UNIVERSIDADE D
COIMBRA

Ricardo Pereira da Silva Lopes

Abordagem Edge Computing para manutenção preditiva de equipamentos de bombagem

Dissertação no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores, área de especialização em Automação, orientada pelo Professor Doutor Urbano José Carreira Nunes e Professor Doutor Luís Conde Bento, apresentada ao Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro 2022

Ricardo Pereira da Silva Lopes

Abordagem Edge Computing para manutenção preditiva de equipamentos de bombagem

Dissertação no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores, área de especialização em Automação, orientada pelo Professor Doutor Urbano José Carreira Nunes e Professor Doutor Luís Conde Bento, apresentada ao Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro 2022

Agradecimentos

O processo de desenvolvimento da dissertação que a seguir se apresenta representa, não só o fim de um ciclo curricular, mas também o início de uma nova fase em que cada aprendizagem e esforço necessário para a sua conclusão se tornaram ensinamentos para a vida.

Começo por agradecer ao meu orientador, Professor Doutor Urbano Nunes e co-orientador Professor Doutor Luís Conde por todo o suporte e direcção que deram ao projecto, toda e qualquer sugestão dada foi sem dúvida enriquecedora para o trabalho desenvolvido.

Agradeço ao Instituto de Sistemas e Robótica - ISR pela oportunidade de trabalhar com disponibilidade de equipamentos e material necessários para o projecto. Este trabalho foi parcialmente suportado pelo ISR-UC e pela FCT através do financiamento UIDB/00048/2020.

Um obrigado à empresa Aqualgae, pela disponibilidade e intermediação com a indústria de aquacultura alvo de estudo.

Um agradecimento aos companheiros de jornada, colegas de trabalho, alguns amigos para a vida.

Por fim, agradeço o apoio dos meus pais, Isidro Lopes e Aurora Lopes, por todo o suporte e apoio ao longo de toda a jornada académica. Quero agradecer também o apoio da minha namorada, Catarina Castro e dos seus pais, Eng. Manuel Castro e Prof.ª Manuela Castro, por toda a motivação e apoio dado ao longo do meu percurso.

Resumo

Com o desenvolvimento de técnicas de Machine Learning cada vez mais robustas para detecção de anomalias, vários sectores da indústria procuram, de forma activa encontrar soluções com o objectivo de maximizar o tempo de operação e vida útil dos equipamentos de que dispõem. No sector da aquacultura é de especial importância o bom funcionamento das máquinas rotativas integradas nos seus sistemas, por se tratarem de componentes vitais para o funcionamento e crescimento das suas produções. O objectivo deste trabalho de investigação, foi desenvolver um modelo de Machine Learning baseado numa estrutura de AutoEncoder (AE) com capacidade de detecção de desvios de comportamento em máquinas rotativas, mantendo a simplicidade e capacidade de implementação em dispositivos de borda. Modelos baseados em AE têm a capacidade de extrair características automaticamente de dados normais, sendo os dados de saída do modelo a reconstrução dos seus dados de entrada. Se o erro associado à reconstrução da entrada for elevado, os dados são considerados anómalos, caso contrario representam dados normais. No trabalho desenvolvido, foram testados métodos de previsão de séries temporais num conjunto de dados disponibilizado pelo Center for Intelligent Maintenance Systems (IMS), de forma a ser escolhido o tipo de rede neuronal recursiva a ser utilizado no modelo AE. O modelo desenvolvido foi treinado e testado no IMS-Dataset e, de seguida, generalizado para um conjunto de dados adquirido em meio industrial. Estes dados foram adquiridos de motores integrados em sistemas cuja função é o controlo da temperatura e a recirculação da água usada em tanques de crescimento de peixes em sistemas de aquacultura. O modelo aplicado demonstrou boa capacidade na detecção de anomalias em séries temporais. Assim, os resultados da implementação do modelo evidenciaram o potencial do uso de abordagens recorrendo a AE's para detecção de anomalias e a sua capacidade de generalização, quando comparados a métodos de regressão convencionais.

Palavras-Chave: Aprendizagem Profunda, Detecção de Anomalias, Redes Neuronais Recursivas, AutoEncoder, Manutenção Preditiva

Abstract

With the development of more advanced anomaly detection techniques, several industry sectors actively seek to find solutions that aim to maximize the operating time and useful lifetime of the machinery. In the aquaculture sector it is of special importance the proper functioning of the rotating machines, as vital components of the fish productions. The objective of this research work was to develop a model of Machine Learning based on an AutoEncoder (AE) structure capable of detecting behavior deviations in machinery, while remaining simple and maintaining the ability to run on edge devices. Models based on AE have the ability to automatically extract features from normal data, the output data being a reconstruction of the input data. If the error between the input and the output data is high, the data is considered anomalous, otherwise it represents normal data. In this research work, forecasting methods of time series were tested on a dataset available from Center for Intelligent Maintenance Systems (IMS), in order to choose the type of Recursive Neural Network to be used in the proposed model. The developed model was tested in the IMS-Dataset and then it was generalized to a self-adquired industrial dataset. This industrial dataset was acquired from motors that integrate systems that control the temperature and do the re-circulation of the water of the fish growth tanks used in aquaculture systems. Our model demonstrated a good performance in the detection of anomalies in time series. Therefore, the results of the model´s implementation evidenced the potential of AE based models for anomaly detection and their generalization ability, when compared to classic regression techniques.

Keywords: Deep Learning, Anomaly Detection, Recurrent Neural Network, AutoEncoder, Predictive Maintenance

Conteúdo

Lista de Acrónimos

ADC Conversor Analógico-Digital.

AE Autoencoder.

AI Inteligência Artificial.

ANN Redes Neuronais Artificiais.

CC Cloud Computing.

CWT Transformada de Wavelet Contínua.

DFSBA Detecção de Falhas em Sistema de Bombagem de Aquacultura.

DL Deep Learning.

EC Edge Computing.

IF Isolation Forest.

IMS Center for Intelligent Maintenance Systems.

IoT Internet of Things.

iTree Árvore de Isolamento.

LPWAN Low Power Wide Area Network.

LSTM Long Short-Term Memory Neural Network.

MAD Desvio Padrão Médio Absoluto.

MAE Mean Absolute Error.

MAPE Mean Absolute Percentage Error.

MBC Monitorização Baseada em Condições.

MEMS Sistemas Micro-Electro-Mecânicos.

ML Machine Learning.

MP Manutenção Preditiva.

MSE Mean Squared Error.

OC-SVM Máquina de Vectores de Suporte de Uma Classe.

ODC Output Data Rate.

RMSE Root Mean Squared Error.

RNN Redes Neuronais Recursivas.

SVM Máquina de Vectores de Suporte.

UN Nações Unidas.

WAPE Weighted Absolute Percentage Error.

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A seguinte dissertação tem como objectivo o desenvolvimento e inclusão de metodologias para manutenção preventiva e detecção de falhas recorrendo a tecnologias de Edge Computing (EC) e métodos de Machine Learning (ML). Pretende-se assim aumentar a robustez de um sistema de Detecção de Falhas em Sistema de Bombagem de Aquacultura (DFSBA). Este capítulo apresenta as motivações que levaram à formulação deste trabalho de dissertação, bem como os principais objectivos e contributos. Resume também o fluxo das diferentes etapas do projecto.

1.1 Contexto e motivação

O relatório das Nações Unidas (UN) intitulado Situação Mundial da Pesca e Aquacultura 2020 aponta para um aumento no volume da produção em aquacultura de 32% até 2030 [9], o que implica que indústrias da área necessitam de sistemas cada vez mais robustos e eficientes. As indústrias de aquacultura estão dependentes da constante captação e recirculação de água para desenvolver as suas produções, para isso bombas de água são usadas em grande número e de forma constante, sendo de especial interesse a maximização da sua vida útil e a preservação das suas boas condições de funcionamento.

No capítulo 2, são descritos métodos para análise de dados recolhidos de máquinas industriais com o objectivo de detecção de falhas e sinalização de manutenções preventivas.

1.2 Formulação do problema e objectivos

Com a elaboração deste trabalho de dissertação, pretendeu-se implementar um modelo para análise e detecção de falhas mecânicas num sistema de bombagem no sector da aquacultura, com a possibilidade de ser generalizado a outras aplicações semelhantes. A instalação que será objecto de estudo utiliza um método de monitorização e detecção de falhas que recorre à análise de sensores de caudal, corrente e pressão, e a uma abordagem que, usando valores de *threshold*, comunica com serviços de Cloud Computing (CC) a possibilidade de existência de falhas assim que essas condições limite sejam ultrapassadas. Com vista a aumentar a robustez e desempenho do sistema DFSBA e de forma a que se consiga generalizar a máquinas semelhantes, importa fazer-se uso de técnicas de ML para detecção de anomalias que necessitem apenas de dados normais para a fase de treino, visto que, o acesso a dados previamente catalogados como anómalos é difícil em meio industrial. Foi feita a inclusão de sensores de vibração, mais concretamente acelerómetros, para análise das vibrações dos motores/bombas em estudo, sendo posteriormente entrada do modelo para detecção de anomalias. Os dados recolhidos pelos sensores são pré-processados com vista a remover possível ruído e posteriormente usados como entrada para treino de uma rede neuronal, como mostra a Fig.1.1.

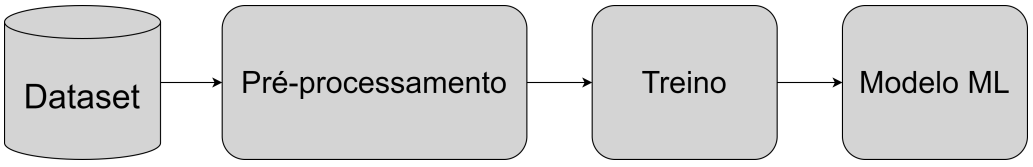


Figura 1.1: Ilustração do processo de tratamento de dados e treino do modelo de ML que se pretende implementar.

Para uma análise em tempo real e devido ao elevado número de dados adquiridos pelos sensores, o uso da tecnologia EC, que traz como vantagens relativamente à CC, menor latência, privacidade dos dados em tratamento e eficiência energética [10], será o foco da abordagem escolhida.

Pretende-se que o modelo de ML desenvolvido cumpra os requisitos necessários para implementação num microcontrolador com capacidade de processamento de algoritmos de ML, mas ainda assim de baixo poder computacional comparado aos recursos oferecidos pelos serviços *Cloud*.

O processo de aquisiçao de dados em mio industrial, o processamento dos dados

A estrutura do sistema actual e a componente do projecto proposto encontra-se na Fig.1.2, onde a área realçada (vermelho) representa a componente que se pretende adicionar ao sistema.

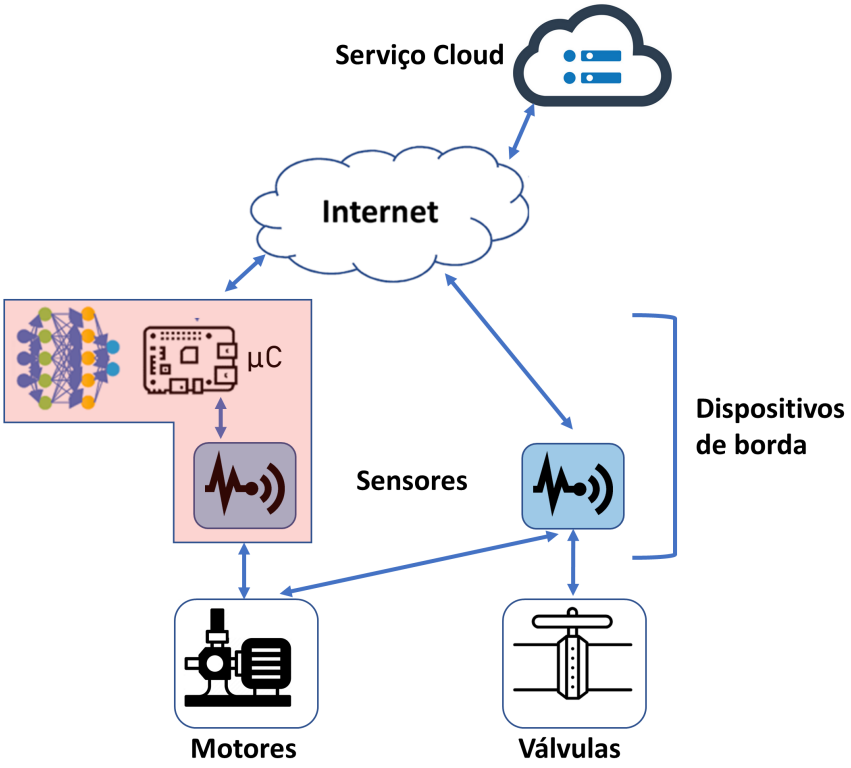


Figura 1.2: Estrutura do caso de estudo e da nova metodologia proposta (realçada a vermelho)

Objectivos complementares:

- Integração e teste do modelo em microcontrolador
- Possibilidade de instalar outros tipos de sensores (por exemplo, acústicos e temperatura) para detecção de anomalias;
- Monitorização das válvulas presentes na infraestrutura de forma a analisar e detectar falhas;
- Utilização dos serviços *Cloud* para processamento do modelo de ML e comparação de desempenho com dispositivos de borda;

1.3 Contribuições

No que diz respeito a modelos de ML para detecção de anomalias, estes, na sua maioria, usam abordagens dependentes de dados catalogados para a sua fase de treino. O modelo desenvolvido é baseado numa estrutura Autoencoder (AE), que apresenta a capacidade de detectar anomalias precisando apenas de dados normais na sua fase de treino. Com vista a esse objectivo, foram identificados os modelos baseados em Redes Neuronais Recursivas (RNN) que têm dado evidencias de melhor desempenho no tipo de dados em estudo. Foram ainda identificadas as métricas a usar na avaliação de previsão de séries temporais. Com o objectivo de validar o modelo desenvolvido, foi feita a aquisição e teste de dados do meio industrial, utilizando o kit STEVAL-STWINKT1B.

Capítulo 2

Fundamentos Teóricos

Este capítulo, aborda conceitos teóricos subjacentes ás técnicas e modelos implementados e avaliados nesta dissertação.

2.1 Monitorização de condição e manutenção preditiva

A Monitorização Baseada em Condições (MBC) e a Manutenção Preditiva (MP) são duas estratégias que visam optimizar a eficiência de equipamentos e reduzir o tempo de paragem para manutenção e os custos durante o ciclo de vida de equipamentos geralmente em meio industrial [11].

A MBC em máquinas rotativas tem como objectivo a análise de várias grandezas com base em dados provenientes de sensores incorporados nas máquinas, como por exemplo vibrações mecânicas, afim de se identificarem potenciais problemas, como desalinhamentos ou falhas devido a fim de vida dos rolamentos [12]. As ferramentas de MBC podem mapear a degradação do equipamento quando uma análise de vibração mostra uma mudança na frequência harmónica dos componentes rotativos do equipamento. As análises no domínio da frequência podem ser baseadas nos dados obtidos de sensores de vibração ou sensores acústicos e podem ser aplicadas em equipamentos tais como: compressores, bombas e motores.

A MP é uma componente chave da indústria 4.0 que envolve a monitorização de equipamentos durante a sua operação, com o objectivo de detectar precocemente sinais que poderão indiciar falhas, podendo adicionalmente integrar modelos com capacidade de estimar o tempo de vida remanescente da máquina. O seu uso é amplamente baseado na MBC por meio de análise de vibração, por se tratar de um dos métodos mais comuns para detecção de desequilíbrios, desalinhamentos, fim de vida de rolamentos e outras anomalias em máquinas rotativas. Esta abordagem usa uma grande gama de ferramentas, como análises estatísticas e ML para prever o estado do equipamento [13].

O desenvolvimento de modelos para MP envolvem o projecto de nós dos sensores inteligentes e configuração de software embarcado em execução nos nós e no *gateway*, até ao desenvolvimento de software a ser integrado na *Cloud* ou num microcontrolador, os modelos para MP requerem uma grande variedade de competências.

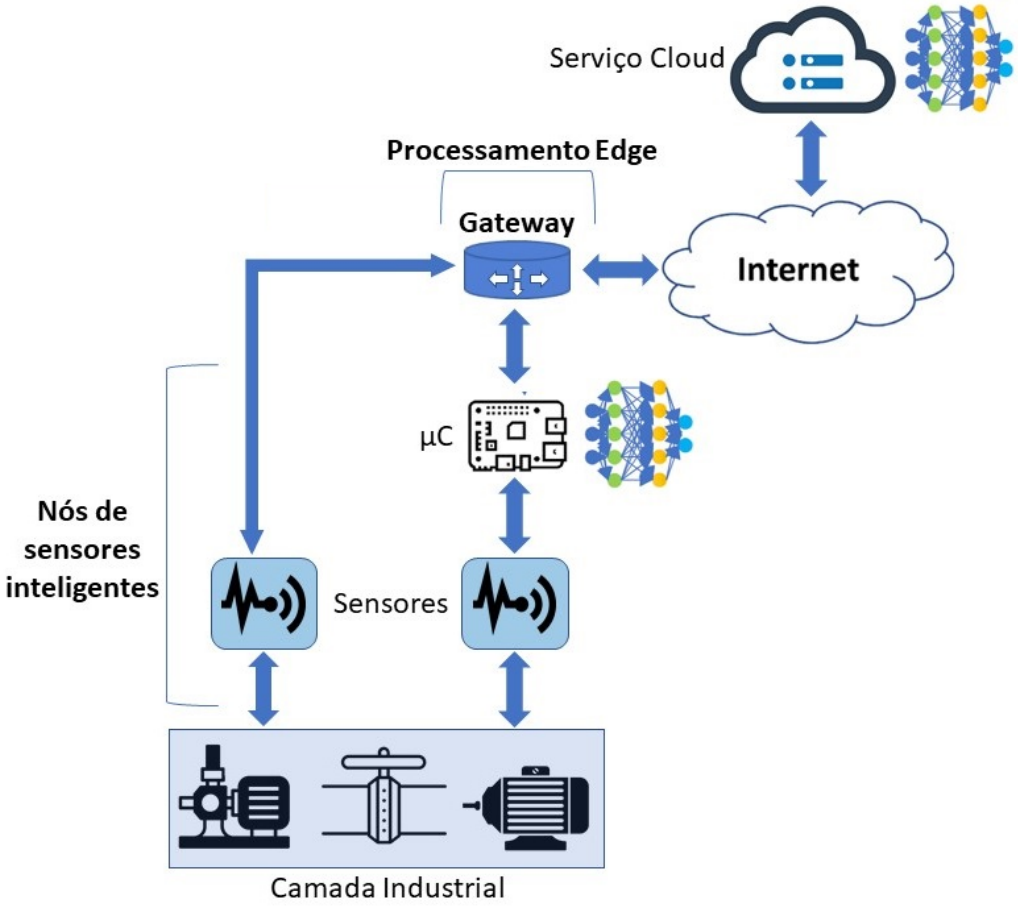


Figura 2.1: Arquitectura típica para monitorização baseada em condições e manutenção predictiva

Uma arquitectura típica de MbC e MP, como se ilustra na Fig. 2.1, integra os seguintes elementos:

Nós de sensores inteligentes que reúnem e registam dados pré-processados para visualização, para monitorização online por operadores e utilização em algoritmos de MP. Os nós de sensores inteligentes também podem processar dados e detectar anomalias reduzindo a latência computacional.

