Learning," vol. Day 2 Tue, August 17, 2021, 08 2021, d021S025R003. [Online]. Available: https://doi.org/10.4043/31297-MS

[13] F. I. sousa, G. L. Lefol, and N. E. G. Sperandio, "Multivariate Real Time Series Data Using Six Unsupervised Machine Learning Algorithms," in *Anomaly Detection - Recent Advances, Issues and Challenges [Working Title]*. IntechOpen, nov 2020, no. tourism, p. 13. [Online]. Available: https://doi.org/10.5772/intechopen.94944

[14] Figueirêdo Ilan Sousa, C. T. Farias, S. W. J. Dantas, M. U. Almeida, V. R. E. Vaz, D. L. Soares, and N. E. G. Sperandio, "Multivariate anomaly detection in rotating machinery of the oil and gas industry using unsupervised machine learning," *Rio Oil and Gas Expo and Conference*, vol. 20, no. 2020, pp. 405–406, dec 2020. [Online]. Available: https://doi.org/10.48072/2525-7579.rog.2020.405

[15] Y. Choi, H. Lim, H. Choi, and I.-J. Kim, "GAN-Based Anomaly Detection and Localization of Multivariate Time Series Data for Power Plant," in *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2020, pp. 71–74.

[16] H. Huang, C. Xu, and S. Yoo, "Interpretable Temporal GANs for Industrial Imbalanced Multivariate Time Series Simulation and Classification," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, ser. SAC '21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 924–933. [Online]. Available: https://doi.org/10.1145/3412841.3441967

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[18] W. Jin, T. Derr, H. Liu, Y. Wang, S. Wang, Z. Liu, and J. Tang, "Self-supervised Learning on Graphs: Deep Insights and New Direction," 2020. [Online]. Available: https://arxiv.org/abs/2006.10141

[19] G. He, Y. Li, and W. Zhao, "An uncertainty and density based active semi-supervised learning scheme for positive unlabeled multivariate time series classification," *Knowledge-Based Systems*, vol. 124, pp. 80–92, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705117301235

[20] E. McEntee, "Enhancing Partially Labelled Data: Self Learning and Word Vectors in Natural Language Processing," 2019.

[21] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative Unsupervised Feature Learning with Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf

[22] N. Komodakis and S. Gidaris, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018. [Online]. Available: https://hal-enpc.archives-ouvertes.fr/hal-01832768

[23] L. L. Santos, I. Winkler, and E. G. S. Nascimento, "RL-SSI Model: Adapting a Supervised Learning Approach to a Semi-Supervised Approach for Human Action Recognition," *Electronics*, vol. 11, no. 9, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/9/1471

[24] M. Noroozi and P. Favaro, "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 69–84.

[25] R. Zhang, P. Isola, and A. A. Efros, "Colorful Image Colorization," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 649–666.

[26] D. A. Guillen Rosaperez, "Self-Learning Methodology for Failure Detection in an Oil- Hydraulic Press : Predictive maintenance," Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS), 2020.

[27] T. Lintonen and T. Räty, "Self-learning of multivariate time series using perceptually important points," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1318–1331, 2019.

[28] L. Wei and E. Keogh, "Semi-Supervised Time Series Classification," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, 2006, pp. 748–753. [Online]. Available: https://doi.org/10.1145/1150402.1150498

[29] C. A. Ratanamahatana and D. Wanichsan, *Stopping Criterion Selection for Efficient Semi-supervised Time Series Classification*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–14. [Online]. Available: https://doi.org/10.1007/978-3-540-70560-4_1

[30] N. Begum, B. Hu, T. Rakthanmanon, and E. Keogh, *A Minimum Description Length Technique for Semi-Supervised Time Series Classification*. Cham: Springer International Publishing, 2014, pp. 171–192. [Online]. Available: https://doi.org/10.1007/978-3-319-04717-1_8

[31] M. González, C. Bergmeir, I. Triguero, Y. Rodríguez, and J. M. Benítez, "On the stopping criteria for k-Nearest Neighbor in positive unlabeled time series classification problems," *Information Sciences*, vol. 328, pp. 42–59, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025515006131

[32] M. N. Nguyen, X.-L. Li, and S.-K. Ng, "Ensemble Based Positive Unlabeled Learning for Time Series Classification," in *Database Systems for Advanced Applications*, S.-g. Lee, Z. Peng, X. Zhou, Y.-S. Moon, R. Unland, and J. Yoo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 243–257.

[33] ——, "Positive unlabeled learning for time series classification," 2011. [Online]. Available: https://www.aaai.org/ocs/index.php/IJCAI/IJCAI11/paper/view/2989/3456

[34] M. Pavlinek and V. Podgorelec, "Text classification method based on self-training and LDA topic models," *Expert Systems with Applications*, vol. 80, pp. 83–93, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417417301665

[35] R. Dorado and S. Ratté, "Semisupervised text classification using unsupervised topic information," in *The Twenty-Ninth International Flairs Conference*, 2016.

[36] M. Gowda Harsha S. and Suhil, G. D. S., and R. L. Narayana, "Semi-supervised Text Categorization Using Recursive K-means Clustering," in *Recent Trends in Image Processing and Pattern Recog-*

nition, M. Santosh K.C. and Hangarge, B. Vitoantonio, and N. Atul, Eds. Singapore: Springer Singapore, 2017, pp. 217–227.

[37] R. E. V. Vargas, C. J. Munaro, P. M. Ciarelli, A. G. Medeiros, B. G. do Amaral, D. C. Barrionuevo, J. C. D. de Araújo, J. L. Ribeiro, and L. P. Magalhães, "A realistic and public dataset with rare undesirable real events in oil wells," *Journal of Petroleum Science and Engineering*, vol. 181, p. 106223, oct 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0920410519306357

[38] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, *Feature selection for high-dimensional data*. Springer, 2015.

[39] E. G. S. Nascimento, O. Tavares, and A. De Souza, "A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance," in *ICAI'2015 - International Conference on Artificial Intelligence*. Las Vegas, Nevada, USA: ICAI'15 - The 17th International Conference on Artificial Intelligence, 2015, pp. 622–628. [Online]. Available: https://www.researchgate.net/publication/282330724_A_Cluster-based_Algorithm_for_Anomaly_Detection_in_Time_Series_Using_Mahalanobis_Distance

[40] E. Keogh and S. Kasetty, "On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration," *Data Mining and Knowledge Discovery*, vol. 7, no. 4, pp. 349–371, 2003. [Online]. Available: https://doi.org/10.1023/A:1024988512476

[41] M. Stuart, "Understanding Robust and Exploratory Data Analysis," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 33, no. 3, pp. 320–321, 1984. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.2307/2988240

[42] J. Zhu, H. Chen, and W. Ye, "A Hybrid CNN-LSTM Network for the Classification of Human Activities Based on Micro-Doppler Radar," *IEEE Access*, vol. 8, pp. 24 713–24 720, 2020.

[43] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[44] I. Y. Susanto, T.-Y. Pan, C.-W. Chen, M.-C. Hu, and W.-H. Cheng, *Emotion Recognition from Galvanic Skin Response Signal Based on Deep Hybrid Neural Networks*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 341–345. [Online]. Available: https://doi.org/10.1145/3372278.3390738

Ilan Sousa Figueirêdo was born in Salvador, Brazil. He received the master's degrees in Computational Modeling and Industrial Technology from SENAI CIMATEC of Salvador, Brazil in 2018. He is currently Ph.D student of Computational Modeling and Industrial Technology from SENAI CIMATEC and he is also an expert in Artificial Intelligence at SENAI CIMATEC's supercomputing center. He is currently part of the Artificial Intelligence Reference Center at SENAI CIMATEC.

Lílian Lefol Nani Guarieiro was born in Lavras, MG, Brazil. She is Graduated in Chemistry at Centro Universitário de Lavras (2003), Master in Organic Chemistry and Specialization in Petroleum Chemistry at Federal University of Rio de Janeiro (2006), PhD in Analytical Chemistry at Federal University of Bahia (2010), Sandwich Doctorate at Virgina Polytechnic Institute in Blacksburg, VA-USA and Pos-Doc by the National Institute of Science and Technology for Energy and Environment (2011). She is currently Coordinator of the Professional Master in Sustainable Development at SENAI CIMATEC. She has experience in Environmental Chemistry, Analytical Chemistry and Organic Chemistry with an emphasis on oil, gas and biofuel.