Gateways cuja função é processar e armazenar dados provenientes dos vários nós de sensores inteligentes ou para actuar como uma ponte de conectividade com a *Cloud* usando tecnologias Ethernet, Wi-Fi, rede móvel ou Low Power Wide Area Network (LPWAN).

Processamento Edge engloba todo o processamento dos dados entre nós de sensores inteligentes e *gateways* com o objectivo de, se necessário, enviar os dados para sistemas de CC, onde análises mais avançadas podem ser realizadas. O Processamento Edge também pode ser dotado da capacidade de processamento de algoritmos de ML para detecção e classificação de anomalias.

2.2 Métodos de diagnóstico de falhas

O diagnóstico de falhas e a monitorização da condição de funcionamento, têm sido estudados para prevenir interrupções indesejadas e dispendiosas no sector industrial, devido a falhas em motores eléctricos. As falhas em motores eléctricos são há muito alvo de estudo e análise, com o propósito de desenvolvimento de equipamentos mais robustos e com vidas úteis maiores. Considerando as falhas ocorridas no rotor, as mais comuns são [14]:

- Falha nos rolamentos;
- Falha no corpo do rotor;
- Desalinhamento de rolamentos;
- Desalinhamento do rotor;
- Perda de lubrificação dos rolamentos;

As principais falhas ocorridas no estator são [14] :

- Vibração na estrutura;
- Danos no isolamento;
- Falhas fase-fase do estator;
- Deslocamento de condutores;
- Falha nas ligações eléctricas;

Na análise e detecção de anomalias acima referidas, consideram-se diversos indicadores sendo alguns deles identificados na Tabela 2.1.

Tabela 2.1: Indicadores para análise de anomalias.

Eléctricos	**Mecânicos**	**Químicos**
Tensão	Vibração	Espectrometria
MCSA	Ruído	Ferrografia
Potência	Binário	XRF
	Temperatura	

Na referida Tabela 2.1, (MCSA) designa, Motor Current Signature Analysis e XRF designa *X-ray fluorescence*. Uma grande parte dos problemas mecânicos mais comuns de equipamentos rotativos, estão relacionados ao desalinhamento, desequilíbrio e fim de vida de rolamentos, sendo a análise de vibração uma ferramenta importante na detecção de ocorrências anómalas [15][16][17]. Os sensores mais comuns usados na aquição vibrações são baseados em tecnologia piezoelétrica, no entanto os Sistemas Micro-Electro-Mecânicos (MEMS) capacitivos têm ganho popularidade neste campo por razões que envolvem flexibilidade, custo e o facto de possuírem características cada vez mais próximas de sensores piezoeléctricos em termos de largura de banda e alcance.

Os MEMS apresentam vantagens comparativamente com os sensores piezoeléctricos, a saber:

- boa estabilidade ao longo do tempo e mais imunes a variações de temperatura;
- saída digital: sem necessidade de Conversor Analógico-Digital (ADC) externo ou outros circuitos de condicionamento de sinal;
- reduzidas dimensões e baixo consumo de energia;
- menor custo.

2.3 Tipos de anomalia

No âmbito de detecção de anomalias, estas podem ser distinguidas dentro de três tipos de anomalias: anomalias pontuais, anomalias de contexto ou anomalias colectivas, como apresentado na Fig.2.2.

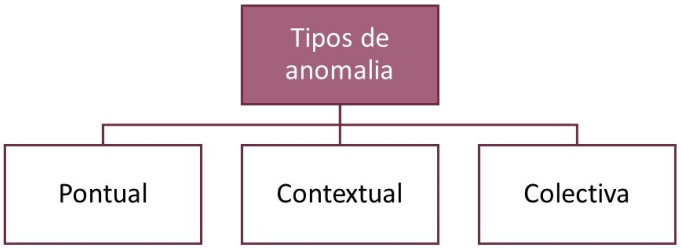


Figura 2.2: Tipos de anomalia

Anomalias Pontuais são caracterizadas como anomalias onde, numa determinada amostra, encontramos uma instância de dados individual com um comportamento fora do padrão em relação aos restantes dados. São pontos que representam um extremo, irregularidade ou desvio que ocorre de modo aleatório e sem um significado específico. Este tipo de anomalia é o mais fácil de detectar e foco da maioria das pesquisas sobre detecção de anomalias [18].

Anomalias Contextuais ocorrem quando temos valores ou conjuntos de valores que, em determinado contexto, se consideram anomalias, mas que esses mesmos valores em outras circunstâncias, são avaliados como normais, ou seja, observar certos valores em diferentes contextos pode dar resultados distintos quando se trata de classificar como anómalos ou não. São determinadas pela combinação de características contextuais e comportamentais. Em anomalias contextuais, o tempo e o espaço são as características mais utilizados [18].

Anomalias Colectivas caracterizam-se por um conjunto de pontos ser considerado anómalo se esses valores, como um conjunto, se desviarem significativamente de todo o conjunto de dados, mas os valores dos pontos de dados individuais não forem anómalos num sentido contextual ou pontual [18]. Em dados de séries temporais, uma anomalia colectiva manifesta-se aquando da ocorrência de um conjunto de dados com valores normais, mas que ocorre fora da época que se espera.

2.4 Métodos clássicos para detecção de anomalias

2.4.1 Máquina de Vectores de Suporte de Uma Classe

Máquina de Vectores de Suporte de Uma Classe (OC-SVM) são a versão não supervisionada das Máquina de Vectores de Suporte (SVM), pois são treinadas com apenas uma classe, tratando-se geralmente apenas dos dados normais [19]. Como os dados não são catalogados, as OC-SVM inferem as propriedades dos casos normais e, a partir dessas propriedades, predizem exemplos que são diferentes dos previamente entendidos como normais [20]. Dado um espaço n-dimensional F, o método OC-SVM separa todos os dados da origem através de um hiperplano e maximiza a distância do hiperplano à origem, resultando assim numa função binária que mapeia a densidade de probabilidade dos dados num plano, com regiões de alta probabilidade definidas como +1 e regiões de baixa probabilidade definidas como -1 [20][19]. A função quadrática para minimização é dada por [19]:

$$min_{\omega,\xi_i,\rho} \frac{\|\omega\|^2}{2} + \frac{1}{\eta n}\sum_{i=1}^{n}\xi_i - \rho \tag{2.1}$$

com:

$$\omega.\Phi(x_i) \geqslant \rho - \xi_i$$

$$\xi \geqslant 0$$

$$i = 1, ..., n$$

onde, n é o número de amostras, w determina a posição do hiperplano de decisão no espaço de características F, $\Phi(x)$ representa uma função não linear, ξ é uma variável responsável por prevenir o *over-fitting* e η um parâmetro regularizador de valores entre [0,1].

Através do uso de Multiplicadores de Lagrange e usando uma função de *kernel* K, para o cálculo de operações de produto escalar, a função de decisão é então definida por [19]:

$$f(x) = sign((\omega\Phi(x_i)) - \rho) = sign(\sum_{i=1}^{n}\alpha_i K(x, x_i) - \rho) \tag{2.2}$$

$\alpha_i, i = 1, ...n$ representam os multiplicadores de Lagrange. As funções de *kernel* mais usadas são: linear, polinomial, sigmoide e função de base radial (RBF).

2.4.2 Isolation Forest

O modelo Isolation Forest (IF) é um modelo não supervisionado que tem como objectivo a detecção de anomalias isolando instâncias, sem depender de qualquer medida de distância ou densidade, aliando-se a duas particularidades no que diz respeito a anomalias: representarem uma minoria numa amostra de dados (em regra) e o facto de conterem valores ou atributos significativamente diferentes dos dados considerados normais [21]. O método tem por base árvores de decisão ou também chamadas neste modelo de Árvore de Isolamento (iTree), usada para isolar instâncias Fig.2.3. No IF, sub-amostras dos dados são processadas pelas iTree baseadas em características intrínsecas dos dados. As amostras que progredirem mais pela iTree tem menos chance de ser consideradas anomalias, pois exigiram mais cortes para isolá-los dos restantes dados. Verificando-se o oposto para dados que mais facilmente e com menos iterações são isolados dos restantes, sendo assim indicados como anomalias/*outliers* [22].

O algoritmo gera recursivamente partições na amostra selecionando aleatoriamente um atributo e, de seguida, seleciona de forma aleatória um valor de divisão para o atributo, entre o mínimo e o máximo valores permitidos para esse atributo [22].

A pontuação de anomalia s numa instância x é calculado por [21]:

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}} \tag{2.3}$$

onde $h(x)$é o tamanho do caminho do ponto x, $E(h(x))$ é a media de $h(x)$ de um conjunto de iTree e $c(n)$ é o valor constante para normalizar o comprimento médio do caminho para n árvores [21], sendo $c(n)$ expresso por:.

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \tag{2.4}$$

onde $H(i)$ é estimado por $ln(i) + e$, i representa a iteração.

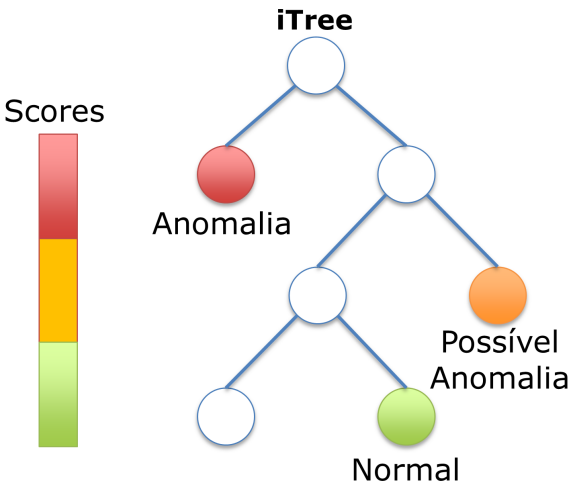


Figura 2.3: Estrutura exemplo do funcionamento de uma Itree.

2.5 Redes neuronais para detecção de anomalias

Com o exponencial aumento das velocidades de computação dos *microchip's* e a abundância de dados gerados por diferentes dispositivos Internet of Things (IoT), evidenciou-se o potencial do uso de técnicas de ML e Deep Learning (DL) em campos estatísticos.

A base destas técnicas de ML são redes neuronais artificiais. A sua estrutura base e os respectivos componentes de uma rede neuronal são ilustrados na Fig.2.4,

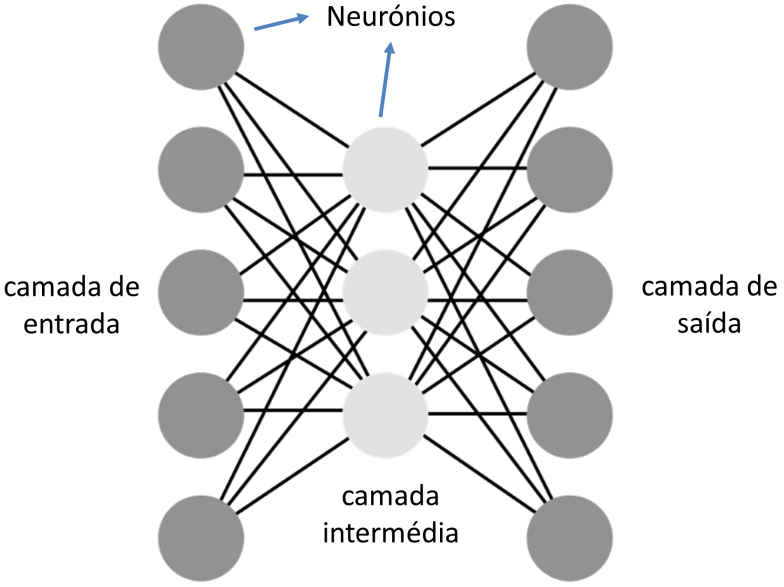


Figura 2.4: Estrutura base de uma ANN.

As ANN são sistemas de computação com nós interligados que tentam reproduzir o funcionamento dos neurónios do cérebro humano. Trata-se de um modelo abstracto onde através da ligação entre neurónios com um certo arranjo se consegue a capacidade de resolver problemas em campos como estatística, tecnologia ou economia. A peça principal deste modelo abstracto são os neurónios, também chamados de nós, que captam informação introduzidas na rede ou proveniente de outros nós e passam para os seguintes ou para a saída da rede como resultado final [23].

No que diz respeito a camadas essenciais de uma rede neuronal, estas podem ser divididas de modo simples em 3 camadas: entrada, oculta/intermédia e de saída. Os neurónios presentes na camada de entrada têm como função receber informação em forma de padrões ou sinais vindos do mundo real, como dados adquiridos por sensores. Os neurónios da rede oculta, que se situam entre a camada de entrada e de saída mapeiam padrões presentes nos dados que foram passados como entrada. Por fim, na camada de saída os neurónios transmitem informações e sinais para o mundo exterior como resultado. A saída de um neurónio pode se tornar a entrada do próximo neurónio. Dependendo do peso e do significado da conexão, a borda tem uma certa ponderação, quanto mais forte, maior será a influência que um neurónio pode exercer sobre a conexão com outro neurónio.

As abordagens baseadas em ANN's podem se dividir em dois grandes grupos, abordagens supervisionadas e não-supervisionadas. A abordagem supervisionada é caracterizada por usar conjuntos de dados previamente catalogados para o seu processo de treino e validação. Com estes dados catalogados o modelo aprende o que são considerados dados normais e dados anómalos. A abordagem não-supervisionada usa algoritmos de ML para analisar e agrupar dados provenientes de conjuntos de dados não catalogados, estes algoritmos têm como função descobrir padrões intrínsecos nos dados, sem que precise da intervenção humana.

As diferentes possibilidades de abordagens e técnicas para detecção de anomalias recorrendo a ANN são apresentadas na Fig.2.5.

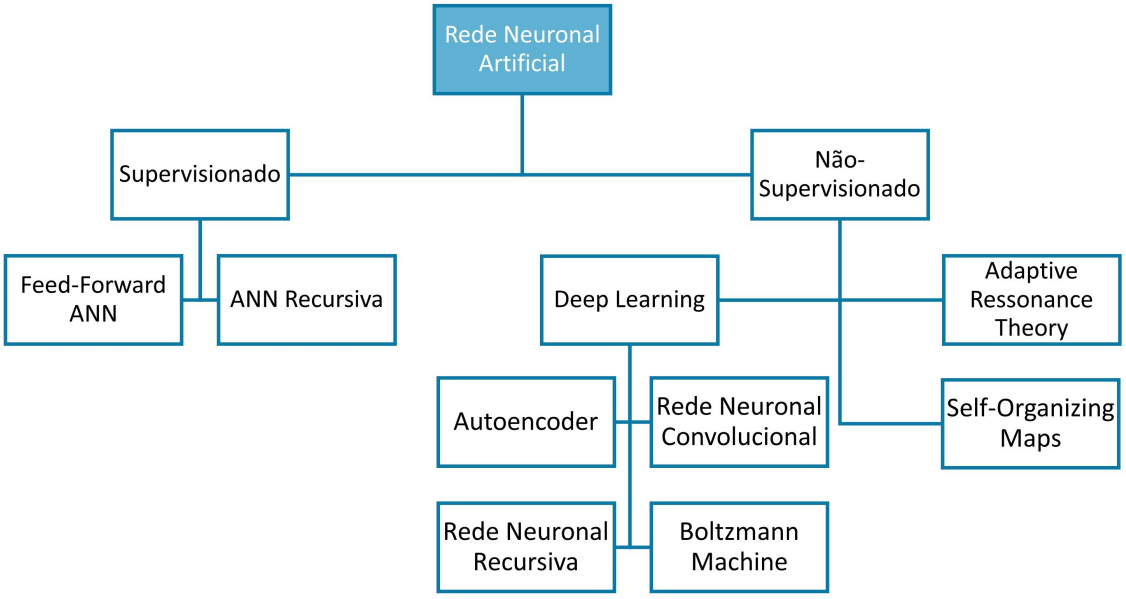


Figura 2.5: Taxonomia de ANN para detecção de anomalias. Adaptado de [1].

2.6 Redes Neuronais Recursivas

As RNN são um tipo de rede neuronal artificial com uma maior capacidade para trabalhar com dados de séries temporais ou dados que envolvam sequências. As redes neuronais *feedforward* comuns destinam-se apenas a pontos de dados, que são independentes uns dos outros. No entanto, se tivermos dados numa sequência tal que um ponto de dados dependa do ponto de dados anterior, precisamos modificar a rede neuronal para incorporar as dependências entre esses pontos de dados. As RNN têm o conceito de 'memória temporal' que recebe como entrada não apenas o dado actual, mas também a saída/estados ocultos anteriores, dispondo de um ciclo interno para o fazer, como é ilustrado na Fig.2.6.

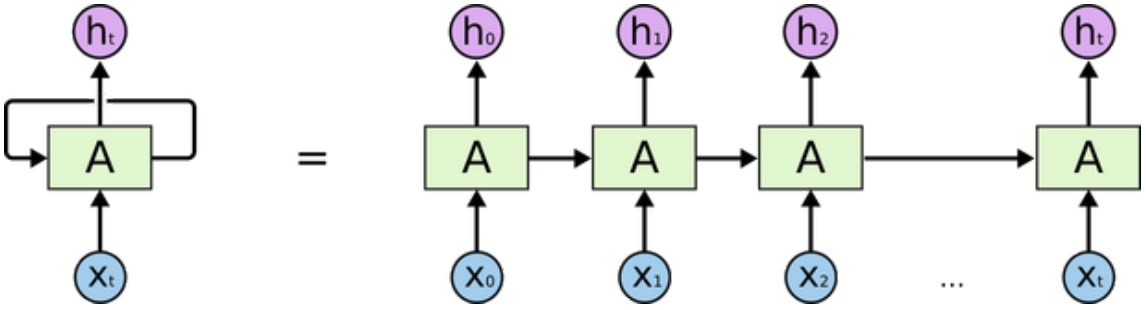


Figura 2.6: Ilustração da estrutura base de uma RNN

2.6.1 Long Short-Term Memory Neural Network

As redes Long Short-Term Memory Neural Network (LSTM) são uma evolução das RNN, introduzida por Hochreiter e Schmidhuber [24], que possuem a capacidade de aprender dependências ao longo do tempo de execução, o que evita um dos principais problemas das RNN, o problema do desaparecimento de gradiente.

As redes LSTM têm conexões de *feedback* que as tornam diferentes das redes neuronais *feedforward* mais comuns. Esta propriedade permite que as LSTM processem sequências inteiras de dados como séries temporais, sem tratar cada ponto na sequência de forma independente, retendo informações úteis sobre dados da sequência para ajudar no processamento de novos dados. Estas características tornam estas redes particularmente boas no processamento de sequências de dados como texto, sons e séries temporais.

Como nas restantes redes neuronais convencionais, as LSTM têm uma estrutura em forma de cadeia de módulos, mas no caso das LSTM, em vez de uma única camada, trata-se de uma estrutura de quatro, como demonstra a Fig.2.7, onde a interação destas confere bons resultados quando se trata de dados sequenciais.

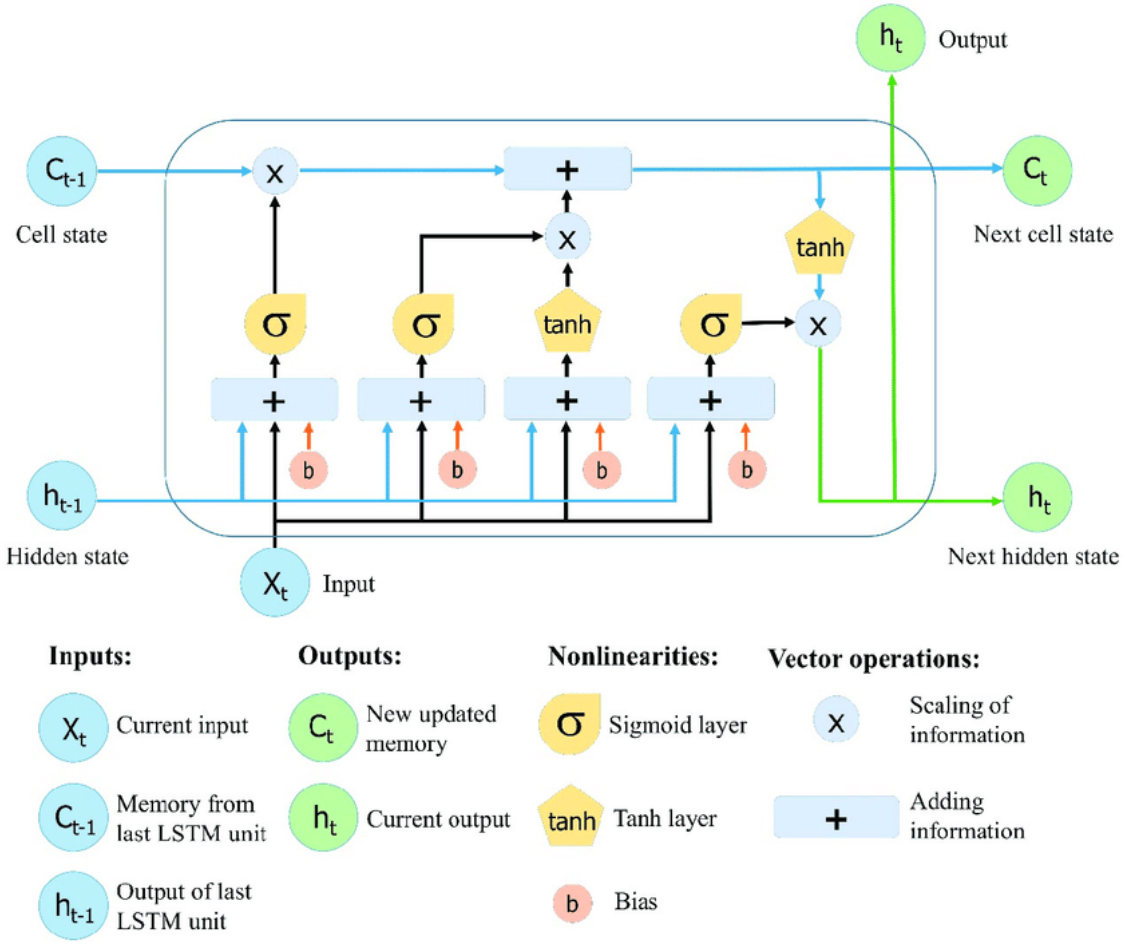


Figura 2.7: Estrutura LSTM. Imagem retirada de [2].

A estrutura básica de uma rede LSTM é composta por blocos chamados células, onde dois estados são transferidos para a célula seguinte, o estado da célula e o estado oculto. Podemos considerar o estado de célula, representado pela linha superior horizontal (azul) da Fig.2.7, C_{t-1}, como a parte principal, em que a informação ai contida pode ter interacções com as diversas estruturas contidas no restante modulo. A LSTM tem a capacidade de remover ou adicionar informações ao estado da célula, controlado por estruturas chamadas *gates*. Os *gates* são uma forma de permitir a passagem de informações, sendo compostos por uma camada de rede neuronal sigmóide e uma operação de multiplicação pontual.

A primeira etapa numa LSTM, é a identificação das informações relevantes de serem passadas ao estado da célula seguinte, essa decisão é tomada por uma camada sigmóide chamada *"forget gate layer"*, esta recebe a saída da última unidade LSTM h_{t-1} no instante de tempo $t-1$ e a entrada actual X_t no instante de tempo t e produz um número entre 0 e 1 para cada estado da célula C_{t-1}.

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \tag{2.5}$$

Onde σ é a função sigmóide, e W_f e b_f são as matrizes de peso e polarização, respectivamente.

A etapa seguinte tem como função a decisão de quais das novas informações serão armazenadas no estado da célula. Primeiro, uma camada sigmóide chamada de *input gate layer* decide quais os valores que irão ser actualizados ou ignorados (0 ou 1). Em seguida, uma camada tangente hiperbólica, aplica peso aos valores passados, decidindo o seu nível de importância (-1 a 1). Estes dois valores são multiplicados para actualizar o novo estado de célula. De seguida é feita a actualização do estado da célula passado, C_{t-1}, para o novo estado da célula C_t.

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \tag{2.6}$$

$$N_t = tanh(W_n[h_{t-1}, X_t] + b_n) \tag{2.7}$$

$$C_t = C_{t-1} f_t + N_t i_t \tag{2.8}$$

Onde, C_{t-1} e C_t são os estados de célula no tempo $t-1$ e t, W e b são o peso e o bias do estado célula.

O estado anterior é multiplicado por f_t, sendo posto de lado as informações assim decididas pelo *forget layer* e adicionado $N_t * i_t$. Estes são os novos valores candidatos, dimensionados de acordo com o quanto decidimos actualizar cada valor de estado.

Na etapa final, o valor de saída h_t é baseado no estado de célula, mas será uma versão filtrada. Esta etapa é composta por uma camada sigmóide que decide quais as partes do estado da célula que iremos produzir. Em seguida, a saída do *gate* sigmóide O_t é multiplicado pelos novos valores da camada tangente hiperbólica do estado de célula C_t, sendo um valor que varia entre -1 e 1.

$$O_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \tag{2.9}$$

$$h_t = O_t tanh C_t \tag{2.10}$$

W_t e b_o, Wo e bo são as matrizes de peso e polarização, respectivamente, da porta de saída.

2.6.2 LSTM Autoencoders

A detecção de anomalias com recurso a AE é uma tarefa com o objetivo de aprender um perfil normal de funcionamento, dado apenas por um conjunto de dados normais e, em seguida, identificar as amostras que não se encontram em conformidade com o perfil normal, identificando-as como anomalias. O uso de AE para essas tarefas segue a suposição de que um AE treinado aprenderia o sub-espaço latente de amostras normais. Uma vez treinado, isso resultaria num erro de reconstrução baixo para amostras normais e alto na reconstrução de dados anómalos.

Assim, em aplicações onde a obtenção de dados anómalos para construção de um conjunto de dados representa uma dificuldade, como é o caso de análise das vibrações de motores de meio industrial, onde só em condições anómalas de funcionamento do motor iríamos adquirir dados considerados anormais, apresenta ser um método promissor.

A motivação para usar AE é a sua capacidade de detectar anomalias com base no facto de esses valores anómalos serem raros e desviarem-se do padrão geral em dados normais. O codificador e o descodificador são os dois componentes principais da estrutura de um AE. As LSTM são um tipo de RNN que pode integrar as informações temporais na rede e mantém um vector de estado oculto que actua como uma memória para as informações do passado [25].

Um autoencoder é, na sua forma mais simples, uma rede neuronal de três camadas, incluindo um codificador, uma camada oculta e um descodificador, como ilustrado na Fig.2.8. O codificador tem como função mapear os dados de entrada de alta dimensão em códigos num espaço de baixa dimensão, e o descodificador reconstrói os dados de entrada a partir dos códigos correspondentes [26].

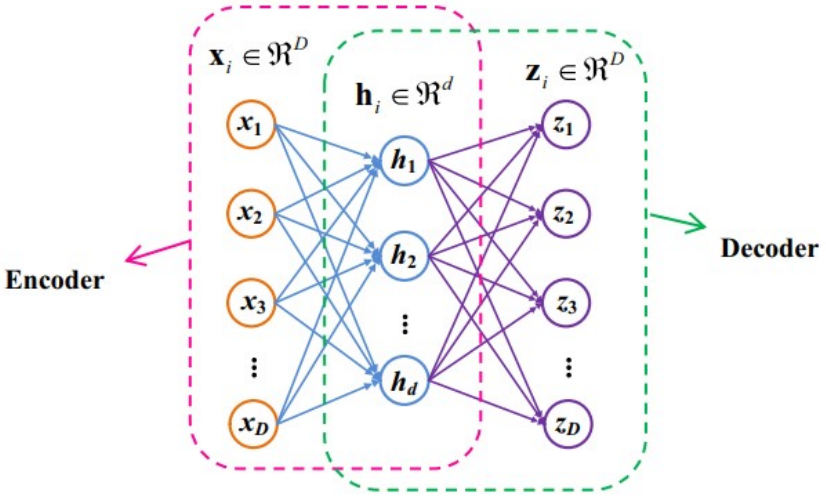


Figura 2.8: Estrutura Autoencoder [3].

Dadas as amostras de treino $x = \{x_1, x_2, x_3...x_m\}$ (para cada amostra $x_i, x_i = [x_1, x_2, ..., x_D]^T$), o codificador transforma o vetor de entrada x numa representação oculta $h = \{h_1, h_2...h_m\}$ (para cada $x_i, h_i = [h_1, h_2, ..., h_d]^T$) por meio de uma função sigmóide como:

$$\boldsymbol{h} = S_f(\boldsymbol{W}^{(1)}\boldsymbol{x} + \boldsymbol{b}^{(1)}) \tag{2.11}$$

$$S_f = 1/(1 + e^{-t}) \tag{2.12}$$

onde x e h são vectores D-dimensionais e d-dimensionais, respectivamente, $W^{(1)}$ é uma matriz de dimensão D e $b^{(1)}$ um vector de polarização *bias* de dimensão d.

Então, o vector h é transformado de volta num vector de reconstrução $z = \{z_1, z_2, z_3...z_m\}$ (para cada $z_i, z_i = [z_1, z_2, ..., z_D]^T$) pelo descodificador da seguinte forma

$$\boldsymbol{Z} = S_f(\boldsymbol{W}^{(2)}\boldsymbol{h} + \boldsymbol{b}^{(2)}) \tag{2.13}$$

onde z é um vector de dimensão D, $W^{(2)}$ é uma matriz de peso de dimensão D e $b^{(2)}$ é um vector de polarização D-dimensional.

O treino do autoencoder visa optimizar o conjunto de parâmetros $\boldsymbol{\Theta} = \{\boldsymbol{W}^1, \boldsymbol{b}^1, \boldsymbol{W}^2, \boldsymbol{b}^2\}$ minimizando a reconstrução de erro. Entre os erros de reconstrução mais usados, estão o Mean Squared Error (MSE) e Mean Absolute Error (MAE) [26],

2.7 Edge Computing

Nas últimas décadas, com a proliferação dos dispositivos IoT, a abordagem EC tornou-se num paradigma da computação que mereceu a atenção tanto da indústria como da academia, sendo uma área de grande foco de investigação e desenvolvimento. Apresenta-se como um paradigma de computação onde os dados dos dispositivos IoT são processados na periferia da rede, ou seja, na própria fonte onde são gerados e adquiridos ou o mais próximo possível da mesma. Esta proximidade traz como vantagens em relação à CC, a redução do consumo de energia, a redução da largura de banda e uma menor latência, ou seja, o tempo de resposta que é necessário ao enviar informações para serem processadas num servidor remoto [27]. Estas características tornaram esta abordagem adequada para diferentes aplicações, como automação industrial, realidade virtual, monitorização de tráfego em tempo real, casas inteligentes, monitorização marítima e de uma forma geral para análise de dados em tempo real.

Com a necessidade de realizar a inferência em dispositivos com baixo nível computacional os modelos de aprendizagem profunda a executar nesses dispositivos têm necessidade de conter certas características, como:

- Computacionalmente leves: os dispositivos de borda têm recursos limitados em termos de capacidade de computação. É primordial que os modelos de DL a implantar em dispositivos de borda sejam computacionalmente leves.

- Dimensões reduzidas: os dispositivos de borda têm capacidades de armazenamento limitadas, sendo necessário desenvolver modelos compactos e optimizados para manterem a capacidade de serem implementados em dispositivos de borda.
- Pré-treinado: os modelos devem ser treinados em dispositivos com poder computacional elevado ou na nuvem e posteriormente implementados nos dispositivos de borda para assim ser capaz de fazer inferências na borda.

Com vista a poder executar algoritmos de ML, tendencialmente pesados computacionalmente, em dispositivos de borda com menor poder computacional comparando com recursos fornecidos pela *Cloud*, várias abordagens e técnicas são usadas com esse objectivo. A forma mais simples, é o uso de modelos projectados com um número reduzido de parâmetros, para assim reduzir a memória necessária e a latência de execução. Esta abordagem tem o desafio de conseguir que modelos de dimensões reduzidas apresentem bom desempenho na tarefa a que estão destinados. A optimização de modelos de redes neuronais é outra das maneiras de tornar possível o uso dos mesmos em dispositivos de borda, procurando sempre minimizar a perda na precisão e performance que essa acção pode produzir. Os métodos usualmente mais usados para optimização de modelos de ML são: quantização e poda.

A quantização de parâmetros tem como objectivo comprimir redes neuronais através da substituição de parâmetros com virgula flutuante para parâmetros inteiros, evitando assim multiplicações com virgula flutuante, computacionalmente mais dispendiosas.

A operação de poda envolve a eliminação das conexões com neurónios menos relevantes na rede, neurónios com pesos mais baixos tornam-se descartáveis. A Fig.2.9 ilustra a operação de poda.

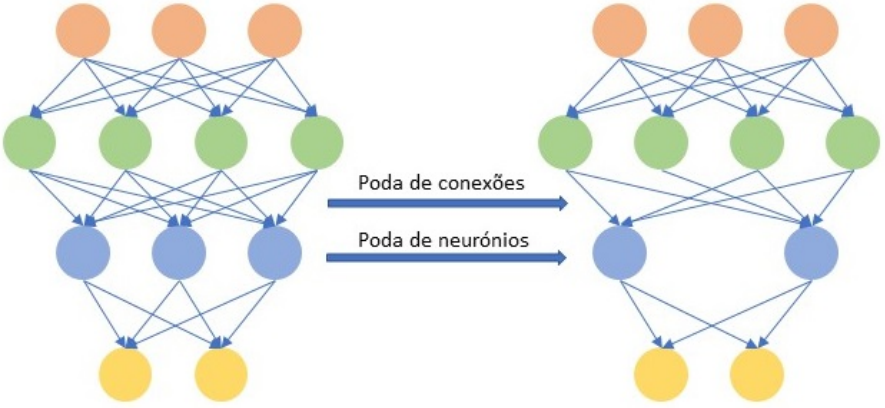


Figura 2.9: Operação de poda para optimização de redes neuronais.

2.8 Posicionamento de sensores em máquinas rotativas

Uma das partes importantes na aquisição de vibrações provenientes de máquinas rotativas para posterior análise e correlação com falhas, é o conhecimento do tipo de comportamento que estas vibrações apresentam, no que diz respeito a gama de valores e direcção em que se propagam. A leitura das vibrações provenientes de equipamentos rotativos pode ser feita através da leitura em três direcções com vista a obter mais informação de ajuda na análise de anomalias que podem ser mais predominantes numa das 3 direcções, como ilustrado na Fig.2.10. A direcção axial é a direcção paralela à linha de centro de um veio ou de uma peça rotativa, enquanto a direcção radial, é em direcção ao centro de rotação de um eixo ou rotor. A medição Tangencial é a medição tangente ou perpendicular ao transdutor radial. Em motores eléctricos as falhas passíveis de ser detectadas por vibração, são propagadas essencialmente por direcções axiais e radiais, com maior predominância em direções radiais [28].

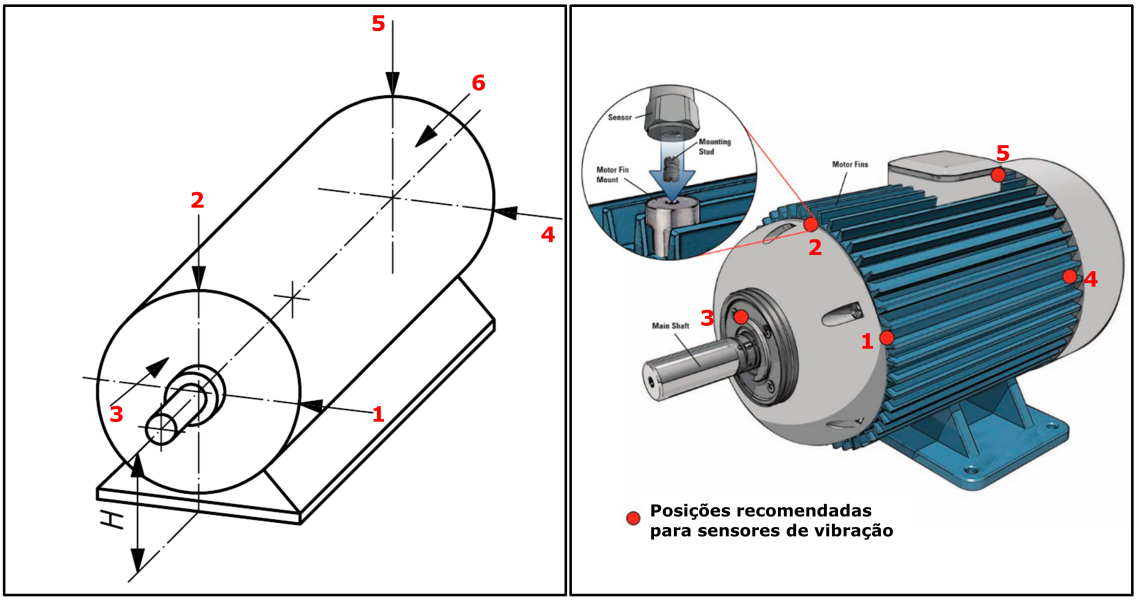


Figura 2.10: ISO 10816 [4] - posições recomendadas para medições de vibração. Posição-1 e posição-4 indicam direcções radiais horizontais e posição-2 e posição-5 indicam direcções verticais radiais, enquanto a posição-3 e posição-6 indicam as direcções axiais. Posição-1, posição-2 e posição-3 localizam-se do lado de carga, enquanto a posição-4, posição-5 e posição-6 localizam-se no lado de não carga. Adaptado de [4][5].

Capítulo 3

Estado da arte

De forma a contextualizar este trabalho com o estado da arte, são apresentados e descritos trabalhos relacionados ao tema da dissertação bem como a taxonomia das técnicas de detecção de anomalia.