Erick Giovani Sperandio Nascimento was born in Vila Velha, ES, Brazil. He is Associate Professor/Reader of Artificial Intelligence (AI) at the Surrey Institute for People-Centred AI, University of Surrey, UK. He is also an Associate Professor at SENAI CIMATEC, Bahia, Brazil. He currently coordinates, leads and participates in RD projects in the areas of AI, computational modeling and supercomputing applied to different areas such as Renewable Energies, Air Pollution, Atmospheric Sciences, Oil and Gas, Health, Advanced Manufacturing, advising undergraduate and graduate students. University graduate. Acts as one of the Principal Investigators of the National Center for Applied Research in Artificial Intelligence (CPA-IA) of SENAI CIMATEC, focusing on Industry, being one of the six CPA-IA in Brazil approved by the Ministry of Science, Technology and Innovation (MCTI) , Foundation for Research Support of the State of São Paulo (FAPESP) and Internet Steering Committee in Brazil (CGI.br). He is a Certified Instructor and University Ambassador of the NVIDIA Deep Learning Institute (DLI) in the areas of Deep Learning, Computer Vision, Natural Language Processing, Recommender Systems, AI applications for anomaly detection and predictive maintenance.

Supplementary Materials: A Novel Self Deep Learning Semi-Supervised Model to Classify Unlabeled Multivariate Time-Series from Offshore Naturally Flowing Wells of OG Industry

Ilan Sousa Figueirêdo, Lílian Lefol Nani Guarieiro, and Erick Giovani Sperandio Nascimento

In this supplementary material, we present information about the balance of the data, the architectural configuration of the deep learning models, the loss curve of the training, comparative evaluation between the semi-supervised and supervised learning and, finally, the pseudo-code.

Appendix A: Dataset Setup

Figure 1 show the amount of database data points used for this study

As the asset is expected to perform within the projected pattern, it is common for the industry to have unbalanced data, ss shown in Figure 1.

Appendix B: Model Setup

The architectures of 1D-CNN and 1DCNN-LSTM Neural Networks are represented in Figure 2A and 2B, repectively.

Appendix C: Loss Curve

In Figures 3 and 5 shows the training loss curve, which are sharper and demonstrates that the model had a successful learning, unlike the last training curve as Figures 4 and 6 shows, which there are almost no learning in the model.

As previously described, we randomly chose 2 samples with QR-PCK data and 1 sample of SC-DHSV data for testing the models.

Our final experiment was a multiclass classification task. Thereby, the proposed model needs to be trained incrementally (one failure class at a time). Therefore, first the model was trained with the normal classes and the QR-PCK. Then, we performed a Transfer Learning to train a multiclass model with a new data. In this experiment, the SC-DHSV was added.

- *Ilan Sousa Figueirêdo, Lílian Lefol Nani Guareiro and Erick Giovani Sperandio Nascimento are with the University Center SENAI CIMATEC, Salvador, BA 41650-010, Brazil. E-mail: erick.sperandio@surrey.ac.uk.*
- *Erick Giovani Sperandio Nascimento is also with the Surrey Institute for People-Centred Artificial Intelligence, Faculty of Engineering and Physical Sciences, University of Surrey, Guildford GU2 7XH, United Kingdom.*

Manuscript received April october, 2022; revised August 26, 2015.

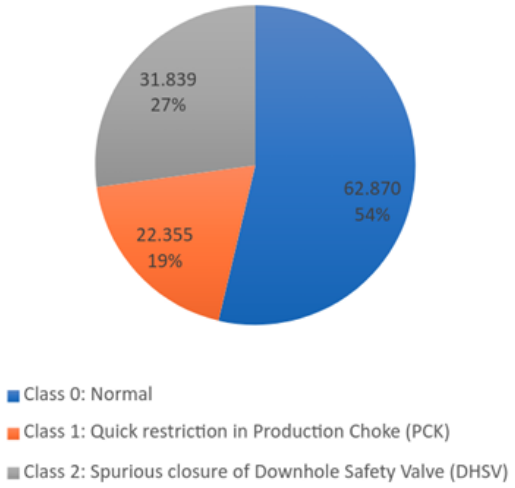


Fig. 1. Data Balancing.

Appendix D: Performance Comparison Assessment

The performance difference between supervised machine learning and self-learning is calculated as follows:

$$PD = Metrics(SL) - Metrics(SL4M) * 100$$

Where PD denotes performance difference, SL means supervised learning and $SL4M$ means the Self-Learning Model for Multivariate Time-Series (proposed model of this work).

Table 1 shows the performance difference between supervised and semi-supervised machine learning models.

Negative values in Table 1 mean that the Self Semi-Supervised Deep Learning Model for MTSl was superior to the supervised learning model.

TABLE 1
Performance difference between the Supervised and Self-Learning approaches (highlighted in bold when the self-learning model is superior to the supervised model).

Model		Normal (%)	QR-PCK (%)	SC-DHSV (%)	AVG (%)	Weighted AVG (%)	SD (%)
1D-CNN	ACC	4.109	4.109	4.109	4.109	4.109	0.000
	PR	9.676	**-5.263**	2.839	2.417	4.351	7.479
	REC	**-2.748**	14.667	3.910	5.276	3.261	8.788
	F1	3.711	5.259	3.374	4.114	4.105	1.005
	SP	8.684	**-1.810**	0.975	2.616	4.505	5.436
	AUC-ROC	0.131	0.980	0.461	0.524	0.425	0.428
	AUC-PRC	0.270	1.630	2.226	1.375	0.956	1.003
1DCNN-LSTM	ACC	2.508	2.508	2.508	2.508	2.508	0.000
	PR	6.881	**-2.553**	**-0.661**	1.222	3.031	4.991
	REC	**-1.398**	9.584	0.770	2.985	2.085	5.817
	F1	2.874	3.782	0.056	2.237	2.710	1.943
	SP	5.838	**-0.851**	**-0.169**	1.606	3.009	3.681
	AUC-ROC	0.056	0.296	0.003	0.118	0.117	0.156
	AUC-PRC	0.088	0.476	0.009	0.191	0.187	0.250

SD: Standard Deviation. AVG: Average. QR-PCK: Quick restriction in Production Choke (PCK). SC-DHSV: Spurious closure of Downhole Safety Valve (DHSV).

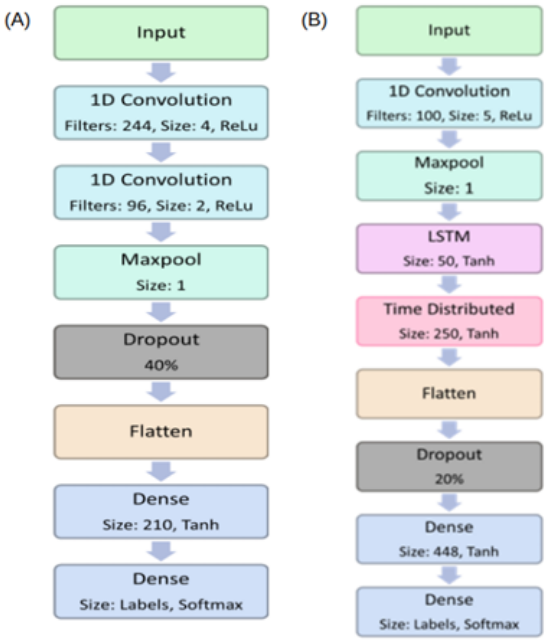


Fig. 2. Architecture diagram representation of (A) 1D-CNN and (B) Hybrid 1DCNN-LSTM models.