No meio industrial, produtividade e rentabilidade são palavras chave. O bom funcionamento dos equipamentos industriais é vital para assegurar essa rentabilidade e produtividade. Falhas ou paragens inesperadas em máquinas industriais são dois dos principais factores de custo acrescido na industria [29]. Assim sendo, fica claro o interesse em assegurar um saudável funcionamento dos equipamentos industriais e uma maximização do seu tempo de vida.

A monitorização do estado dos equipamentos industriais, principalmente equipamentos rotativos tem vindo a ser realizado recorrendo ao estudo e análise de dados recolhidos por sensores colocados nestes equipamentos, como meio de detecção e prevenção de possíveis avarias [30]. No que diz respeito a falhas ocorridas em máquinas rotativas, 30% são provenientes de danos ou alterações nos rolamentos [31]. O uso de dados retirados dos equipamentos em análise para implementação de algoritmos de ML são usados com propósitos como diagnóstico de falhas, onde a ocorrência de falhas pode ser detectada sem que haja uma paragem e que seja preciso reparo do equipamento alvo de monitorização [32][21]. Uma abordagem mais interessante é a de previsão de falhas, onde através da monitorização de parâmetros do dispositivo são detectadas variações que podem resultar numa futura avaria, sendo assim possível fazer uma manutenção preventiva afim de evitar a ocorrência da falha e uma paragem não prevista da máquina em análise [33][34].

No que diz respeito a técnicas para detecção de anomalias existe uma grande variedade de abordagens e diferentes características importantes aquando da escolha da técnica a implementar. De seguida é mostrada a taxonomia de técnicas de detecção de anomalias, Fig.3.1.

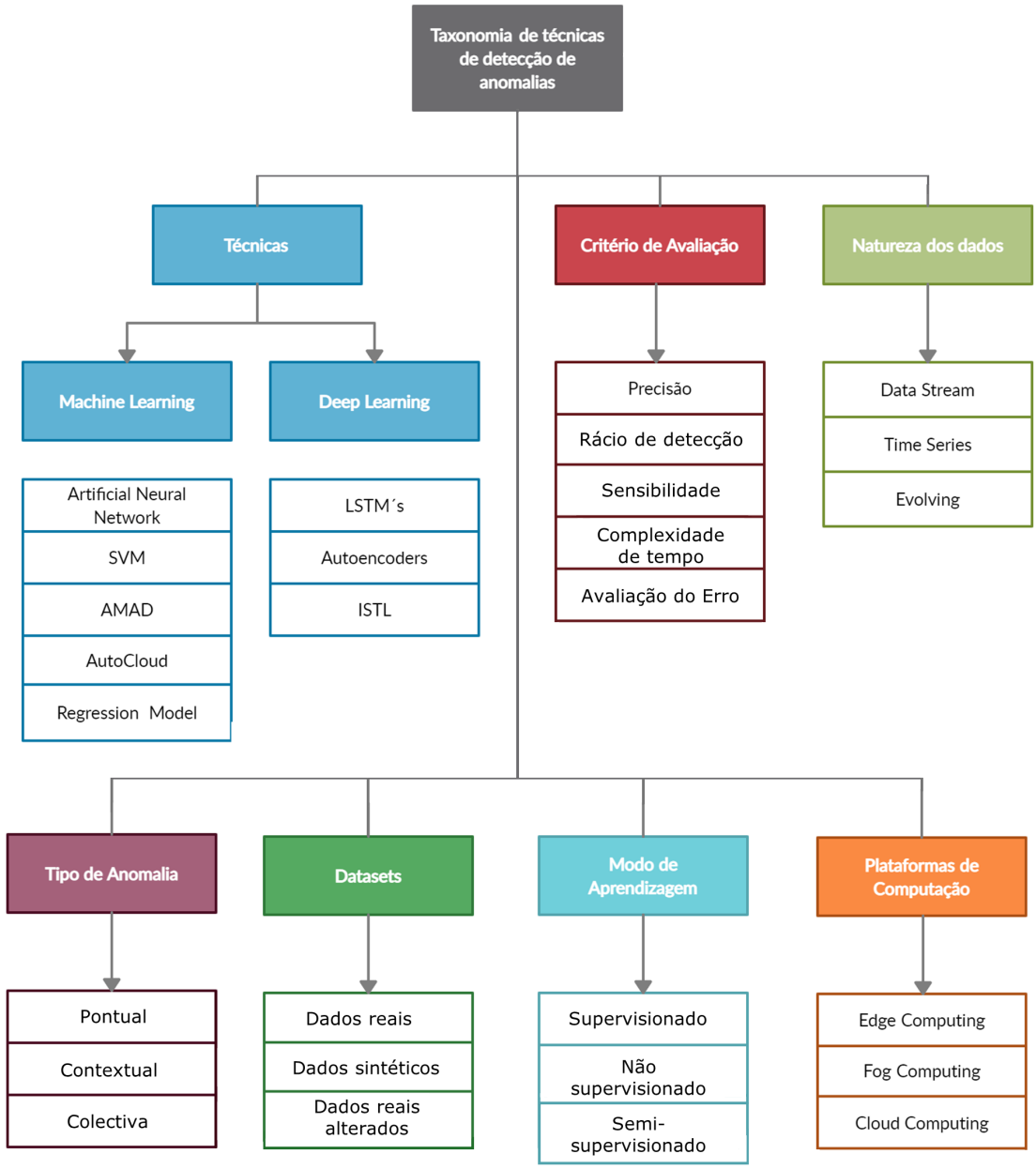


Figura 3.1: Taxonomia de técnicas para detecção de anomalia. Adaptado de [6].

O estado de funcionamento dos rolamentos em máquinas eléctricas é um dos casos de estudo para detecção de anomalias por meio de técnicas de Inteligência Artificial (AI) como modelos de DL [32] [35] [36], SVM [37], redes neuronais (RN) [38] [39], Extreme Learning Machine (ELM) [40], e AE [41] [42] [43] [44] [45].

Redes neuronais convolucionais (CNN) [46], RNN [47] e máquinas Boltzmann [48] são alguns dos métodos com grande relevância e desempenho que surgiram para o diagnostico de falhas. Particularmente, no caso de RNN, a maioria dos dados usados para o prognóstico e monitorização do estado são dados no domínio do tempo que registam mudanças nas condições mecânicas, detectadas através de vibração, temperatura ou pressão da máquina ao longo do tempo.

B. Samanta [49], fez uso das características dos sinais de vibração extraídos de uma máquina rotativa com rolamentos normais e defeituosos, tendo estes sido usados como entradas para uma ANN, consistindo numa camada de entrada, uma camada oculta e a camada de saída. A camada de entrada consiste em cinco nós, um para a raiz quadrada média, variância, assimetria, curtose e sexto momento central normalizado dos sinais de vibração. Foram usadas duas camadas ocultas com diferentes números de neurónios. A ANN é treinada usando o algoritmo de retro-propagação com um subconjunto de dados experimentais para condições conhecidas da máquina. A camada de saída consiste em dois nós binários que indicam o estado da máquina - rolamentos normais ou com defeito. A metodologia proposta tem a vantagem de realizar o diagnóstico de falhas através de um rápido modelo de treino, usando dados pré-processados e apenas algumas entradas.

Donghyun Park [50], fez uso de redes Long Short-Term Memory (LSTM) para detecção de falhas em robôs manipuladores, tendo sido usado um dispositivo de borda capaz de recolher, processar, armazenar e analisar dados usando um computador *single-board* e um conjunto de sensores de vibração, temperatura e pressão para monitorização durante a operação da máquina.

Gang Qian [51] demonstrou o grande potencial do uso de abordagens EC para análise em tempo real de dados para o diagnostico de falhas e para o controlo de máquinas rotativas, com as vantagens conhecidas do EC como, redução de tempos de resposta, menor ocupação de largura de banda e menor utilização do armazenamento. Através do uso de sinais de vibração e das correntes do motor, foi criado um *framework* em que mostrou resultados de precisão elevada para diagnóstico de falhas eléctricas e mecânicas e controlo do motor em situações de emergência.

Mais recentemente, Orrù et al.[52] aplicou um modelo de ML para previsão de falhas numa bomba centrífuga utilizada na indústria de petróleo e gás. A análise de dados é baseada em dados recolhidos de sensores incorporados na máquina. Os dados de temperatura, pressão e vibração recolhidos são pré-processados para remover ruído e codificados sucessivamente para treinar o modelo. Dois algoritmos diferentes foram usados e comparados - o SVM e o *Multilayer Perceptron (MLP)*. Os resultados deste trabalho mostraram que o modelo permite detectar adequadamente as tendências de desvios do sistema em relação ao comportamento normal de operação e gerar alertas de previsão de falhas como um sistema de apoio à decisão de manutenção.

Yi Gu [53] propôs uma abordagem baseada em DL para detecção de falhas em rolamentos. Para detectar, localizar e identificar falhas em rolamentos, foi feito uso de um AE de redução de ruído para abstrair as características dos sinais de vibração adquiridos e de seguida a característica obtida é fornecida como entrada para o classificador de rede backpropagation (BP). Neste trabalho o método proposto foi testado em cinco conjuntos de dados e comparado com métodos tradicionais usados em problemas semelhantes como Backpropagation neural networks (BPNNs), SVM, uma rede neuronal de apenas 1 camada oculta, *Extreme Learning Machine* e um modelo AE multicamadas (*Stacked Autoencoder*). O modelo desenvolvido mostrou melhores resultados no que diz respeito a performance de detecção de anomalias e classificação superior a todos os métodos em comparação.

Yang [54] apresentou o que se entende ser o primeiro trabalho usando uma rede LSTM para diagnóstico de falhas em máquinas rotativas, através de medições obtidas de sensores acoplados a um simulador do sistema de turbina eólica. Com esta abordagem expôs as potencialidades das redes LSTM de aprender dependências temporais devido ao módulo de repetição e memoria na estrutura da rede. Com a aquisição de dados da caixa de velocidades com 4 categorias possíveis, sendo elas "sem falha", "falha de dentes", "rachadura"e "dente lascado", foi efectuado o treino da rede para os diferentes estados para assim na fase de teste avaliar o desempenho da rede a catalogar os diferentes estados, tendo adquirido resultados que validam a boa capacidade de detecção de anomalias usando redes LSTM para o tipo de aplicação em estudo.

Kaji [55] fez uso da transformação de dados em bruto adquiridos por sensores de vibração, para imagens 2-D através da técnica Transformada de Wavelet Contínua (CWT). As imagens 2-D geradas são usadas como entrada de um modelo de DL, mais concretamente um AE convolucional treinado com dados relativos ao funcionamento em estado saudável, com o objectivo de criar um indicador de saúde (HI) da máquina em estudo. Semelhante aos dados de treino, o AE treinado pode reconstruir imagens com pequenas erros de reconstrução. A distância entre os dados de condição normal e o estágio de falha é medido pela fórmula da Distância de Mahalanobis e assim criado o indicador de saúde.

Sabtain Ahmad [56] propôs um método que se concentra no comportamento normal da máquina, através do uso de um modelo de LSTM-AE para monitorização de condição de máquinas rotativas e detecção de anomalias. O trabalho apresenta em paralelo com o método automático de extração de características o método IF. O modelo LSTM-AE aprende as características do normal funcionamento de uma máquina rotativa usando os sinais de vibração e assim modela o que será considerado o estado saudável da máquina. O modelo treinado, tem a capacidade de, através de sinais de vibração, monitorizar de forma autónoma a condição de máquinas rotativas e extrair características no domínio do tempo. Os resultados mostraram que o modelo LSTM-AE desempenha de forma superior na maioria dos testes de precisão e rácio de falsos positivos. Nos testes em que o método manual superou o modelo LSTM-AE, este ainda assim apresenta menor número de falsos positivos. O desempenho de métodos convencionais de detecção, como IF depende muito dos recursos cuidadosamente escolhidos o que limita a aplicabilidade geral destes métodos. Concluíram assim que o método LSTM-AE treinado apenas com os dados normais/saudáveis não só é capaz de monitorizar a condição de saúde de uma máquina rotativa em particular como pode ser generalizado para máquinas de características semelhantes. Na Tabela 3.1 é apresentado um resumo dos métodos acima descritos.

Tabela 3.1: Tabela resumo de propostas semelhantes na área de detecção de anomalia.

Artigo	**Sensor**	**Dataset**	**Pré-Process.**	**Método**	**Descrição**
[54]	binário, acelerómetro	Caixa de velocidades	Sparse FFT	LSTM	Demonstra a capacidade de redes LSTM de apreender dependências temporais em dados sequenciais para detecção de anomalias numa plataforma de teste de turbina eólica.
[55]	acelerómetro	IMS-Dataset	CWT	CAE	Faz uso da transformação de series temporais para imagens 2D (CWT) para entrada de um CAE com objectivo de construir o HI de máquinas rotativas.
[56]	acelerómetro	IMS-Dataset	Normalização	LSTM-AE	Explora o potencial de LSTM-AE para detecção de anomalias em máquinas rotativas e o seu poder de generalização comparado com métodos tradicionais como IF.

Os resultados acima apresentados, mostram as potencialidades do uso de redes LSTM para detecção de anomalias em séries temporais, e em especial em dados de vibração de motores eléctricos, representando assim uma possível opção de abordagem para o caso de estudo que se apresenta.

Capítulo 4

Ferramentas de Suporte

4.1 IMS Bearing Dataset

O IMS Bearing Dataset é fornecido pelo Center for Intelligent Maintenance Systems (IMS), da Universidade de Cincinnati, onde o equipamento de teste de rolamentos contém quatro rolamentos instalados num eixo, mantidos a uma rotação de 2000 RPM por um motor AC acoplado ao eixo por meio de correias de fricção. Uma carga radial de aproximadamente 2700 kg é aplicada no eixo e no rolamento por um mecanismo de mola [7]. Todos os rolamentos são lubrificados por um sistema de circulação de óleo que regula o fluxo e a temperatura do lubrificante.

Um íman instalado no tubo de retorno de óleo, recolhe detritos do óleo como evidência de degradação do rolamento em que, excedendo um certo limite o teste é interrompido.

Os rolamentos foram instalados no eixo conforme mostrado na Fig. 4.1. Acelerometros *Integrated Circuit Piezoelectric* (ICP) de quartzo de alta sensibilidade foram instalados na caixa do mancal (dois acelerómetros para cada rolamento [eixos x e y] para o conjunto de dados 1, um acelerómetro para cada rolamento para os conjuntos de dados 2 e 3).

Quatro termopares foram acoplados na parte externa de cada rolamento para registar a temperatura do rolamento com o objectivo de monitorizar o sistema de lubrificação. Os dados de vibração são recolhidos a cada 10 minutos por um período de 1 segundo, com uma taxa de amostragem de 20 kHz, resultando em conjuntos de dados de 20480 pontos por cada leitura [7]. Todas as falhas detectadas durante os testes ocorreram após exceder o tempo de vida projectado do rolamento, sendo superior a 100 milhões de rotações.

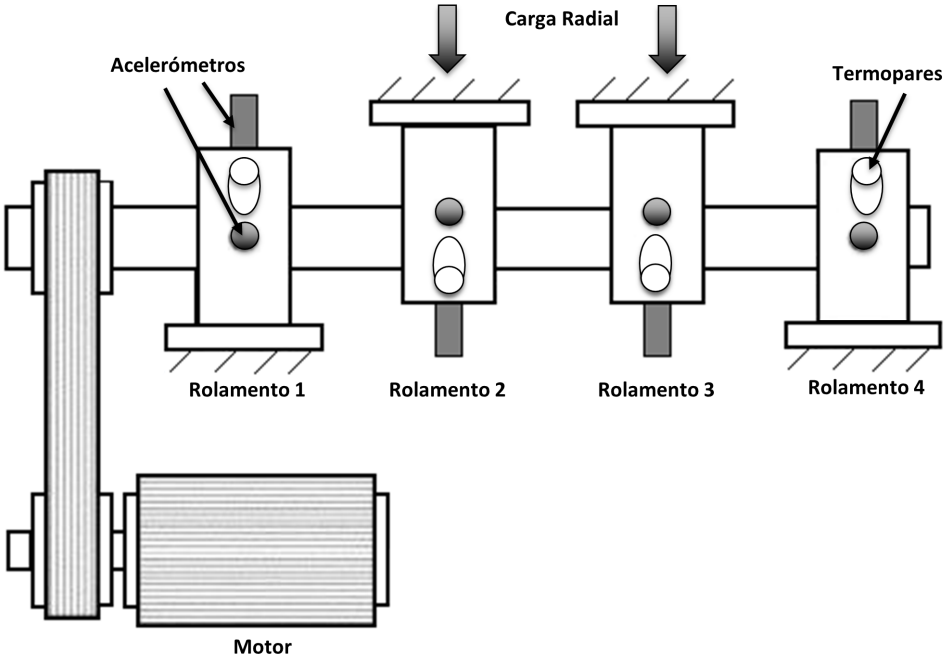


Figura 4.1: Equipamento de teste de rolamentos usado para os testes até fim de vida e aquisição dos seus sinais de vibração para o IMS-Dataset. Adaptado de [7].

Na Tabela 4.1, são apresentadas as principais caraterísticas do IMS-Dataset.

Tabela 4.1: Descrição do formato e características do IMS-Dataset

Duração da aquisição	12 Fevereiro 2004, 10:32:39 a 19 Fevereiro 2004, 06:22:39
Frequência de aquisição	1 segundo a cada 10 minutos com Output Data Rate (ODC) de 20 kHz
nº de ficheiros	984
nº de canais	4
Distribuição dos canais	Rolamento 1 – Ch 1; Rolamento 2 – Ch 2; Rolamento 3 – Ch 3; Rolamento 4 – Ch 4
Intervalo de aquisição	a cada 10 minutos
Formato do ficheiro	ASCII
Descrição	No final do teste até fim de vida, ocorre falha no rolamento 1

4.2 Kit de desenvolvimento STEVAL-STWINKT1B

O kit STEVAL-STWINKT1B, ilustrado na Fig.4.2, é um kit de desenvolvimento que permite com relativo baixo custo o desenvolvimento e teste de aplicativos industriais avançados de IoT, com foco em monitorização de condições e manutenção preditiva [57]. O kit apresenta uma variedade de sensores de nível industrial incorporados, como sensores de vibração, microfones de aplicação industrial, sensores de temperatura e humidade entre outros, associados a um microcontrolador de baixa potência. O kit é complementado com um conjunto de pacotes de software e bibliotecas de *firmware* optimizadas, bem como uma aplicação Android, todos fornecidos para ajudar a acelerar os ciclos de design para soluções de ponta a ponta [57].

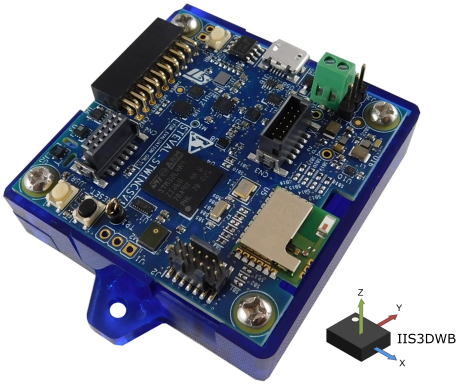


Figura 4.2: Kit de desenvolvimento STEVAL-STWINKT1B.

Deste kit de desenvolvimento, destacam-se as seguintes características principais:

- ARM Cortex-M4 MCU 120 MHz com FPU, 2048 kbytes memória Flash (STM32L4R9)
- Capacidade de acolher Cartão SD para armazenamento de dados
- *Bluetooth low energy* v5.0 tecnologia wireless e Wi-Fi (com a adição da placa STEVAL-STWINWFV1), RS485 e conectividade USB OTG
- Dispõem ainda de uma ampla gama de sensores industriais de IoT:
  - Sensor de vibração de 3-eixos digital (IIS3DWB)
  - Acelerómetro 3D + 3D Gyro iNEMO unidade de medida inercial (ISM330DHCX) com core ML
  - Sensor de movimento de alta performance MEMS (IIS2DH)
  - Magnetómetro de 3-eixos de baixo consumo (IIS2MDC)
  - Sensor de pressão absoluta digital (LPS22HH)

- Sensor de humidade relativa e temperatura (HTS221)
- sensor de temperatura local digital de baixa tensão (STTS751)
- Microfone MEMS de aplicação industrial (IMP34DT05)
- Microfone MEMS analógico com resposta em frequência até 80 kHz (IMP23ABSU)

4.2.1 Pacote de funções FP-SNS-DATALOG1

O pacote de funções FP-SNS-DATALOG1 disponibilizado para o kit de desenvolvimento 4.2, fornece uma solução abrangente para guardar dados de qualquer combinação de sensores e microfones configurados até a taxa de amostragem máxima de 20 kHz. Os dados dos sensores podem ser armazenados num cartão micro SD (Secure Digital High Capacity - SDHC) formatado com o sistema de arquivos FAT32 ou transmitidos para um PC via USB (classe WinUSB) usando o software complementar fornecido para Windows e Linux. O FP-SNS-DATALOG1 permite configurar a placa via arquivo .JSON, bem como iniciar e controlar a aquisição de dados. Os comandos podem ser enviados de um host via interface de linha de comando. O aplicativo permite ainda o controlo via Bluetooth através da aplicação disponível "ST BLE Sensor"que permite gerenciar as configurações da placa e do sensor, iniciar/parar a aquisição de dados no cartão SD, entre outras funcionalidades. Para a leitura e análise dos dados do sensor adquiridos usando o pacote de funções FP-SNS-DATALOG1, são fornecidos scripts em Python e Matlab.

4.3 TensorFlow

O TensorFlow é uma plataforma de código aberto para ML que visa reunir e facilitar a implementação de técnicas de ML e DL, operando em larga escala e em ambientes heterogéneos [58][59]. Faz uso de gráficos de fluxo de dados, onde os nós no gráfico representam operações matemáticas, enquanto as arestas do gráfico representam as matrizes de dados multi-dimensionais (tensores) que fluem entre eles. Esta arquitetura flexível permite que os algoritmos de ML sejam descritos como um gráfico de operações conectadas [59], podendo ser treinados e executados em GPUs, CPUs e TPUs de várias plataformas sem reescrever código, desde dispositivos portáteis a servidores de alto desempenho.

4.4 TensorFlow Lite

O TensorFlow Lite é uma estrutura de DL de código aberto, que converte um modelo pré-treinado no TensorFlow num formato especial para microcontroladores que pode ser optimizado a nível de velocidade de computação e/ou armazenamento. O formato gerado pode, se necessário, sofrer ainda operações de optimizaçao como quantização e poda e ser implementado em dispositivos de borda como telemóveis ou microcontroladores para processamento e inferência. O processo desde conversão do modelo até ao processo de inferência é representado na Fig.4.3.

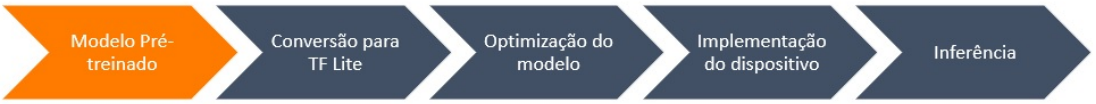


Figura 4.3: Processo de conversão de um modelo Tensorflow Lite. Adaptado de [8].

Após a obtenção de um modelo pré-treinado, quer seja com função de classificação, detecção de anomalias, entre outros, é feita a conversão do modelo pré-treinado para uma versão do Tensorflow Lite (.tflite), este formato assegura uma boa eficiência comparado com o modelo não convertido e representa uma versão mais leve que ocupa menos espaço, o que torna os modelos .tflite uma opção para trabalhar em dispositivos móveis/borda [8].

No que diz respeito à optimização do modelo e apesar da sua conversão por si só já trazer vantagens para a implementação dos modelos em microcontroladores, esta simples conversão pode não ser suficiente para se conseguir um bom funcionamento no que diz respeito a latência e capacidade computacional geralmente mais baixas dos microcontroladores. Para isso poderão ser usadas operações como quantização e poda para que estes modelos de DL ocupem menos espaço nos dispositivos de borda e necessitem de menos memória, o que se traduz num modelo com maior capacidade de rápidas inferências.

4.5 Keras

A Keras é uma biblioteca desenvolvida em Python que potencializa o desenvolvimento de soluções baseadas em abordagens DL para detecção de anomalias, classificação, condução autónoma, entre outras aplicações. Disponibiliza modelos e blocos de construção de redes neuronais frequentemente usados, como camadas, funções de activação, optimizadores e uma variedade de ferramentas para facilitar o trabalho com dados de imagem, texto e series temporais, simplificando o código necessário a ser desenvolvido para se obter e testar modelos de DL [60].

4.6 Scikit-learn

Scikit-learn é uma biblioteca baseada em Python que visa disponibilizar ferramentas e modelos de ML dos mais recentes aos mais tradicionais [61]. É uma ferramenta capaz de integrar soluções de classificação, regressão, *clustering*, pré-processamento, entre outras actualmente usadas em diversas áreas cientificas [62]. A biblioteca foi criada de modo a interagir com pacotes numéricos e científicos centradas nas bibliotecas Python amplamente usadas, NumPy e SciPy.

4.7 DARTS

Darts é uma biblioteca de ML em Python desenvolvida para trabalhar com series temporais, dispondo de um conjunto de ferramentas e modelos que vão dos clássicos como ARIMA até aos mais recentes modelos de DL para previsão. Oferece funcionalidades como suporte a séries multidimensionais, meta-aprendizagem em séries múltiplas, treino em grandes conjuntos de dados, incorporação de dados externos, conjuntos de modelos e suporte avançado para previsão probabilística [63]. A biblioteca possui um tipo de variável, **TimeSeries**, que se caracteriza por ser um array tridimensional da forma *(time, component, sample)* que representa uma série temporal, onde *component* representa a dimensão da série multivariável, e *sample* representa amostras de séries temporais estocásticas [63]. Além dos modelos de previsão mencionados tem ao dispor modelos de pré-processamento como filtro de Kalman e filtro Gaussiano e métricas para validação dos modelos testados.

4.8 Avaliação de técnicas de previsão e medidas de precisão

A área de previsão de dados tornou-se de grande interesse pelos resultados promissores em várias áreas científicas e industriais. Com o evoluir das técnicas de previsão, estudos foram realizados com o objetivo de identificar os métodos mais precisos para previsão de séries temporais, sendo que o desempenho dos métodos de previsão varia de acordo com a medida de precisão utilizada [64][65]. Uma boa medida de precisão deve fornecer um resumo informativo e claro da distribuição de erros, confiabilidade, validade de construção, complexidade computacional, independência de escala, sensibilidade a mudanças e interpretabilidade. É sugerido por muitos trabalhos, que nenhuma medida isolada pode ser superior a todas as outras nestes critérios [66][67].

As métricas apresentadas de seguida servem para avaliar o erro entre os valores obtidos da previsão e os valores reais da serie temporal, onde F_t é o valor da previsão no tempo t e Y_t o valor real da serie temporal no tempo t. Na avaliação de desempenho das diferentes métricas quando menor o valor obtido do erro mais próxima a previsão se encontra do valor real da serie temporal.

Mean Absolute Error (MAE)

A métrica MAE, Erro médio absoluto, indica-nos o valor em média do erro que esperamos encontrar ao longo da previsão. Esta métrica não indica o tamanho relativo do erro, o que pode dificultar na interpretação do que se trata de erros elevados ou erros baixos. É menos sensível a grandes desvios do que perdas quadráticas. Para colmatar este possível problema torna-se útil o uso da métrica no seu estado percentual, Mean Absolute Percentage Error (MAPE).

$$MAE = \frac{1}{n}\sum_{t=1}^{t=n}|F_t - Y_t|$$

Mean Absolute Percentage Error (MAPE)

A métrica MAPE, Percentagem do Erro médio absoluto, indica-nos o valor percetual da média do erro que esperamos encontrar na previsão. Como apresenta no seu dominador o valor Y_t apresenta problemas quando o valor é zero ou valor muito elevado. Valores extremos mas pouco frequentes não são severamente penalizados.

$$MAPE = \frac{1}{n}\sum_{t=1}^{t=n}\frac{|F_t - Y_t|}{Y_t} * 100\%$$

Mean Squared Error (MSE)

A MSE definida como a média dos erros quadráticos, incorpora tanto a variância (a dispersão dos valores previstos entre si) quanto o *bias* (a distância do valor previsto ao valor real). Tratando-se de uma métrica quadrática, penaliza erros elevados ou outliers em relação a pequenos erros. Apesar de resolver problemas que encontramos na métricas MAE e MAPE como os valores extremos ou valores nulos (zero), apresenta desvantagem quando aplicada a pequenas quantidades de dados.

$$MSE = \frac{1}{n}\sum_{t=1}^{t=n}(F_t - Y_t)^2$$

Root Mean Squared Error (RMSE)

Trata-se da raiz quadrada da MSE que apresenta como vantagem uma maior facilidade de interpretação pois o valor da Root Mean Squared Error (RMSE) apresenta-se na mesma unidade de medida do valor previsto. Pode ser comparado ao MAE para determinar se a previsão contém erros elevados, mas pouco frequentes.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{t=n}(F_t - Y_t)^2}$$

Normalized Root Mean Squared Error (NRMSE)

Calculada através da normalização da RMSE, tendo dois métodos tradicionais de calcular, usando a média ou a gama de valores reais. É geralmente usada para comparar diferentes conjuntos de dados ou diferentes modelos de previsão que têm diferentes escalas.

$$NRMSE = \frac{RMSE}{\bar{Y(t)}}$$

Weighted Absolute Percentage Error (WAPE)

A métrica Weighted Absolute Percentage Error (WAPE) é útil pois aplica uma "pesagem", que ajuda a distinguir os erros menores dos erros maiores, ou seja, evita que os valores de erro pequenos sejam considerados iguais ou superiores aos valores maiores.

$$WAPE = \frac{1}{\sum_{t=1}^{n} Y_t}\sum_{t=1}^{t=n}|F_t - Y_t|$$

O uso da média da métrica WAPE pode ser útil para dados de baixo volume, onde cada observação tem prioridade diferente ao avaliar modelos de previsão. As observações com prioridade mais alta têm um valor de peso mais alto.

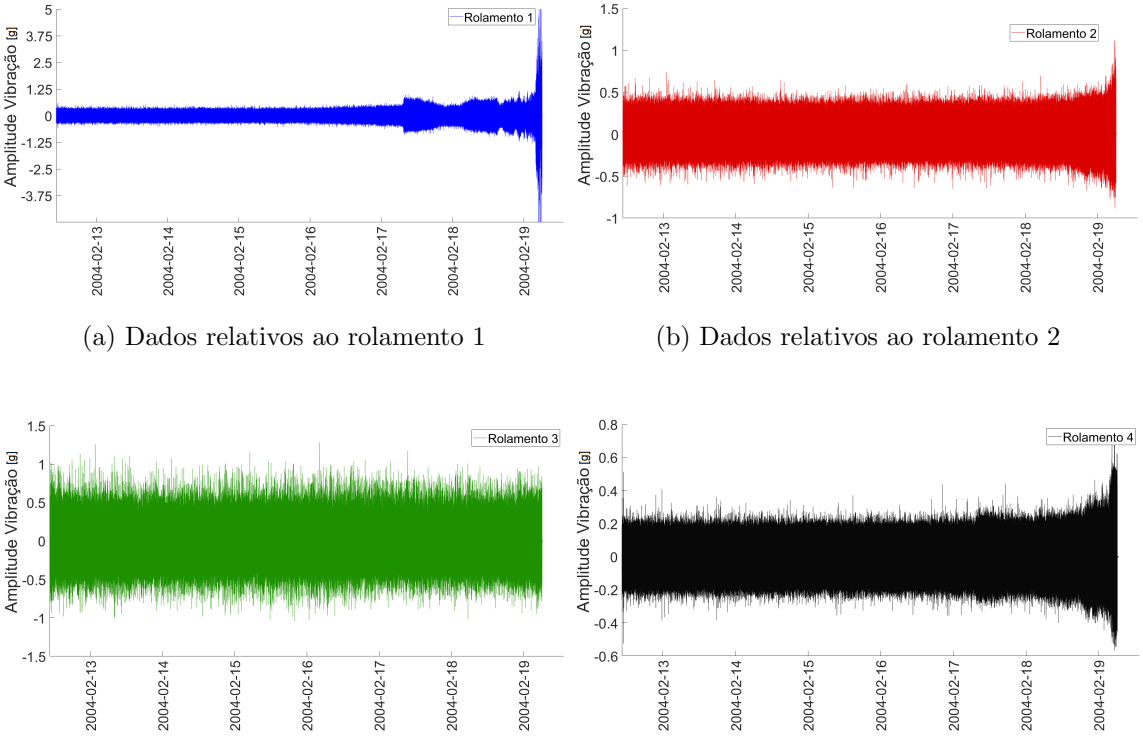
Capítulo 5

Trabalho Desenvolvido

No seguinte capítulo são descritos e detalhados os passos executados para o desenvolvimento da abordagem proposta, como o processamento dos dados, a escolha dos parâmetros e hiperparâmetros do modelo, testes no IMS-Dataset e o desenvolvimento de software para aquisição e tratamento de dados em meio industrial.

5.1 Processamento de Dados

De forma a se entender o perfil e comportamento dos dados relativos ao IMS-Dataset é feito o seu tratamento e visualização. Como descrito na Secção 4.1, é facultada a informação que o rolamento 1 apresenta avaria, mais concretamente o seu fim de vida. Devido ao formato em que os ficheiros são disponibilizados no IMS-Dataset, onde o nome do ficheiro representa a data em que foram adquiridos os dados foi necessário fazer a sua leitura e conversão através de código Python com vista a preservar essa informação, de extrema importância na análise que se pretende executar. De seguida, e devido às dimensões dos dados disponíveis (superiores a 20 milhões), foi feita a sua análise e observação através do MatLab. Sendo de seguida apresentados os resultados na Fig.5.1.



(a) Dados relativos ao rolamento 1

(b) Dados relativos ao rolamento 2

(c) Dados relativos ao rolamento 3

(d) Dados relativos ao rolamento 4

Figura 5.1: Dados em bruto relativos aos 4 rolamentos presentes no equipamento de teste do qual foi adquirido o IMS-Dataset. De notar as diferentes escalas entre figuras devido ao perfil dos dados em análise.

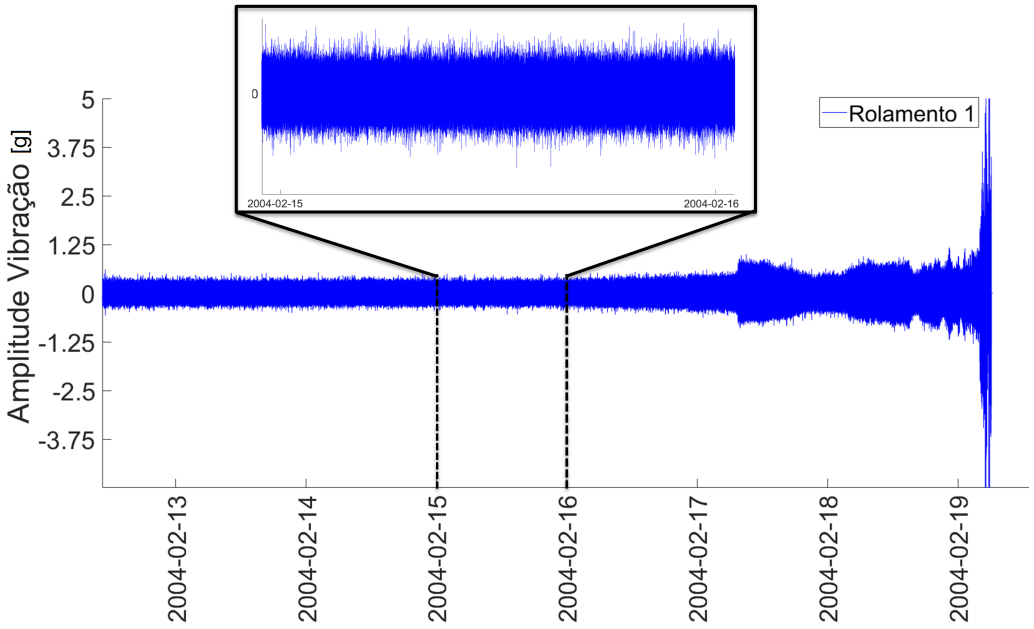


Figura 5.2: Dados em bruto relativos ao rolamento 1 do IMS-Dataset.

Por observação da Fig.5.1, é possível constatar que apesar de se evidenciar na parte final dos dados variações que possivelmente são consequência de falha do rolamento, fim de vida do mesmo, é difícil por observação concluir em que momento começa a anomalia em causa. Analisando a Fig.5.2, devido ao perfil do sinal, muita informação que à priori aparenta ser de comportamento normal do rolamento, pode apresentar já indícios de uma anomalia. Essa percepção do que é referente a dados normais e anormais é importante para a futura divisão dos dados em dados de treino e de teste. Assim sendo, com o objectivo de melhor visualizar as informações intrínsecas do conjunto de dados foi realizado o seu processamento através do Desvio Padrão Médio Absoluto (MAD). O processamento MAD de um conjunto de dados é a distância média entre cada ponto e a média da amostra, com isto temos uma percepção sobre a variabilidade do conjunto de dados em análise. O MAD de uma série temporal $x_1, x_2, ..., x_n$ é calculado com,

$$MAD = \frac{1}{n} \sum_{i=1}^{n} |x_i - \bar{x}| \tag{5.1}$$

onde x_i é o valor actual, $\bar{x}$ a média e n a dimensão da série temporal. O resultado desta operação é demonstrado na Fig.5.3.