APPENDIX E: PSEUDO-CODE

The equation to automate the CF of C-AMDATS is formally presented in the following Algorithm 1:

The pseudo-code of the proposed algorithm is presented in the Algorithm 2. The algorithm starts in line 1 by randomly selecting the sample from the entire unlabeled dataset. After that, in lines 2-9, it starts the unsupervised phase ($t = 0$). Firstly, in this phase, hidden patterns are found and the anomaly probability (or anomaly score) of each pattern are calculated. For that, in lines 10-18, the algorithm uses the anomaly score to establish the anomalous and normal patterns. The Pattern with the lowest anomaly score is determined as normal, the remainder are determined as anomalous. The loop of lines 3-18 terminates when there are no more selected samples left. After this loop, we have the first pseudo-labeled set D^0_{pseu}. In lines 20-36 starts the supervised phase. Firstly, in this phase, in line 21, the algorithm builds a new training dataset. For $t = 1$, the D^1_{train} is equal than D^0_{train}. This happens because in the first iteration $D^0_{train} = D^0_{pseu}$. So, using the training set, in lines 22-26, a pre-processing step is performed for feature extraction, reshape and rescaling. The following step, in lines 27-29, a transfer learning can be applied, if there is a pre-trained model and $t = 1$. In line 31 the supervised model is trained. After that, in lines 32-33, more unlabeled samples are loaded for model prediction. In line 34, the softmax layer computes the predicted probability of each class. The last step, in line 36, the confidence layer filters out the confident pseudo-labels. The loop of lines 20-38 only ends when the stopping criterion is met (Equation 11).

Algorithm 1 Calculating the Clustering Factor CF

Input: Multivariate time-series $X \epsilon \mathrm{R}^{nxm}$
Output: Clustering Factor CF
1: Scaled multivariate time-series $X_{Scaled} = MinMaxScaler\,(X, [0, 1])$
2: Select $m = argmin\,\{kurtosis\,(X_{Scaled})\}$
3: $CF = \frac{\mu(X^m_{Scaled})}{\sigma(X^m_{Scaled})}$

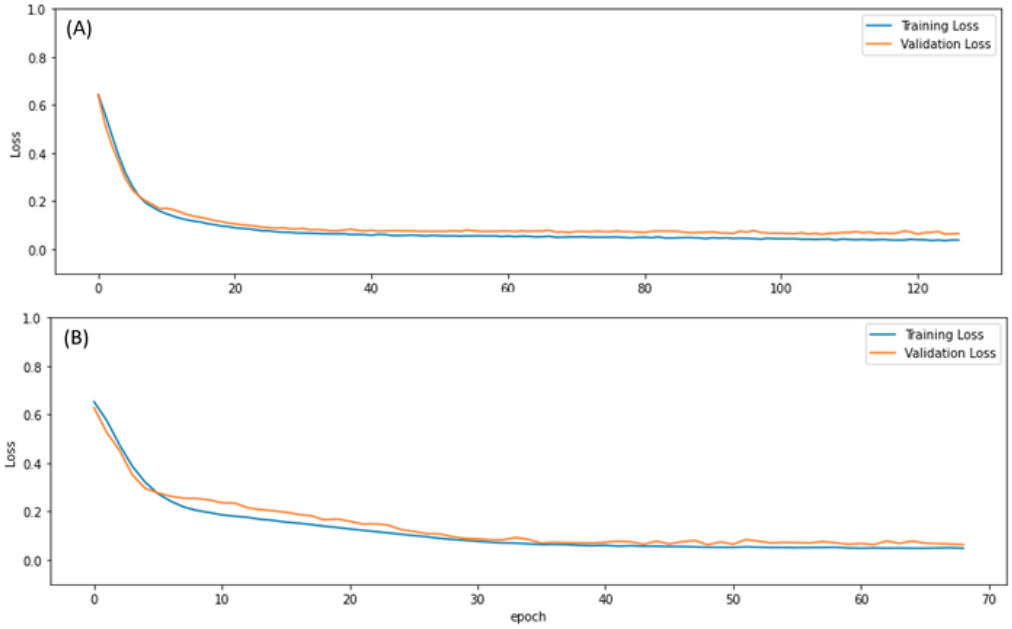


Fig. 3. Training loss of first iteration ($T = 1$) for normal and Quick restriction in Production Choke classes, for (A) 1D-CNN and (B) Hybrid 1DCNN-LSTM models

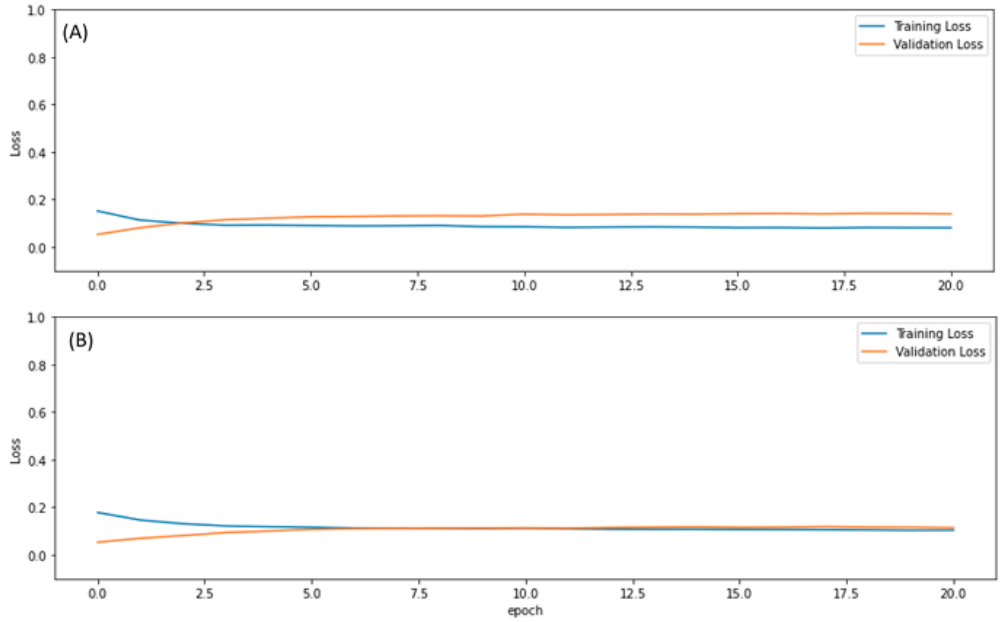


Fig. 4. Training loss of last iteration ($T = t$) for normal and Quick restriction in Production Choke classes, for (A) 1D-CNN and (B) Hybrid 1DCNN-LSTM models

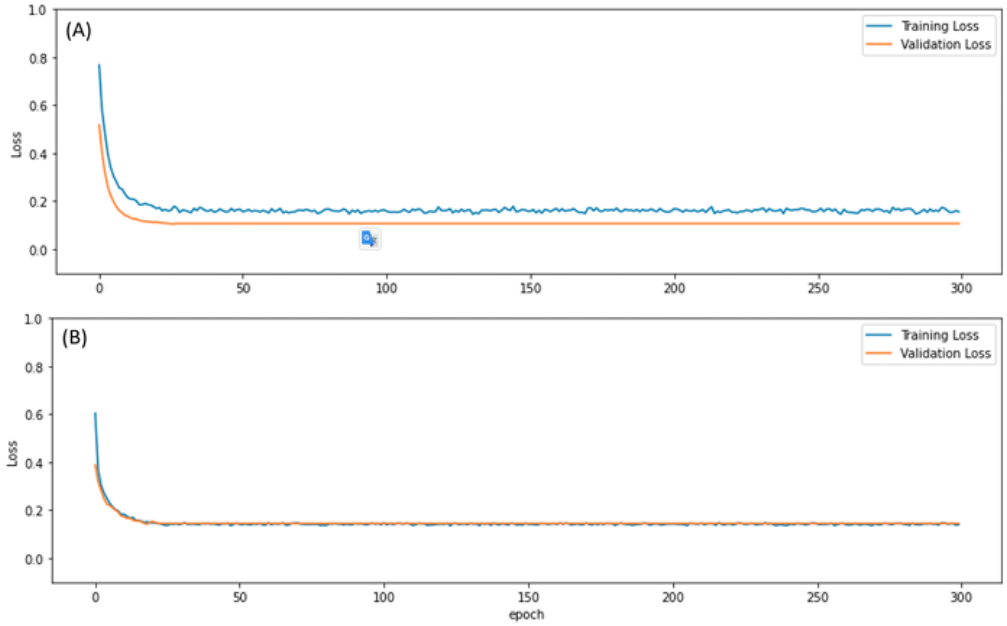


Fig. 5. Training loss of first iteration ($T = 1$) for normal and Spurious closure of Downhole Safety Valve classes, for (A) 1D-CNN and (B) Hybrid 1DCNN-LSTM models

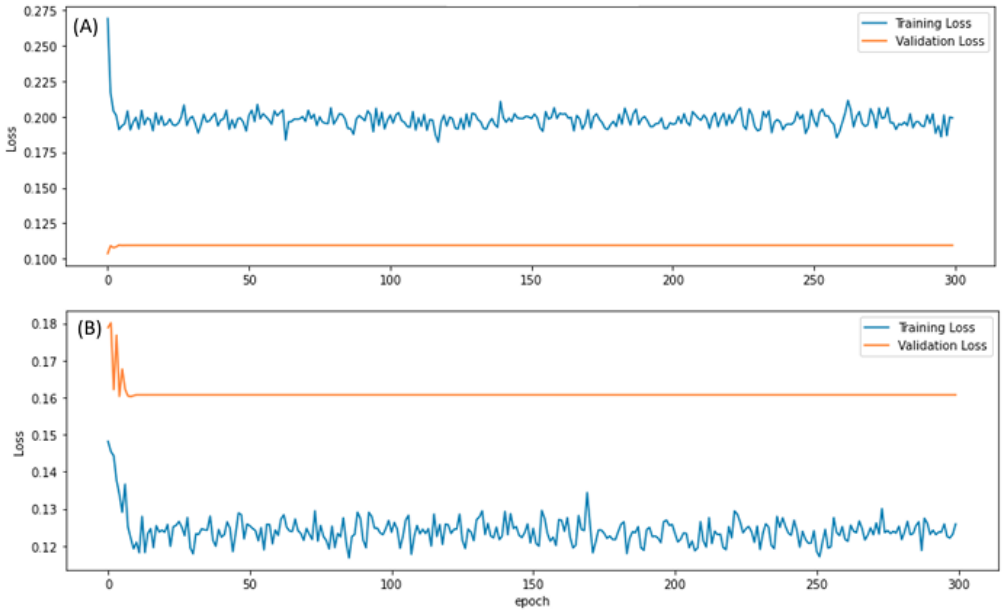


Fig. 6. Training loss of last iteration ($T = t$) for normal and Spurious closure of Downhole Safety Valve classes, for (A) 1D-CNN and (B) Hybrid 1DCNN-LSTM models

Algorithm 2 The Proposed Self Semi-Supervised Deep Learning Model for MTS

Input: Unlabeled data D_u, Window step s, Window size N, and $Look\ Back$
Output: Pseudo-Labeled D_{pseu}
Data: Multivariate time-series $X \epsilon R^{n \times m}$

$D_u^0 \leftarrow$ randomly select a sample of D_u
$t = 0$
While $t = 0$ do
    Calculate Clustering Factor CF ▷ As Algorithm 1
    Run C-AMDATS $(D_u^t = D_{train}^t)$ ▷ As Nascimento, et al., (2015)
        Find hidden patterns $|P|$
        Compute the Anomaly Score of each pattern $|P|$
        Sort patterns $|P|$ by Anomaly Score
    Return Anomaly Score
    for $i = 0$ to $|P|$ do
        If P$[i] = Min\,(Anomaly\ Score)$ then
            $\hat{y}\,(P\,[i]) = 0$
        Else
            $\hat{y}\,(P\,[i]) = 1$
        End if
        $D_{pseu}^t \leftarrow$ Join D_u^t with $\hat{y}$ As Eq. 3)
    end for
    $t \leftarrow +1$
end while
while stopping criteria is not satisfied do ▷ As Eq. 11
    $D_{train}^t \leftarrow D_{train}^{t-1} \cup D_{pseu}^{t-1}$ ▷ As Eq. 9
    Function(pre-processing) ▷ As Section 3.3.2
 Feature Extraction D_{train}^t through sliding window(s, N)
        Reshape D_{train}^t for $LookBack$
        Scaled D_{train}^t with $MinMaxScaler\,([0, 1])$
    end function
    if exist pre-trained model then
        Load model
    end if
    Supervised deep learning training (D_{train}^t)
    $D_u^t \leftarrow$ randomly select a new sample of D_u
    Supervised Deep Learning Predict (D_u^t)
    Compute predicted probability of each class ▷ As Eq. 7
    $D_{pseu}^t \leftarrow$ confidence layer filters out the confident pseudo-labels ▷ As Eq. 8
    $t \leftarrow t + 1$
end while

4. Conclusões e Trabalhos Futuros

A presente tese apresentou evidências de aprendizagem de máquina semi-supervisionada para classificação de falhas em linhas de produção de O&G offshore de poços surgentes sem dados rotulados. As conclusões deste trabalho foram:

- O desempenho do modelo semi-supervisionado alcançou um resultado muito próximo do modelo supervisionado, treinado com milhares de dados anotados. A título de exemplo, os modelos semi-supervisionado e supervisionado tiveram um AUC-PRC de 98% e 97%, respectivamente;
- Entretanto, o modelo semi-supervisionado requer um tempo maior para ser desenvolvido de aproximadamente 8 a 10 vezes que o modelo supervisionado. Pois, diferentemente do modelo supervisionado, que pode ser desenvolvido em um único treinamento, o modelo proposto é desenvolvido de forma incremental e interativamente. No entanto, quando comparado ao tempo que os especialistas levariam para anotar manualmente um grande conjunto de dados, o tempo de processamento da Autoaprendizagem profunda semi-supervisionado se justifica;
- O principal gargalo do tempo de processamento elevado é a etapa de aprendizado de máquina não supervisionada, que pode levar de 75 a 85% do tempo de desenvolvimento do modelo de Autoaprendizagem profunda semi-supervisionada.
- Portanto, o modelo proposto neste trabalho, pode ser relevante como ferramenta de anotação de dados e diagnóstico de eventos indesejáveis para cenários com pouco ou nenhum dado anotado de séries temporais multivariadas.

Ainda há campos de melhorias no algoritmo que podem ser explorados, portanto, seguem três sugestões:

- Otimizar o tempo de processamento do C-AMDATS através de programação paralelizada e uso de GPU, principalmente no da distância de Mahalanobis requer a inversão da covariância (ŠMELKO, et al., 2021; GAVAHI, et al., 2015);
- Adaptar a etapa de aprendizagem não supervisionada do modelo para possibilitar reconhecer padrões anômalos de modo multiclasse. Existem

vários métodos de agrupamento para particionar o conjunto de dados em grupos ou classes e lidar com os valores discrepantes presentes nos dados (TIWARI, AGARWAL, 2023). Com isto, tornaria o modelo mais aderente a classificar mais tipos de falhas a cada ciclo de treinamento;

- Aprofundar o estudo do modelo com métodos de aprendizagem de máquina por reforço (Jian, et al., 2023). Dessa forma, seria possível aumentar o desempenho do modelo proposto.

5. Produção Científica

No total seis trabalhos científicos foram publicados no período do doutorado. O primeiro trabalho teve o objetivo de explorar o estado da arte da aprendizagem de máquina não supervisionada para detecção de anomalia de séries temporais multivariadas. O segundo trabalho deu continuidade na avaliação comparativa dos algoritmos de aprendizagem de máquina não supervisionada para dados reais oriundo do monitoramento de ativos industriais offshore. O terceiro trabalho estendeu a análise comparativas para quatro estudos de casos reais da indústria de óleo e gás offshore. O quarto trabalho apresentou uma análise comparativa dos algoritmos de aprendizagem não supervisionada para dados de produção de óleo e gás offshore. O quinto trabalho apresentou os resultados preliminares da aprendizagem de máquina semi-supervisionada para dados reais sem rótulos. O sexto trabalho apresentou os resultados da aprendizagem de máquina semi-supervisionada para classificação de falhas multiclasse.