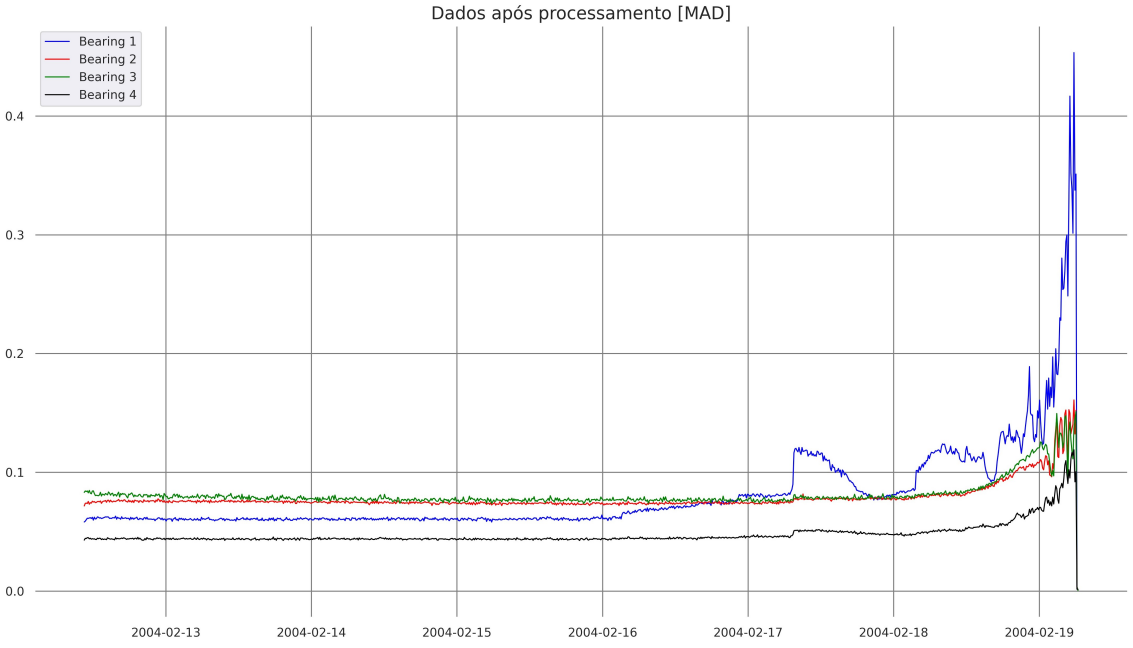


Figura 5.3: Dados relativos ao pós-processamento MAD dos 4 rolamentos do IMS-Dataset

Devido às características do sistema do qual foram adquiridos os dados do IMS-Dataset, documentado na Secção 4.1 e por observação da Fig.5.3, importa assegurar que os valores anómalos detectados no rolamento 1 (documentado como sendo o que apresenta falha) não influenciam a leitura e por consequência a detecção de anomalias nos restantes rolamentos, assim optou-se por fazer toda a análise isolada do rolamento 1. Procedeu-se ainda à remoção dos últimos valores medidos por se tratarem de valores de paragem de medição (leitura errática) e ainda a um limite máximo dos valores para evitar saturação, tendo sido escolhido o valor 0.45 como limite máximo, como demonstra a Fig.5.4.

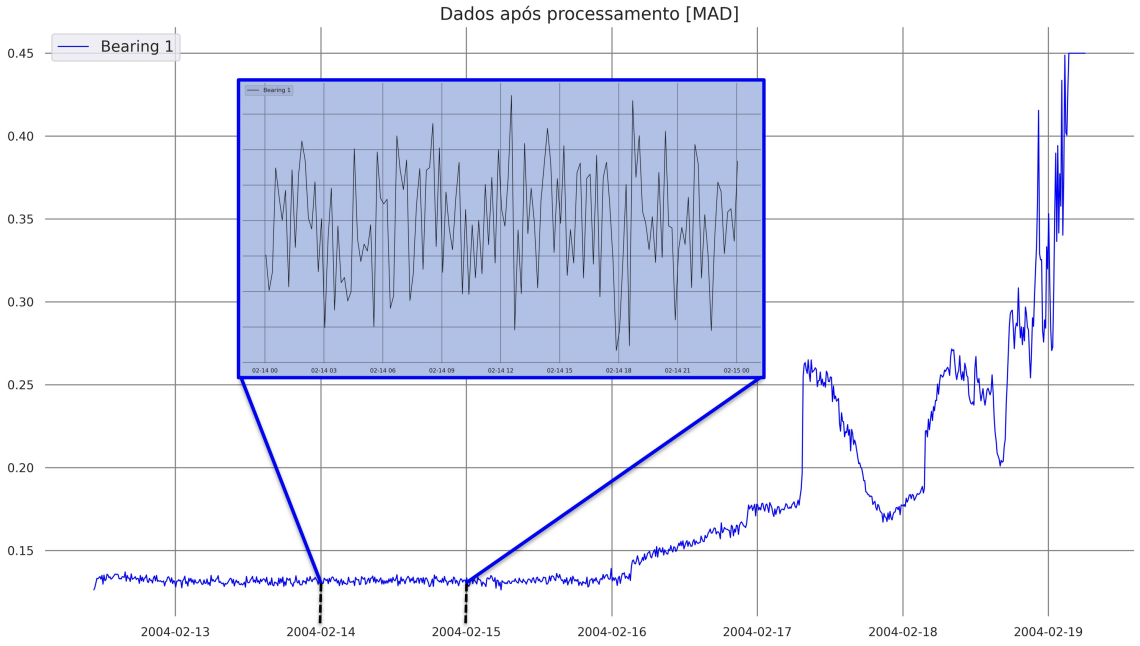


Figura 5.4: Dados relativos ao pós-processamento MAD e normalização dos dados do rolamento 1, com remoção dos valores finais e um limite máximo de 0.45 para evitar saturação

Apesar das informações disponibilizadas do IMS-Dataset indicarem como sendo o rolamento 1 a apresentar problemas de fim de vida, não é detalhado com exactidão o momento em que ocorre a leitura de valores referentes à anomalia, para isso, importa por observação dos dados processados escolher o que serão entendidos como dados normais e como dados de teste/anómalos.

Na Fig.5.5 são mostrados os dados em bruto relativos ao rolamento 1 e as divisões escolhidas dos dados para análise.

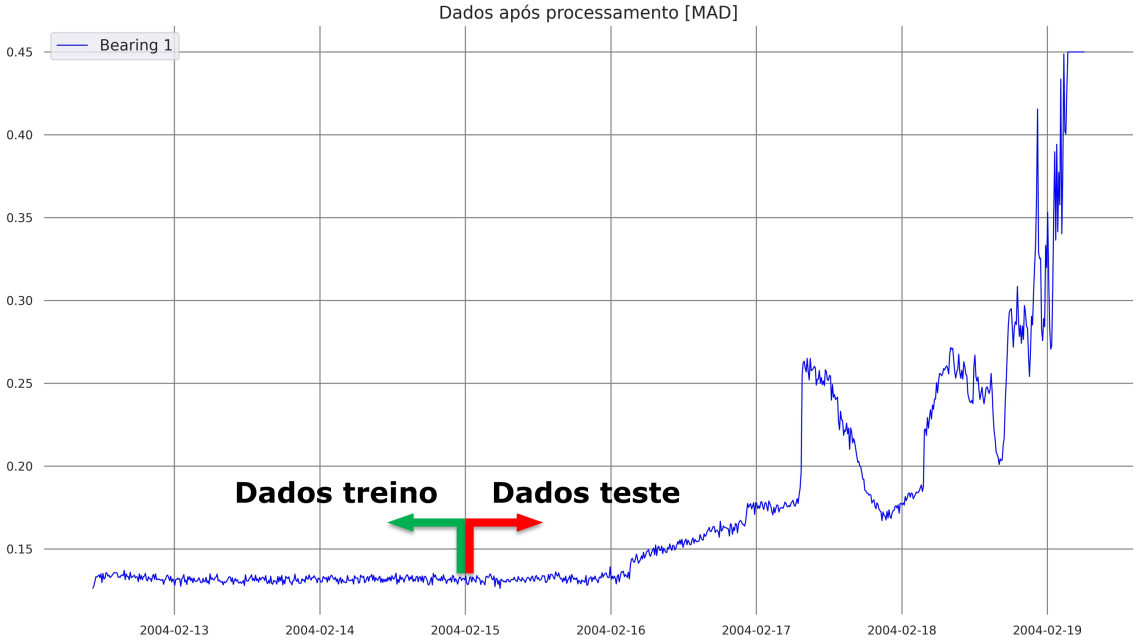


Figura 5.5: Escolha de dados de treino e dados de teste

Por observação dos dados após processamento, é possível detectar uma variação perceptível após dia 16, tendo em conta esta informação, a escolha relativamente aos dados considerados normais serão os dados até dia 15, sendo os restantes considerados dados de teste.

5.2 Testes de previsão com diferentes modelos

Com os recursos da ferramenta Darts, descrita na secção 4.7, foram elaborados testes de previsão da série temporal em estudo através de modelos recursivos e caracterizados por apresentarem bom desempenho em análise de séries temporais, como RNN, RNN-LSTM, RNN-GRU e TCN. Os testes em questão fazem uso dos dados relativos a meio dia anteriores à data marcada como inicio da previsão para a componente de treino do modelo. Com vista a aumentar o desempenho de previsão dos modelos, ao invés da previsão ser feita para 24 horas numa só previsão, é feita a previsão dos dados em janelas deslizantes de 2 horas o que equivale no nosso estudo a 12 pontos. Os dados de previsão foram concatenados produzindo uma série temporal de dados representativa das 24 horas que se pretende fazer previsão. Nos resultados seguintes são mostrados os testes para previsão dos dados de 1 dia (144 pontos) com EPOCH=150. O resultado das previsões dos modelos acima referenciados é apresentado na Fig.5.6.

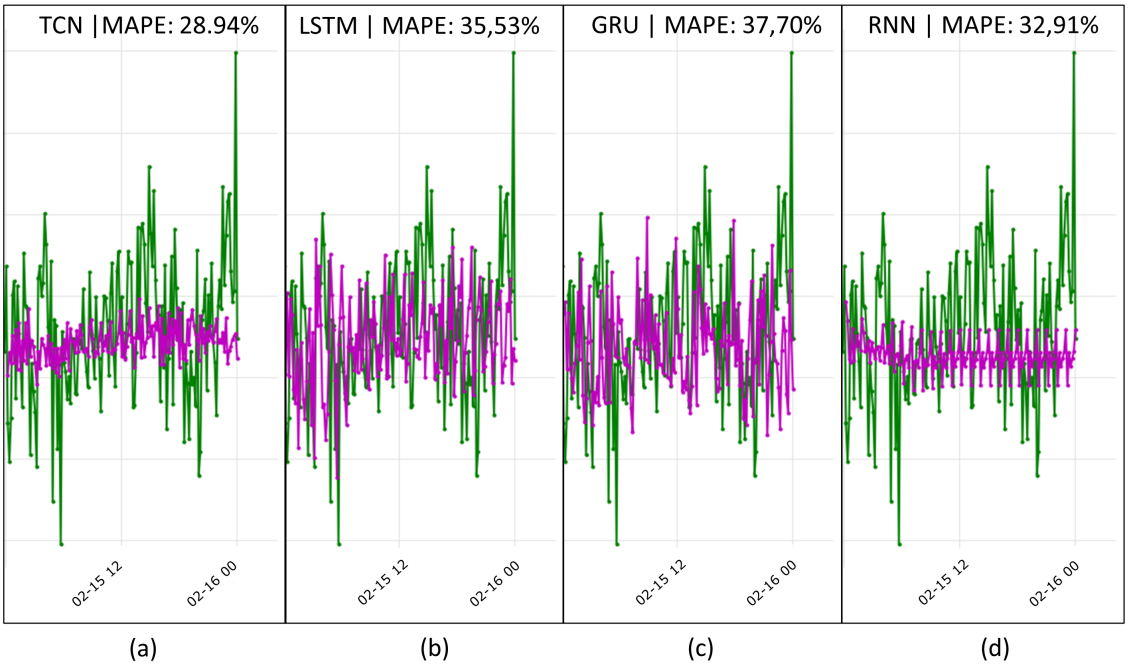


Figura 5.6: Previsões das 24h do dia 15 dos diferentes modelos em teste: (a) TCN; (b) LSTM; (c) GRU; (d) RNN

Por análise do resultado das previsões dos modelos e tendo como avaliação da previsão a métrica MAPE, observa-se que apesar dos modelos TCN e RNN apresentarem melhores resultados por avaliação da métrica, não têm a capacidade de seguir de forma tão fiel o perfil da série temporal em análise, como os modelos LSTM e GRU.

Importa assim fazer uma análise mais extensa de diferentes métricas de avaliação de desempenho dos modelos de previsão em estudo. As séries temporais originais e de previsão serviram como entrada para uma função de cálculo de algumas das principais métricas de avaliação de modelos de previsão [68], sendo os resultados apresentados na Fig.5.7 e Fig.5.8.

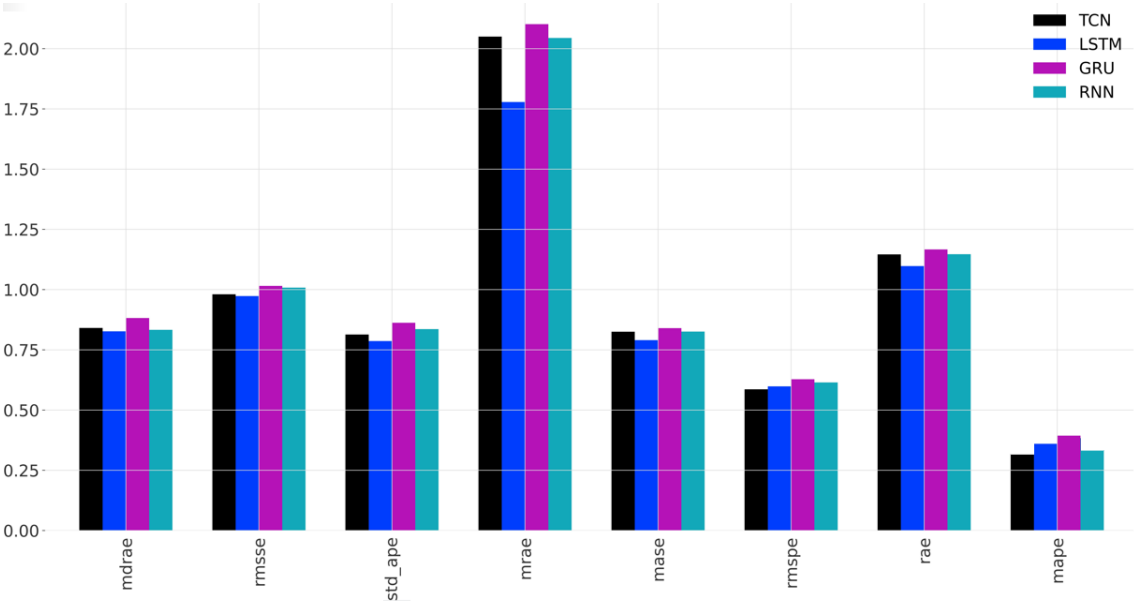


Figura 5.7: Avaliação de desempenho dos diferentes modelos

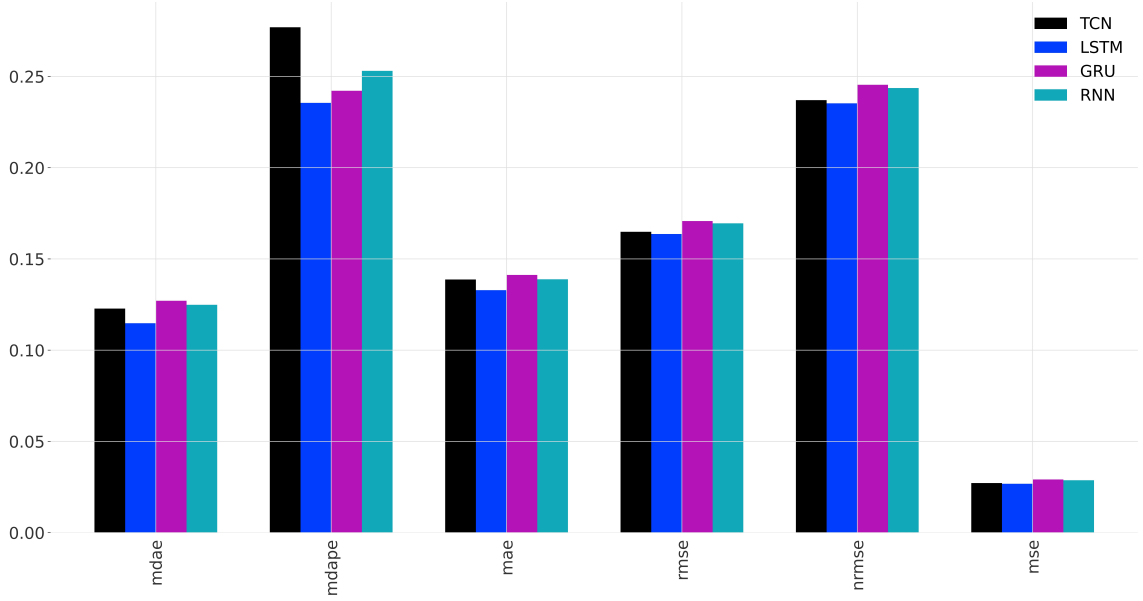


Figura 5.8: Avaliação de desempenho dos diferentes modelos

Com os resultados obtidos da função de avaliação dos modelos de previsão [68], verifica-se um desempenho superior de forma geral do modelo LSTM, ainda que de forma pouco significativa em algumas das métricas em análise. É demonstrada assim a capacidade das redes baseadas em camadas LSTM para propósitos de detecção de falhas em séries temporais, como descrito nos trabalhos apresentados no Capítulo 3. Os resultados apresentados são úteis para a escolha da métrica usada como função de perda do modelo AE, apresentado de seguida na Secção 5.3.

5.3 Modelo LSTM-AE

Com os resultados obtidos nos testes realizados na Secção 5.2 e nos trabalhos apresentados no Capítulo 3, a estrutura de AE será baseada em camadas LSTM, com vista a ser construido um modelo AE-LSTM capaz de detectar anomalias em dados de vibração provenientes de máquinas rotativas. A estrutura base de uma estrutura AE é ilustrada na Fig.5.9.

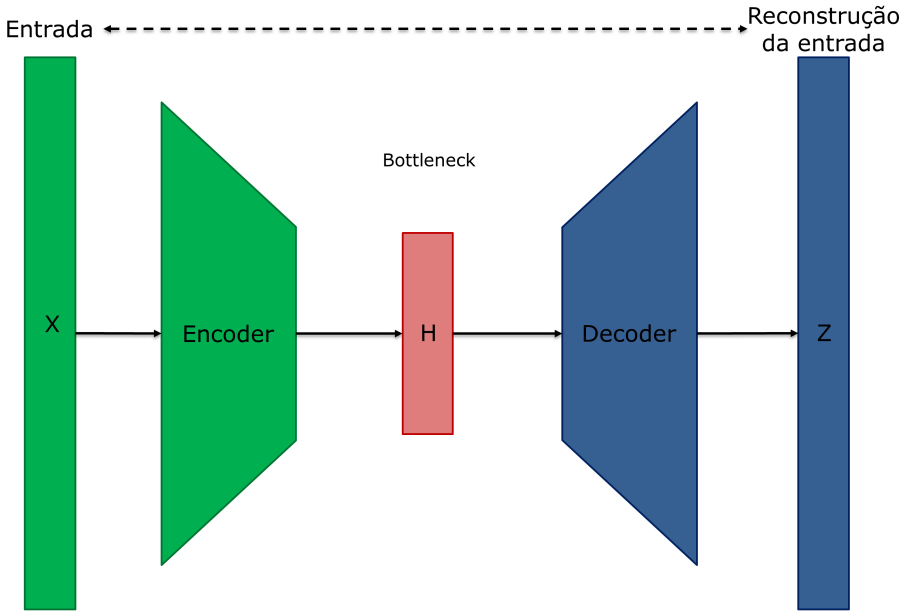


Figura 5.9: Ilustração base de uma arquitectura de Autoencoder.