Seguem as referências dos trabalhos:

1. FIGUEIRÊDO, Ilan; GUARIEIRO, Lílian Lefol Nani; NASCIMENTO, Erick Giovani Sperandio. Multivariate real time series data using six unsupervised machine learning algorithms. In: **Anomaly Detection-Recent Advances, Issues and Challenges**. IntechOpen, 2020.
2. FIGUEIRÊDO, Ilan Sousa; SILVA, Wenisten Jose Dantas; CARVALHO, Tássio Farias; MIRANDA, Ubatan Almeida; FILHO, Leonildes Soares de Melo; VARGAS, Ricardo Emanuel Vaz; NASCIMENTO, Erick Giovani Sperandio. Multivariate Anomaly Detection in Rotating Machinery of the Oil and Gas Industry using Unsupervised Machine Learning. In: **Rio Oil & Gas Expo and Conference 2020**. Brazilian Petroleum, Gas and Biofuels Institute - IBP, 2020.
3. FIGUEIRÊDO, Ilan Sousa; SILVA, Wenisten Jose Dantas; CARVALHO, Tássio Farias; GUARIEIRO, Lílian Lefol. Nani; SANTOS, Alex Álisson Bandeira; DE MELO FILHO, Leonildes Soares; VARGAS, Ricardo Emmanuel Vaz; NASCIMENTO, Erick Giovani Sperandio. Unsupervised Machine Learning for Anomaly Detection in Multivariate Time Series Data of a Rotating Machine from an Oil and Gas Platform. Journal Systemics, Cybernetics and Informatics (JSCI). 2021. Disponível em: https://www.iiisci.org/journal/PDV/sci/pdfs/ZA422HO21.pdf

4. FIGUEIRÊDO, Ilan Sousa et al. Detecting interesting and anomalous patterns in multivariate time-series data in an offshore platform using unsupervised learning. In: **Offshore Technology Conference**. OnePetro, 2021.
5. FIGUEIRÊDO, Ilan Sousa; SILVA, Wenisten Jose Dantas; CARVALHO, Tássio Farias; FILHO, Leonildes Soares de Melo; VARGAS, Ricardo Emanuel Vaz; NASCIMENTO, Erick Giovani Sperandio. Patterns and anomalies recognition in multivariated time series of the offshore oil & gas industry using unsupervised machine learning. II Seminário de Inteligência Artificial Aplicada à Indústria do Petróleo 2020. Disponível em: https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s41405/?playlistId=playList-00798472-e29f-4916-a567-3eef81a70841
6. NASCIMENTO, E. G. S. N.; GUARIEIRO, L. L. N.; FIGUEIRÊDO, I. S. A Novel Self Deep Learning Semi-Supervised Approach to Classify Unlabeled Multivariate Time Series Data. GPU Technology Conference - GTC 2022. Disponível em: https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s41405/?playlistId=playList-00798472-e29f-4916-a567-3eef81a70841

REFERÊNCIAS

ABID, A.; KHAN, M. T.; KHAN, M. S. Multidomain Features-Based GA Optimized Artificial Immune System for Bearing Fault Detection. **IEEE Transactions on Systems, Man, and Cybernetics: Systems**, v. 50, n. 1, p. 348–359, jan. 2017.

AHANI, Ida Kalate; SALARI, Majid; SHADMAN, Alireza. Statistical models for multi-step-ahead forecasting of fine particulate matter in urban areas. **Atmospheric Pollution Research**, v. 10, n. 3, p. 689-700, 2019.

ALDRICH, C.; AURET, L. **Unsupervised Process Monitoring and Fault Diagnosis with Machine Learning Methods**. London: Springer, 2013.

ALI, Y. H. Artificial Intelligence Application in Machine Condition Monitoring and Fault Diagnosis. In: **Artificial Intelligence - Emerging Trends and Applications**. V. 275. InTech, 2018.

ALSAIF, Saif Abdulmohsen; ALOTHMAN, Tameem Saud. Predict Drilling Equipment Failure Using AI-Based Sound Waive Analysis Methodology. In: **Offshore Technology Conference**. OnePetro, 2022.

ALVES, LUCAS; NASCIMENTO, ERICK GIOVANI SPERANDIO; MOREIRA, DAVIDSON MARTINS. Hourly tropospheric ozone concentration forecasting using deep learning. **WIT Transactions on Ecology and the Environment**, v. 236, p. 129-138, 2019.

AMRUTHNATH, N.; GUPTA, T. **A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance**. 2018 5th International Conference on Industrial Engineering and Applications (ICIEA). IEEE, 2018. p. 355-361.

ARENA, Paolo et al. Image processing for medical diagnosis using CNN. **Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment**, v. 497, n. 1, p. 174-178, 2003.

BEHRENS, Vanessa; VIETE, Steffen. **A note on Germany's role in the fourth industrial revolution**. Arbeitspapier, 2020.

BISWAL, S.; SABAREESH, G. R. **Design and development of a wind turbine test rig for condition monitoring studies**. 2015 International Conference on Industrial

Instrumentation and Control, ICIC 2015. IEEE, 2015. p. 891-896.

BORGI, T. et al. **Data analytics for predictive maintenance of industrial robots**. 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET). IEEE, jan. 2017. p. 412-417. Disponível em: <http://ieeexplore.ieee.org/document/7983729/>

CARVALHO, T. P. et al. A systematic literature review of machine learning methods applied to predictive maintenance. **Computers & Industrial Engineering**, v. 137, n. April, p. 106024, nov. 2019.

CHEN, Z. et al. Real-Time Bearing Remaining Useful Life Estimation Based on the Frozen Convolutional and Activated Memory Neural Network. **IEEE Access**, v. 7, p. 96583–96593, 2019.

CHOLLET, Francois. Deep learning with Python. Simon and Schuster, 2021.

CINI, N.; YALCIN, G. A Methodology for Comparing the Reliability of GPU-Based and CPU-Based HPCs. **ACM Computing Surveys**, v. 53, n. 1, p. 1–33, 29 maio 2020.

CORTE, Vinicius Dalla; SANTOS, Vagner Kaefer Dos; CASANOVA, Dalcimar. Chatbot baseado em rede neural Long Short-Term Memory (LSTM): um estudo de caso baseado no livro William Shakespeare. In: X COMPUTER ON THE BEACH. 2019. 2019. 484-492 p.

DA SILVA FILHO, J. I.; ABE, Jair Minoro. **Fundamentos das Redes Neurais Paraconsistentes – Destacando Aplicações em Neurocomputação**. São Paulo: Editora Arte & Ciência, 2001.

DANIYAN, I. et al. Artificial intelligence for predictive maintenance in the railcar learning factories. **Procedia Manufacturing**, v. 45, n. 2019, p. 13–18, 2020.

DE OLIVEIRA DA COSTA, P. R. et al. Remaining useful lifetime prediction via deep domain adaptation. **Reliability Engineering & System Safety**, v. 195, p. 106682, 2020.

DEL CAMPO, Felipe Arias et al. Auto-adaptive multilayer perceptron for univariate time series classification. **Expert Systems with Applications**, v. 181, p. 115147, 2021.

DIN, E. N. Maintenance-Maintenance terminology. **Trilingual version EN**, v. 13306, p. 2017, 2018.

DONG, Y. et al. Prognostic and health management for adaptive manufacturing systems with online sensors and flexible structures. **Computers & Industrial Engineering**, v. 133, p. 57–68, jul. 2019.

ERBE, H. et al. INFOTRONIC TECHNOLOGIES FOR E-MAINTENANCE REGARDING THE COST ASPECTS. **IFAC Proceedings Volumes**, v. 38, n. 1, p. 1–12, 2005.

FARIA, Elisangela Lopes de. **Redes Neurais Convolucionais e Máquinas de Aprendizado Extremo Aplicadas Ao Mercado Financeiro Brasileiro**, f. 147. 2018. Tese (Engenharia Civil) - Coppe/ufrj, Rio de Janeiro, 2018.

FERRERO BERMEJO, J. et al. Review and Comparison of Intelligent Optimization Modelling Techniques for Energy Forecasting and Condition-Based Maintenance in PV Plants. **Energies**, v. 12, n. 21, p. 4163, 31 out. 2019.

FIGUEIRÊDO, Ilan Sousa; SILVA, Wenisten Jose Dantas; CARVALHO, Tássio Farias; MIRANDA, Ubatan Almeida; FILHO, Leonildes Soares de Melo; VARGAS, Ricardo Emanuel Vaz; NASCIMENTO, Erick Giovani Sperandio. Multivariate Anomaly Detection in Rotating Machinery of the Oil and Gas Industry using Unsupervised Machine Learning. In: **Rio Oil & Gas Expo and Conference 2020**. Brazilian Petroleum, Gas and Biofuels Institute - IBP, 2020.

FUSKO, Miroslav; RAKYTA, Miroslav; EDL, Milan. PREDICTIVE MAINTENANCE 4.0**. InvEnt 2019: Industrial Engineering–Invention for Enterprise**, 2019.

GAO, R. et al. Cloud-enabled prognosis for manufacturing. **CIRP Annals - Manufacturing Technology**, v. 64, n. 2, p. 749–772, 2015.

GAVAHI, Mohsen et al. High performance GPU implementation of k-NN based on Mahalanobis distance. In: **2015 International Symposium on Computer Science and Software Engineering (CSSE)**. IEEE, 2015. p. 1-6

GOMES SOARES, S.; ARAÚJO, R. An on-line weighted ensemble of regressor models to handle concept drifts. **Engineering Applications of Artificial Intelligence**, 2015.

GOUSSEAU, W. et al. **Analysis of the Rolling Rlement Bearing Data Set of the Center for Intelligent Maintenance Systems of the University of Cincinnati**. 13th International Conference on Condition Monitoring and Machinery Failure Prevention Technologies,

CM 2016/MFPT 2016. Charenton, France: 2016. Disponível em: https://hal.archives-ouvertes.fr/hal-01715193/

GUTZWILLER, Robert S.; REEDER, John. Human interactive machine learning for trust in teams of autonomous robots. In: **2017 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)**. IEEE, 2017. p. 1-3.

HAYKIN, Simon. **Redes neurais: princípios e prática**. Bookman Editora, 2008.

HOERL, R. W.; SNEE, R. D.; DE VEAUX, R. D. Applying statistical thinking to 'Big Data' problems. **WIREs Computational Statistics**, v. 6, n. 4, p. 222–232, 2014.

HUUHTANEN, T.; JUNG, A. **Predictive maintenance of photovoltaic panels via deep learning**. 2018 IEEE Data Science Workshop (DSW). IEEE, jun. 2018. p. 66-70. Disponível em: <https://ieeexplore.ieee.org/document/8439898/>

JARDINE, A. K. S.; LIN, D.; BANJEVIC, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. **Mechanical Systems and Signal Processing**, v. 20, n. 7, p. 1483–1510, out. 2006.

JIAN, Yifan et al. LAFD-Net: Learning with Noisy Pseudo Labels for Semi-Supervised Bearing Fault Diagnosis. **IEEE Sensors Journal**, 2023.

JIN, Wenjing et al. CPS-enabled worry-free industrial applications. In: 2017 **Prognostics and System Health Management Conference (PHM-Harbin)**. IEEE, 2017. p. 1-7.

JUNIOR, AMILTON SR; NASCIMENTO, ERICK GIOVANI SPERANDIO; MOREIRA, DAVIDSON MARTINS. Assessing recurrent and convolutional neural networks for tropospheric ozone forecasting in the Region of Vitória, Brazil. **WIT Trans Ecol Environ**, v. 244, p. 101-112, 2020.

KOLOKAS, N. et al. **Forecasting faults of industrial equipment using machine learning classifiers**. 2018 IEEE (SMC) International Conference on Innovations in Intelligent Systems and Applications (INISTA) 2018. IEEE, 2018. p. 1-6.

KORBICZ, J.M. KOŚCIELNY, Z. K. AND W. C. Fault Diagnosis: Models, Artificial Intelligence, Applications. **Kybernetes**, v. 34, n. 5, p. 742–743, jun. 2005.

KUMAR, A.; CHINNAM, R. B.; TSENG, F. An HMM and polynomial regression based approach for remaining useful life and health state estimation of cutting tools.

Computers and Industrial Engineering, 2019.

LE, Xuan-Hien et al. Application of long short-term memory (LSTM) neural network for flood forecasting. **Water**, v. 11, n. 7, p. 1387, 2019.

LECUN, Yann et al. **GradientBased Learning Applied to Document Recognition**. 1998. 46 p. Disponível em: http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf. Acesso em: 10 Ago. 2022.

LEE, J. et al. Predictive Manufacturing System - Trends of Next-Generation Production Systems. **IFAC Proceedings Volumes**, v. 46, n. 7, p. 150–156, 2013.

LEE, J.; KAO, H.-A.; YANG, S. Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment. **Procedia CIRP**, v. 16, p. 3–8, 2014.

LEE, Jay; BAGHERI, Behrad; KAO, Hung-An. Recent advances and trends of cyber-physical systems and big data analytics in industrial informatics. In: **International proceeding of int conference on industrial informatics (INDIN)**. 2014. p. 1-6.

LI, H. et al. Remaining useful life prediction using multi-scale deep convolutional neural network. **Applied Soft Computing**, v. 89, p. 106113, abr. 2020.

LINTONEN, Timo; RÄTY, Tomi. Self-learning of multivariate time series using perceptually important points. **IEEE/CAA Journal of Automatica Sinica**, v. 6, n. 6, p. 1318-1331, 2019.

LIU, H. et al. Unsupervised fault diagnosis of rolling bearings using a deep neural network based on generative adversarial networks. **Neurocomputing**, v. 315, p. 412–424, nov. 2018.

LUGER, George F. **Inteligência Artificial**. Tradução Daniel Vieira. 6. ed. Pearson, 2014. 614 p.

MANTOVANI, William Amaro. **Utilização de Redes Neurais Recorrentes na Caracterização de Cargas Não Lineares em Sistemas Elétricos**. Ilha Solteira-SP, 2011. 73 p. Dissertação (Engenharia Elétrica) - Universidade Estadual Paulista Júlio de Mesquita Filho.

MARINS, Matheus A. et al. Fault detection and classification in oil wells and production/service lines using random forest. **Journal of Petroleum Science and Engineering**, v. 197, p. 107879, 2021.

MOBLEY, R. Keith. **An introduction to predictive maintenance**. Elsevier, 2002.

MARTINO, C. DI et al. **Lessons Learned from the Analysis of System Failures at Petascale: The Case of Blue Waters**. 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. IEEE, jun. 2014. p. 610-621. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6903615>

MORETTIN, Pedro A.; TOLOI, Clélia M. C. **Análise De Séries Temporais**. 2. ed. São Paulo: Blucher, 2006. 538 p.

NASCIMENTO, E. G. S. N.; GUARIEIRO, L. L. N.; FIGUEIRÊDO, I. S. A Novel Self Deep Learning Semi-Supervised Approach to Classify Unlabeled Multivariate Time Series Data. GPU Technology Conference 2022. Disponível em: https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s41405/?playlistId=playList-00798472-e29f-4916-a567-3eef81a70841

NGUYEN, V. H. et al. An architecture of deep learning network based on ensemble empirical mode decomposition in precise identification of bearing vibration signal. **Journal of Mechanical Science and Technology**, v. 33, n. 1, p. 41–50, 2019.

NIE, B. et al. **A large-scale study of soft-errors on GPUs in the field**. 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, mar. 2016. p. 519-530 Disponível em: <http://ieeexplore.ieee.org/document/7446091/>

NIKOLIC, B. et al. PREDICTIVE MANUFACTURING SYSTEMS IN INDUSTRY 4.0: TRENDS, BENEFITS AND CHALLENGES. **Annals of DAAAM & Proceedings**, v. 28, 2017.

O'GRADY, S. **The RedMonk Programming Language Rankings: January 2020**. Disponível em: <https://redmonk.com/sogrady/2020/02/28/language-rankings-1-20/>. Acesso em: 14 jul. 2020.

QIAO, Weibiao et al. The forecasting of PM2. 5 using a hybrid model based on wavelet

transform and an improved deep learning algorithm. **IEEE Access**, v. 7, p. 142814-142825, 2019.