Para construção do modelo proposto para análise de séries temporais, através da ferramenta Keras descrita na Secção 4.5. Importa numa primeira etapa proceder à transformação dos dados em respeito à entrada de um AE-LSTM. A entrada para uma camada LSTM é um tensor 3D *[samples, timesteps, feature]* onde *samples* define o número de amostras de entrada do modelo que dispomos, *timesteps* define a dimensão do sub-conjunto de dados que se pretende fornecer à estrutura LSTM, e *feature* o número de características em análise presentes na nossa serie temporal. No caso em questão e como apresentado na Secção 5.1, pretendemos utilizar uma *feature*, os valores de vibração do rolamento 1. Após a estrutura dos dados de entrada da rede estar concluída importa escolher uma estrutura adequada para o modelo, isto é, a configuração da sua estrutura e hiperparâmetros como, o número de camadas, o número de unidades na camada, o tipo de função de activação, o regularizador, o seu optimizador de compilação, *learning rate* e a função de perda.

No que diz respeito à escolha da função de perda e tendo em conta a análise das métricas de avaliação de métodos de previsão apresentada na Secção 4.8, foi feito uso da função de perda disponível na biblioteca Keras e com melhor desempenho nessa mesma análise, tendo a escolha recaído sobre a métrica MAE.

Tendo em vista a quantidade de diferentes configurações e parâmetros possíveis de configurar importa fazer uso de ferramentas de auto-parametrização fornecidas pela biblioteca Keras.

O KerasTuner é uma estrutura de optimização de hiperparâmetros configurável e fácil de usar para resolver a questão da procura dos melhores hiperparâmetros para o modelo em estudo [69][70]. Os hiperparâmetros são as variáveis que governam o processo de treino e a topologia do modelo de ML. Essas variáveis permanecem constantes ao longo do processo de treino e impactam diretamente o desempenho do modelo. Os hiperparâmetros podem ser divididos entre dois tipos [71]:

- Hiperparâmetros do modelo que influenciam a dimensão do modelo, como o número e a largura das camadas ocultas;
- Hiperparâmetros do algoritmo que influenciam a velocidade e a qualidade do algoritmo, como o *learning rate*.

A biblioteca de auto-parametrização dispõe de algoritmos de pesquisa como RandomSearch, Otimização Bayesiana e Hiperbanda para encontrar os melhores valores dos hiperparâmetros do modelo em teste. Considerando o tipo de estrutura AE que se pretende criar, os parâmetros que se pretende configurar são apresentados na seguinte tabela 5.1,

Tabela 5.1: Tabela com os hiperparâmetros em teste.

Hiperparâmetro	**Valor**
camadas	[4, 6]
neurónios/*units*	[4,.., 216], step = 2
função de activação	relu, sigmoid, softmax, tanh, elu
learning rate	0.01, 0.001, 0.0001

As funções de activação são responsáveis por ajudar o modelo a aprender padrões complexos presentes nos dados em análise. Estas funções ajudam a decidir que dados são relevantes, e então passados aos neurónios seguintes, impedindo os modelos de se tornaram simples modelos de regressão linear. Algumas das funções de activação mais usadas são apresentadas na Fig.5.10,

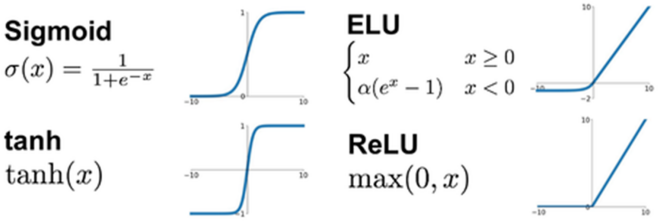


Figura 5.10: Funções de activação para modelos ANN

Na análise dos melhores hiperparâmetros e restantes variáveis do modelo proposto, importa não só a configuração de parâmetros do modelo que apresentem melhor desempenho no que diz respeito ao valor da perda, mas também a sua estabilidade, com o objectivo de na fase de teste do modelo com os dados de teste, diminuir ao máximo a possibilidade de ocorrência de falsos positivos devido a grandes oscilações nos valores de perda. A métrica de desempenho dos diferentes testes de hiperparâmetros são a análise das curvas de perda na fase de treino e validação e a sua suavidade.

De seguida, na Fig.5.11, são apresentados alguns dos resultados das curvas de perda, com os testes de hiperparâmetros realizados através da ferramenta KerasTuner.

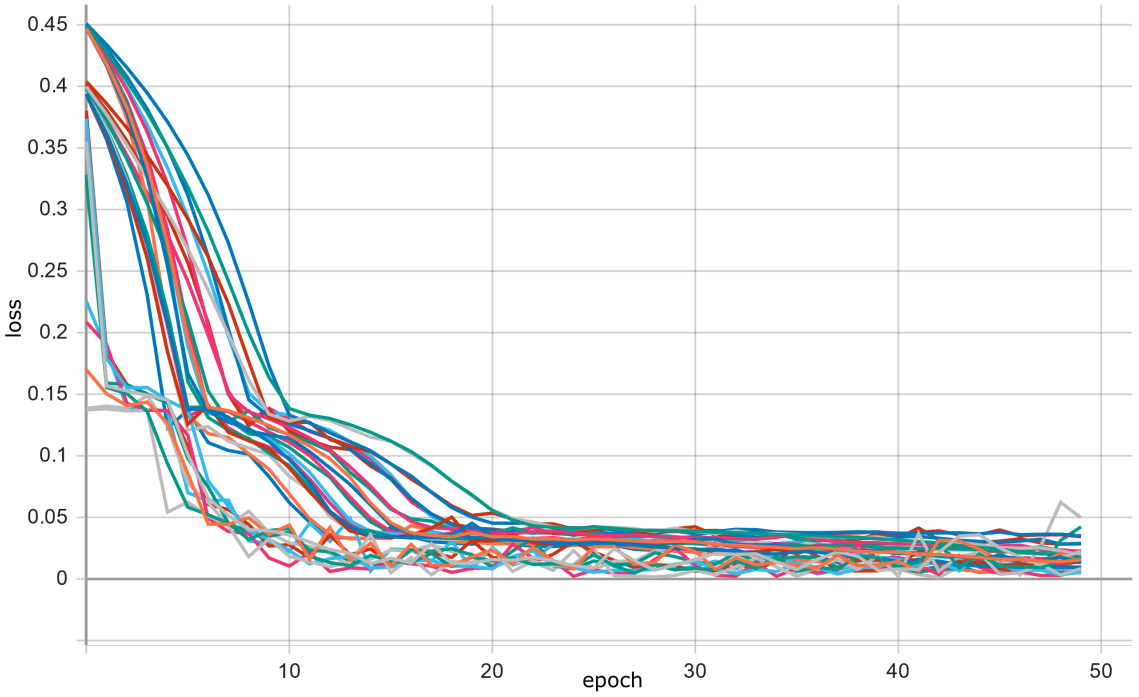


Figura 5.11: Gráfico ilustrativo das curvas de perda na fase de treino e validação dos diferentes hiperparâmetros testados.

De forma a avaliar o comportamento das curvas de perda apresentadas na Fig.5.11, tão importante quanto o valor da perda, interessa no caso de modelos de reconstrução que esta perda seja estável, sem grandes mudanças de valores que prejudiquem a fase de teste do modelo.

Na Fig.5.12, são apresentados dois casos que realçam este facto. A execução número 001 apresenta um valor de perda menor, mas mais inconstante quando comparado ao da execução 013, traduzindo-se num modelo pior no que diz respeito à sua fase de teste de dados.

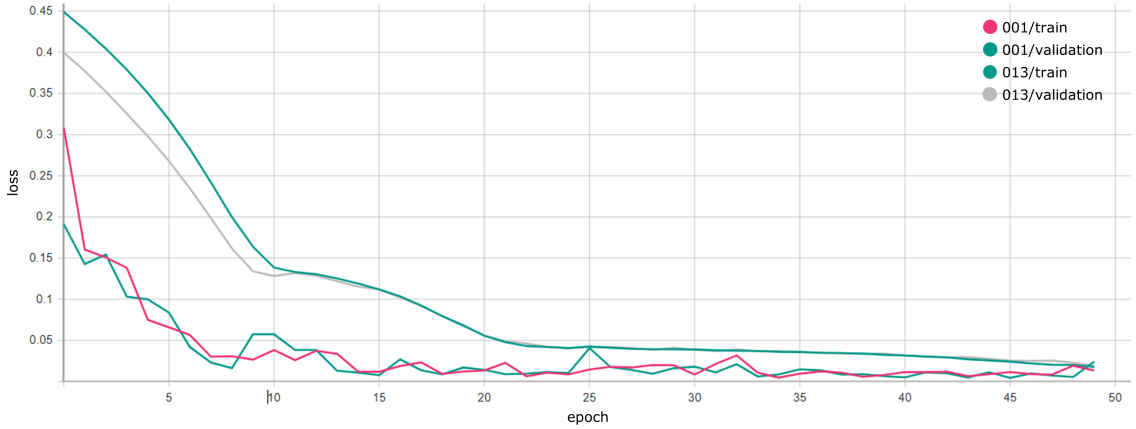


Figura 5.12: Gráfico ilustrativo das curvas de perda na fase de treino e validação dos diferentes hiperparâmetros testados.

Tendo em consideração o comportamento pretendido nas curvas de perda para escolha dos parâmetros e hiperparâmetros do modelo, os resultados dos parâmetros selecionados são apresentados na Tabela 5.2 e Tabela 5.3.

Tabela 5.2: Parâmetros da rede após avaliação de desempenho.

Camada	**Kernel units**	**Função de activação**
Lstm1	32	ReLu
Lstm2	4	ReLu
Lstm3	4	ReLu
Lstm4	32	ReLu

Tabela 5.3: Parâmetros do compilador após avaliação de desempenho.

Compilador	
Learning rate	0.01
Optimizador	Adam
Função de perda	MAE

A escolha da função de perda foi baseada nas métricas disponibilizadas na biblioteca Keras em concordância com os resultados da análise realizada em 5.2, tendo recaído sobre a função de perda MAE por ser a que apresentou uma maior valorização de redes baseadas em LSTM. A estrutura do modelo após escolha dos parâmetros e hiperparâmetros é apresentada na Fig.5.13.

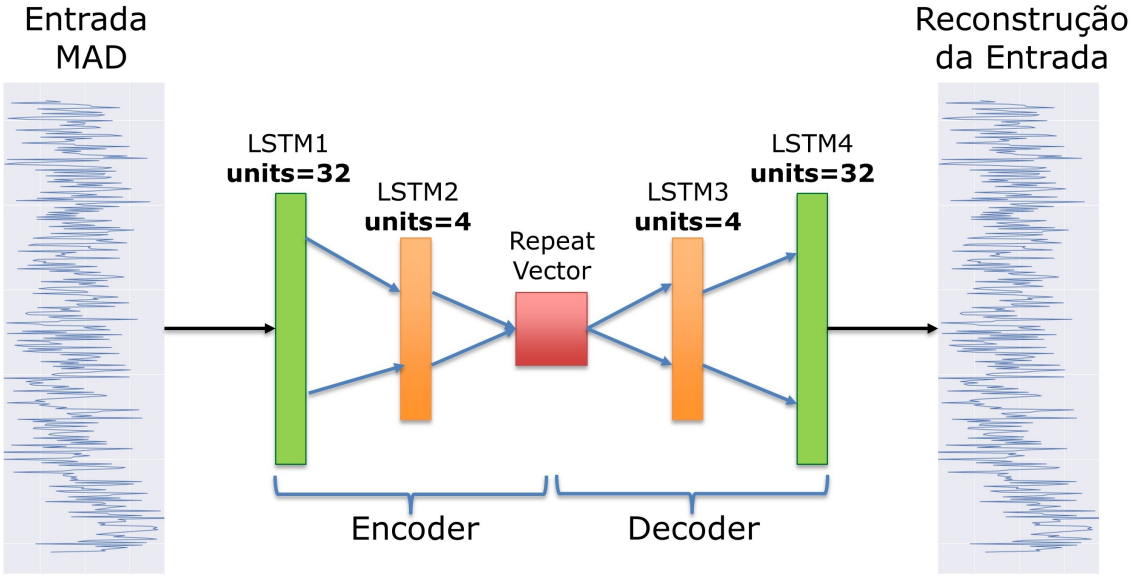


Figura 5.13: Arquitectura modelo proposto após escolha dos seus parâmetros e hiperparâmetros

Após a escolha de todos os parâmetros e hiperparâmetros procedeu-se ao treino do modelo, tendo sido usado um *batch size* de 25, este hiperparâmetro define o número de amostras a serem trabalhadas antes de actualizar os parâmetros internos do modelo. Terminada a fase de treino e validação é feita a curva de distribuição dos erros na fase de treino para uma melhor percepção do peso de cada erro nos dados estabelecidos como normais.

O *threshold* do nosso modelo é escolhido através da curva do erro de reconstrução do modelo em dados normais, como sugere a Fig.5.14.

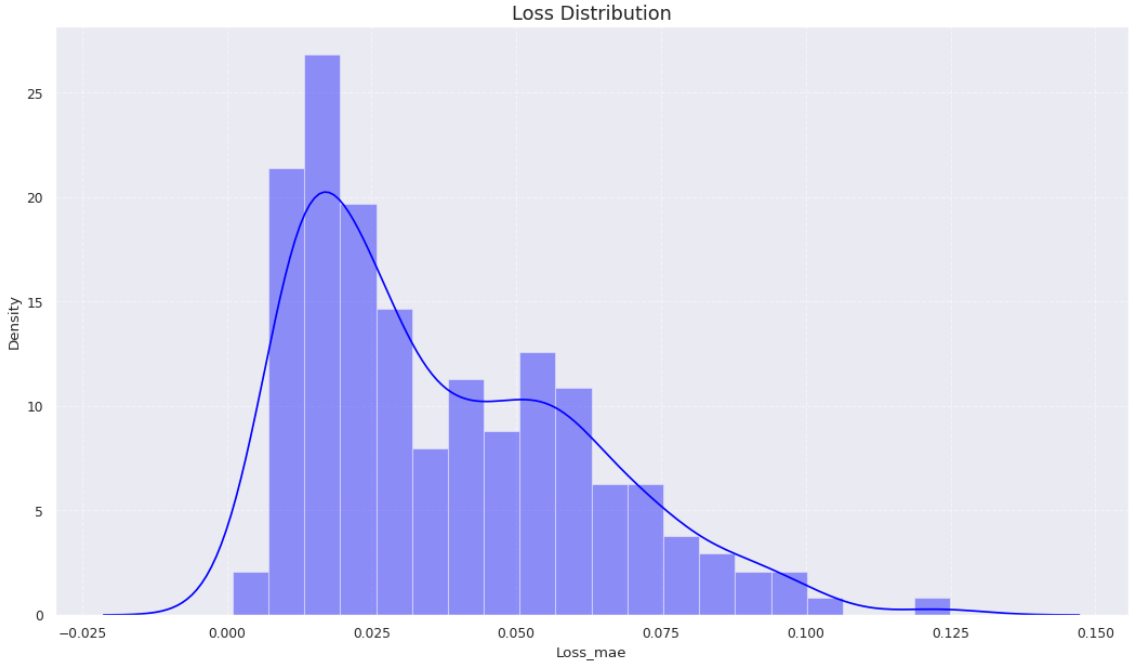


Figura 5.14: Distribuição do erro de reconstrução do modelo em dados normais e validação

Com o objectivo de manter o modelo com relativa flexibilidade a variações, o *threshold* é calculado de forma automática através do valor máximo de erro de reconstrução durante a fase de treino, acrescido de uma tolerância de 15%. Este limite definido é responsável pela detecção de dados anómalos assim que o erro de reconstrução do modelo ultrapasse esse valor.

O modelo pré-treinado foi convertido para formato .tflite compatível com dispositivos de baixo poder computacional, através da biblioteca TFLite, descrita na Secção 4.4. O modelo convertido apresenta dimensões inferiores a 70 KB, munindo assim o modelo da capacidade de poder ser implementado em dispositivos de borda de baixo poder computacional e com restrições no que diz respeito a memória RAM e ROM.

5.4 Aquisição de dados em meio industrial

As bombas disponibilizadas pela empresa de aquacultura ao qual se pretende validar o modelo, Fig.5.15, são bombas de recirculação de água em sistemas de criação de peixes em fase prematura e bombas do sistema de regulação de temperatura da água. Tratam-se de bombas em sistemas vitais da indústria da aquacultura, onde apesar do uso de bombas suplentes para obtenção de redundância no sistema, interessa ainda assim implementar modelos para aumento da sua fiabilidade e compreensão do comportamento ao longo da vida útil.

Figura 5.15: Bombas em meio industrial de aquacultura responsáveis pelo controlo de temperatura da água em tanques de desenvolvimento de peixe.

Os instrumentos tradicionais para adquirir vibrações são baseados em tecnologia piezoelétrica, mas MEMS capacitivos têm ganho popularidade neste campo por razões que envolvem flexibilidade, custo e o facto de possuírem características cada vez mais próximas de sensores piezoeléctricos em termos de largura de banda e alcance.

Como descrito na Secção 4.2, o kit de desenvolvimento de que dispomos possui um acelerómetro de 3-eixos do tipo MEMS, que nos permite uma resposta em frequência ate 6 kHz, o que abrange grande parte dos problemas de um motor no que diz respeito à análise por vibração [28]. O acelerómetro de 3-eixos IIS3DWB do tipo MEMS, possui características apropriadas para aplicações industriais de monitorização de vibrações, sendo as suas principais características:

- Sensor de 3-eixos com saída digital;
- Diferentes escalas configuráveis de medição: $\pm 2/\pm 4/\pm 8/\pm 16$ g;
- Resposta em frequência até 6 kHz (± 3 dB);
- Alta estabilidade da sensibilidade sobre temperatura e choques mecânicos;
- Temperatura de funcionamento entre -40 e +105 °C.

Com o objectivo de validação da abordagem proposta para detecção de anomalias, interessa avaliar o desempenho do modelo em dados adquiridos no meio industrial ao qual pretendíamos adicionar esta capacidade de detecção. Com recurso ao kit de desenvolvimento apresentado na Secção 4.2, em especifico ao sensor IIS3DWB e ao pacote de funções descrito na Secção 4.2.1, foi realizado um processo de aquisição, tratamento e teste de dados de meio industrial, como mostra a Fig.5.16.