RAUCH, E.; LINDER, C.; DALLASEGA, P. Anthropocentric perspective of production before and within Industry 4.0. **Computers and Industrial Engineering**, 2020.

ROSAPEREZ, Diego Alonso Guillen. Self-Learning Methodology for Failure Detection in an Oil-Hydraulic Press: Predictive maintenance. 2020.

RUIZ-SARMIENTO, J.-R. et al. A predictive model for the maintenance of industrial machinery in the context of industry 4.0. **Engineering Applications of Artificial Intelligence**, v. 87, n. October 2019, p. 103289, jan. 2020.

RUIZ-SARMIENTO, J.-R.; GALINDO, C.; GONZALEZ-JIMENEZ, J. Building Multiversal Semantic Maps for Mobile Robot Operation. **Knowledge-Based Systems**, v. 119, p. 257–272, 2017.

SENJOBA, Lesego et al. One-Dimensional Convolutional Neural Network for Drill Bit Failure Detection in Rotary Percussion Drilling. **Mining**, v. 1, n. 3, p. 297-314, 2021.

SOARES, S. G. **Ensemble learning methodologies for soft sensor development in industrial processes**. [s.l.] University of Coimbra, 2015.

SCHUH, G. et al. Increasing data integrity for improving decision making in production planning and control. **CIRP Annals**, v. 66, n. 1, p. 425–428, 2017.

SHIN, J.-H.; JUN, H.-B.; KIM, J.-G. Dynamic control of intelligent parking guidance using neural network predictive control. **Computers & Industrial Engineering**, v. 120, p. 15–30, jun. 2018.

ŠMELKO, Adam et al. GPU-Accelerated Mahalanobis-Average Hierarchical Clustering Analysis. In **Euro-Par 2021: Parallel Processing: 27th International Conference on Parallel and Distributed Computing, Lisbon, Portugal, September 1–3, 2021, Proceedings 27**. Springer International Publishing, 2021. p. 580-595.

SPERANDIO NASCIMENTO, E. G.; TAVARES, O.; DE SOUZA, A. **A Cluster-based Algorithm for Anomaly Detection in Time Series Using Mahalanobis Distance**. ICAI'2015 - International Conference on Artificial Intelligence. Las Vegas, Nevada, USA: ICAI'15 - The 17th International Conference on Artificial Intelligence, 2015. p. 622. Disponível em:

https://www.researchgate.net/publication/282330724_A_Cluster-based_Algorithm_for_Anomaly_Detection_in_Time_Series_Using_Mahalanobis_Distance

TIWARI, Sadhana; AGARWAL, Sonali. Empirical analysis of chronic disease dataset for multiclass classification using optimal feature selection based hybrid model with spark streaming. **Future Generation Computer Systems**, v. 139, p. 87-99, 2023.

TRAN, Thanh; PHAM, Nhat Truong; LUNDGREN, Jan. A deep learning approach for detecting drill bit failures from a small sound dataset. **Scientific Reports**, v. 12, n. 1, p. 1-13, 2022.

TURAN, Evren M.; JÄSCHKE, Johannes. Classification of undesirable events in oil well operation. In: **2021 23rd International Conference on Process Control (PC)**. IEEE, 2021. p. 157-162.

USUGA CADAVID, J. P. et al. Machine learning applied in production planning and control: a state-of-the-art in the era of industry 4.0. **Journal of Intelligent Manufacturing**, v. 31, n. 6, p. 1531–1558, 11 ago. 2020.

VAFAEI, N.; RIBEIRO, R. A.; CAMARINHA-MATOS, L. M. Fuzzy early warning systems for condition based maintenance. **Computers and Industrial Engineering**, 2019.

VARGAS, R. E. V. et al. A realistic and public dataset with rare undesirable real events in oil wells. **Journal of Petroleum Science and Engineering**, v. 181, p. 106223, out. 2019.

WAN, J. et al. A Manufacturing Big Data Solution for Active Preventive Maintenance. **IEEE Transactions on Industrial Informatics**, 2017.

WANG, C. C.; TOO, G. P. J. Rotating machine fault detection based on HOS and artificial neural networks. **Journal of Intelligent Manufacturing**, 2002.

WANG, Ping et al. A hybrid-wavelet model applied for forecasting PM2. 5 concentrations in Taiyuan city, China. **Atmospheric Pollution Research**, v. 10, n. 6, p. 1884-1894, 2019.

WANG, X.-B. et al. Ensemble extreme learning machines for compound-fault diagnosis of rotating machinery. **Knowledge-Based Systems**, 2019.

WEI, G. et al. Reliability modeling with condition-based maintenance for binary-state deteriorating systems considering zoned shock effects. **Computers & Industrial**

Engineering, v. 130, p. 282–297, abr. 2019.

YUAN, J.; LIU, X. Semi-supervised learning and condition fusion for fault diagnosis. **Mechanical Systems and Signal Processing**, v. 38, n. 2, p. 615–627, jul. 2013.

ZHANG, J. et al. Long short-term memory for machine remaining life prediction. **Journal of Manufacturing Systems**, v. 48, p. 78–86, 2018.

ZHANG, Lanyi et al. Trend analysis and forecast of PM2. 5 in Fuzhou, China using the ARIMA model. **Ecological indicators**, v. 95, p. 702-710, 2018.

ZHAO, R. et al. Deep learning and its applications to machine health monitoring. **Mechanical Systems and Signal Processing**, v. 115, p. 213–237, 2019.

ZHU, Z. et al. A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis. **Neurocomputing**, v. 323, p. 62–75, jan. 2019.

ZUCATELLI, P. J. et al. An investigation on deep learning and wavelet transform to nowcast wind power and wind power ramp: A case study in Brazil and Uruguay. **Energy**, v. 230, p. 120842, 2021.

APÊNDICE A. Resultados da Utilização da MLP na Abordagem de Autoaprendizagem Semi-supervisionada

Neste apêndice são apresentados os resultados de uma MLP para uma análise comparativa com os modelos utilizados no experimento computacional desta tese.

Apesar de ser uma das arquiteturas de RNA mais simples, MLP continua sendo um tópico de interesse para a comunidade científica, pois os perceptrons de multicamadas ainda são frequentemente usados devido à sua flexibilidade e capacidade de ajustar uma ampla gama de funções suaves e não lineares com altos níveis de precisão (DEL CAMPO, et al., 2021).

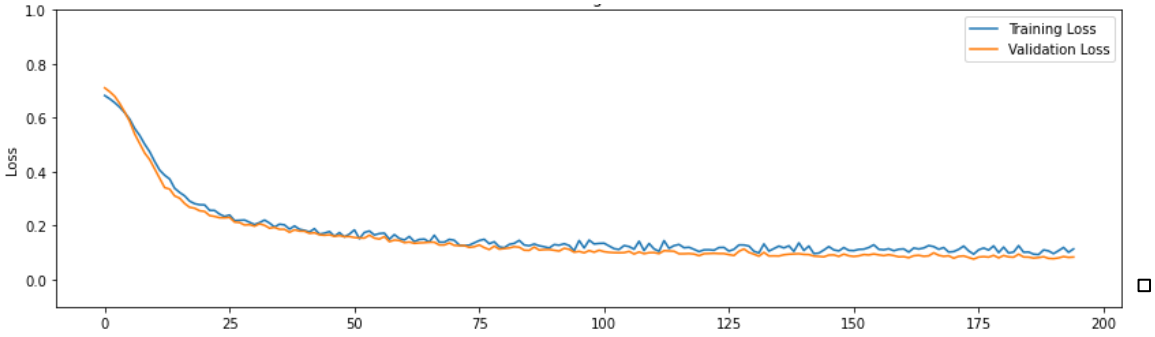
Portanto, diante do exposto, as mesmas configurações do experimento da aprendizagem de máquina semi-supervisionada (apresentadas seção 3.2) foram replicadas para um modelo com topologia de MLP. Desse modo, o modelo MLP substituiu os modelos CNN-1D e CNN1D-LSTM do Manuscrito 4.

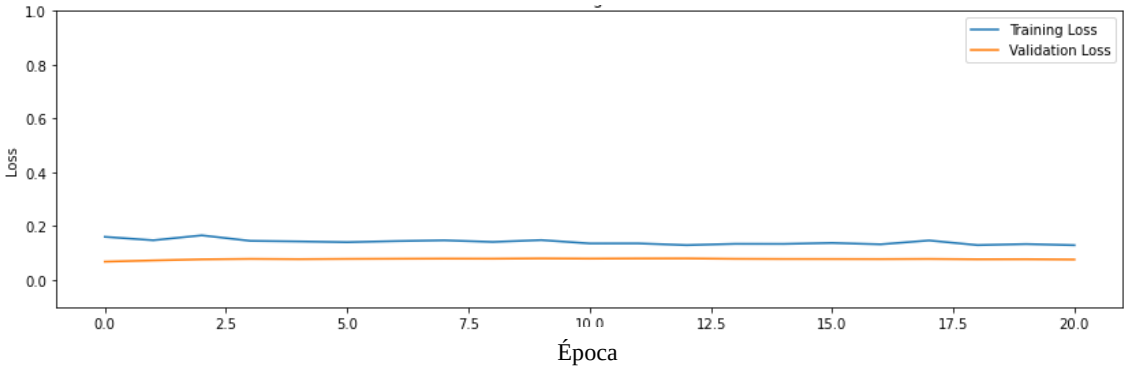
A seguinte arquitetura de RNA foi utilizada:

Figura AP.A.1. Representação do diagrama de arquitetura do modelo MLP.

Dessa forma, o modelo MLP foi utilizado na etapa de aprendizagem de máquina supervisionada para uma avaliação da capacidade de Autoaprendizagem. Desse modo, a Figura AP.A.2 mostra a (A) curva de perda (*loss*) de treinamento do modelo para $T=1$ e a (B) curva de perda (*loss*) de treinamento para $T=t$.

Figura AP.A.2. Curva de treinamento do modelo MLP para as classes Quick Restriction in Production Choke e Normal, (A) primeira iteração $(T=1)$ e (B) última iteração ($T=t$).





Na Figura AP.A.2, a curva para $T=1$ apresentou uma característica acentuara, o que significa que teve uma aprendizagem bem-sucedido. A curva para $T=t$ com uma característica quase retilínea e constante, mostrando que nesse treinamento não teve mais ganhos na aprendizagem do modelo.

Dessa forma, após a etapa de treinamento, a Tabela AP.1 apresenta a avaliação de desempenho do modelo MLP para as classes Quick Restriction in Production Choke (QR-PCK) e Normal. Essa avalição utilizou-se dados que não passaram no treinamento (dados de teste).

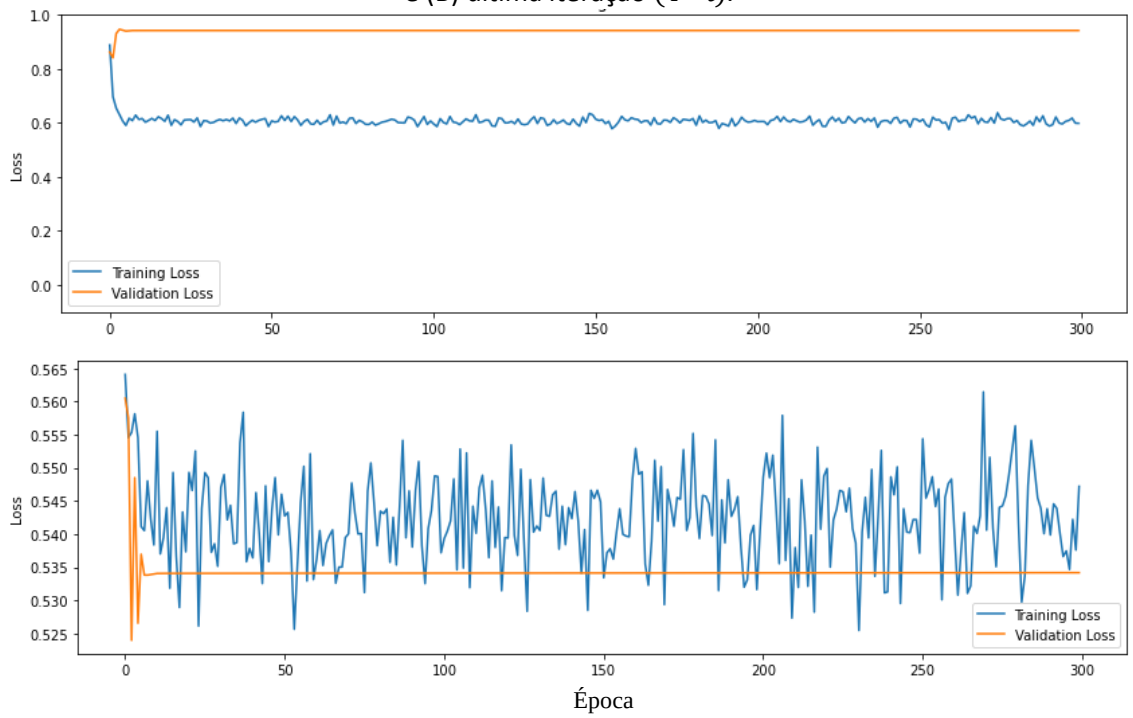
Tabela AP.A.1. Avaliação de desempenho de autoaprendizagem semi-supervisionada do modelo MLP para as classes Quick Restriction in Production Choke e Normal.

	Accuracy (%)	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**Specificity (%)**	**AUC-ROC (%)**	**AUC-PRC (%)**
Normal	91,254	100	81,529	89,825	100	98,652	98,744
QR-PCK	91,254	85,755	100	92,331	81,529	98,652	98,423
Média	91,254	92,877	90,764	91,078	90,764	98,652	98,583
Desvio Padrão	0,000	10,072	13,060	1,772	13,060	0,000	0,227

O resultado apresentado na Tabela AP.A.1 mostrou que o modelo MLP foi moderadamente inferior aos modelos propostos no Manuscrito 4. Entretanto, ainda precisaria incluir a classe Spurious Closure Down Safety Valve. Portanto (SC-DHSV) para realizar uma análise comparativa adequada.

Logo, usou-se Transfer Learning para evoluir o modelo de uma classificação binária para multicasse. Portanto, a partir de novos dados e Transfer Learning, realizou-se um novo ciclo de treinamento de um novo modelo incluindo a classe SC-DHSV. Desse modo, a Figura AP.A.3 mostra a (A) curva de perda (*loss*) de treinamento do modelo para $T=1$ e a (B) curva de perda (*loss*) de treinamento para $T=t$.

Figura AP.A.3. Curva de treinamento do modelo MLP para as classes Quick Restriction in Production Choke, Spurious Closure Down Safety Valve e Normal, (A) primeira iteração *(T = 1)* e (B) última iteração (*T=t*).



A Figura AP.A.2 apresentou *Underfitting* em ambos os treinamentos, demonstrando uma limitação do modelo para treinar com novos dados.

Por fim, a Tabela AP.A.2 apresenta os resultados alcançados do modelo MLP para as classes QR-PCK, SC-DHSV e Normal:

Tabel AP.A.2. Avaliação de desempenho de Autoaprendizagem Semi-Supervisionada do modelo MLP para as classes Quick Restriction in Production Choke, Down Safety Valve e Normal.

	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Specificity (%)	AUC-ROC (%)	AUC-PRC (%)
Normal	57,298%	59,928%	78,256%	67,876%	48,462%	87,430%	83,310%
QR-PCK	57,298%	36,598%	17,869%	24,014%	84,088%	69,303%	49,778%
SC-DHSV	57,298%	63,664%	70,203%	66,774%	81,276%	89,975%	83,783%
Média	57,298%	53,397%	55,443%	52,888%	71,275%	82,236%	72,290%
Desvio Padrão	0,000%	14,668%	32,788%	25,012%	19,807%	11,272%	19,498%

A Tabela AP.A.2 apresenta um resultado consideravelmente inferior aos resultados apresentados no Manuscrito 4 *(Table 3. Performance comparison of Semi-Supervised Self Learning and Supervised Learning)*. Portanto, não foi possível obter com essa topologia de RNA um resultado superior ou equiparável às topologias CNN-1D e CNN1D-LSTM.