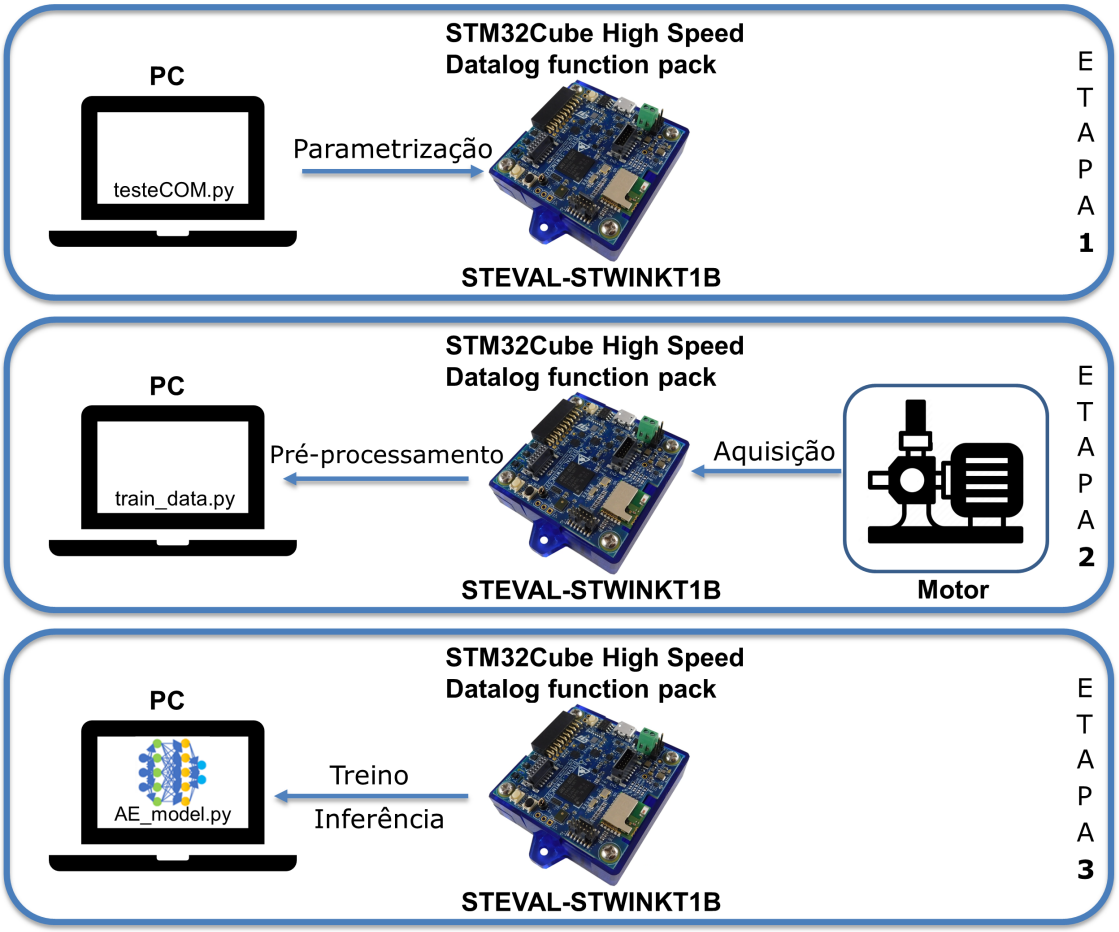


Figura 5.16: Diagramas de fluxo para aquisição e teste de dados de meio industrial no modelo proposto

Numa primeira etapa e verificando-se o kit conectado, é feita a parametrização do pacote de funções descrito na Secção 4.2.1, com vista a escolher o ODC e a escolha do eixo do sensor do qual pretendemos fazer análise, visto que se pretende analisar os dados relativos a deslocações radiais isolamos apenas os dados relativos ao eixo-z do sensor IIS3DWB. O ODC escolhido foi de 20 kHz de modo a seguir o ODC usado na aquisição do IMS-Dataset.

Na segunda etapa é feita a aquisição dos dados do motor recorrendo ao sensor IIS3DWB e a leitura e pré-processamento dos dados em concordância com o que foi executado para os dados do IMS-Dataset. Como dispomos de uma quantidade de dados inferior ao verificado no IMS-Dataset o processamento MAD é feito em conjuntos de dados de 2048 dados, ou seja conjuntos dez vezes menores que no IMS-Dataset.

Na terceira e última etapa, os dados pré-processados são normalizados e transformados num tensor 3-D necessário para que seja possível alimentar o modelo LSTM desenvolvido e assim executar o treino e teste do modelo para os dados recém adquiridos.

Na Tabela 5.4 é demonstrado um resumo das principais funções dos ficheiros Python apresentados.

Tabela 5.4: Tabela resumo do software desenvolvido para aquisição de dados com o kit de desenvolvimento STEVAL-STWINKT1B

testeCOM.py	**train_data.py**	**AE_model.py**
• Escolha do sensor	• Aquisição de dados	• Normalização dos dados
• Escolha do eixo	• Pré-processamento	• Reshape dados para tensor 3-D
• ODC a 20kHz		• Treino e teste do modelo

Capítulo 6

Resultados Experimentais

6.1 Resultados IMS-Dataset

Com recurso das ferramentas disponibilizadas pela biblioteca *Scikit-Learn* descrita na Secção 4.6, foi feita uma análise para detecção de anomalias através de dois métodos de regressão clássicos, o IF e SVM. Os resultados destes métodos para detecção de falhas no IMS-dataset são apresentados de seguida.

Isolation Forest (IF)

Na Fig.6.1 é apresentado o resultado do modelo IF para detecção de anomalias no IMS-Dataset, estando representado a vermelho os pontos considerados anomalias e azul os dados considerados normais. O modelo foi configurado com os seguintes parâmetros: (estimators=200, max samples='auto', contamination='0.4', max features=1.0 e verbose=0).

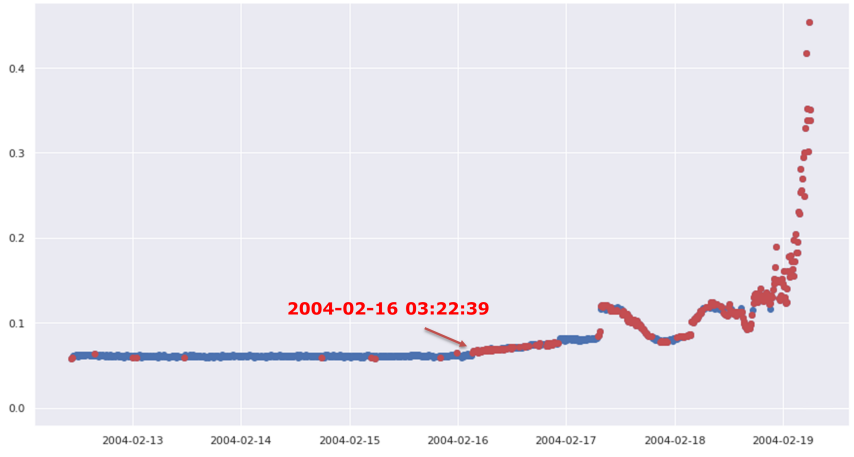


Figura 6.1: Teste de detecção de anomalias usando o modelo IF.

Tendo em vista a ocorrência de falsos positivos e falsos negativos, o que torna o modelo menos interessante para uso em ambiente industrial, optou-se por considerar a sua detecção de anomalia, somente na ocorrência de duas detecções consecutivas. Desta forma, o modelo IF tem a sua primeira deteção de anomalias na data 2004-02-16 às 03:22:39.

Support Vector Machines (SVM)

Os resultados para detecção de anomalias no IMS-Dataset pelo modelo SVM, com os seguintes parâmetros, [kernel = 'rbf', gamma = 0.005, nu = 0.5], são apresentados na Fig.6.2.

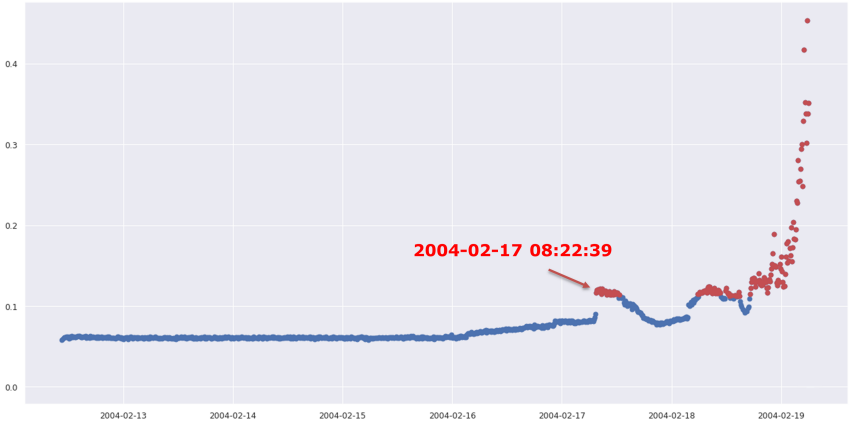


Figura 6.2: Teste de detecção de anomalias usando o modelo SVM.

Apesar dos resultados obtidos no modelo SVM, e ao contrario do IF, não apresentarem falsos positivos, há a ocorrência de falsos negativos e a detecção de anomalia é tardia, ou seja, o modelo é pouco sensível às variações do IMS-Dataset.

Por análise dos resultados de detecção de anomalias dos modelos acima testados, percebe-se diferenças no desempenho dos diferentes métodos. O método IF apesar de detectar como anomalia grande parte dos dados que se entende por anómalos tem como problema a detecção de falsos positivos antes de dia 16 e alguns períodos de falsos negativos, próximo de dia 17. No caso do modelo SVM, apesar da vantagem de não registar falsos positivos tem uma detecção de anomalias só para valores elevados relativamente ao que se entende como sendo anomalias, tornando-se um modelo com resultados pouco sensíveis a variações.

A estrutura do modelo desenvolvido e as operações que levam à detecção de anomalias são demonstradas na Fig.6.3.

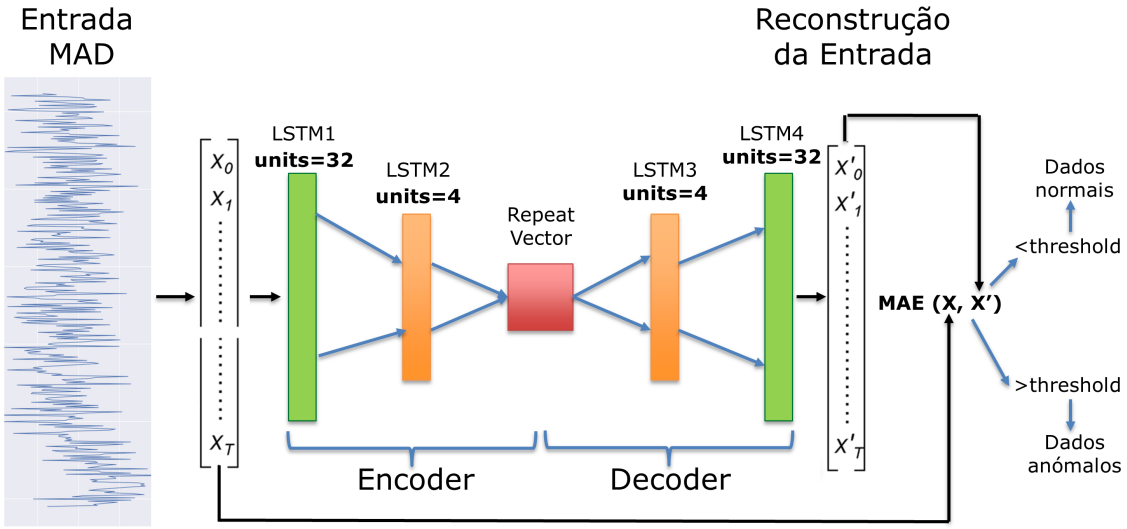


Figura 6.3: Arquitectura modelo proposto

De seguida são mostrados os resultados do modelo definido na Secção 5.3, sendo alimentado com os dados de teste previamente selecionados e com o valor de *threshold* calculado após a fase de treino. Os dados selecionados como teste são passados pelo modelo pré-treinado para avaliar o seu erro, relativo à operação de reconstrução da entrada. Os dados de erro de reconstrução do modelo na fase de treino e teste são concatenados e demonstrados na Fig.6.4, onde valores superiores ao limite calculado são detectados como anomalia.

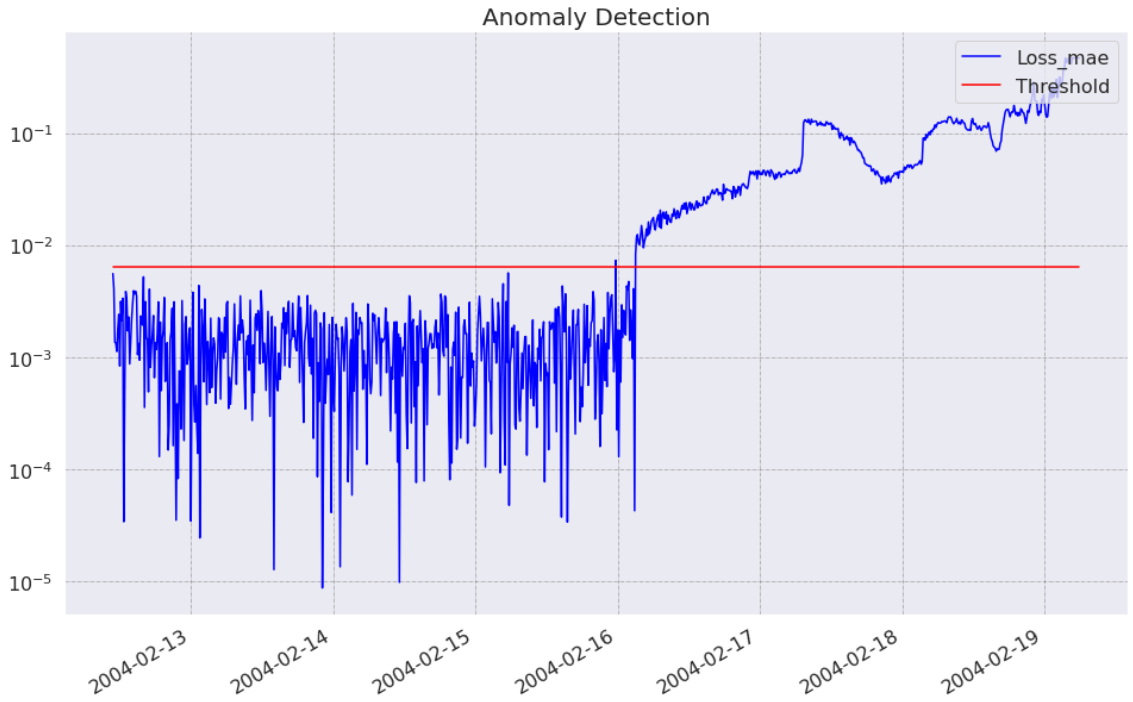


Figura 6.4: Resultados da detecção de anomalia do modelo AE-LSTM desenvolvido para dados relativos ao IMS-dataset com timestep=1

Com a análise realizada, o primeiro valor detectado acima do *threshold*, ainda que de forma isolada, ocorre em 2004-02-15 23:52:39, sendo a partir de 2004-02-16 03:12:39 todos os valores detectados como anomalias, o que resulta num total de 532 amostras normais e 452 consideradas anómalas no IMS-dataset.

Tendo em conta o modo de cálculo do *threshold* usado e a sua margem no erro de reconstrução do modelo o valor anómalo detectado a 2004-02-15 23:52:39, é já considerado como alerta para o comportamento da máquina em questão.

Na Tabela 6.1, são apresentados os resultados para detecção de anomalias no IMS-Dataset, em dois dos trabalhos citados na Secção3,nos dois métodos clássicos testados, IF e SVM e no modelo DFSBA desenvolvido.

Tabela 6.1: Comparativo de detecção de anomalia no IMS-Dataset.

Modelo	**Data da Detecção de anomalia**
[56]	2004-02-17 01:32:39
[55]	2004-02-16 07:32:39
IF	2004-02-16 03:22:39
SVM	2004-02-17 08:22:39
DFSBA	**2004-02-15 23:52:39**

Com objectivo de utilizar as capacidades de "memória temporal"das redes LSTM, importava analisar se um modelo preparado para receber os dados de entrada em blocos de dados *(timestep* > 1) melhoraria a interpretação do modelo aquando de dados anómalos. Assim, foi construído um modelo em que os dados de entrada são agrupados em grupos de 5 ($timestep = 5$) procedendo-se ao restante da análise de forma semelhante acima descrito, os resultados são apresentados de seguida, Fig. 6.5.

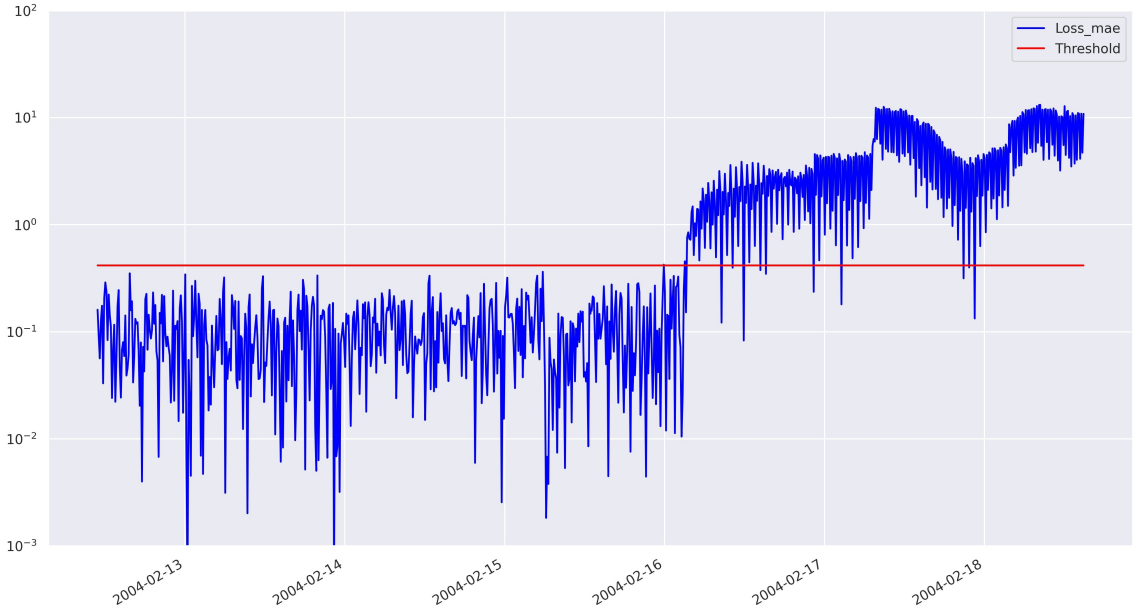


Figura 6.5: Detecção de anomalias no IMS-dataset para timestep = 5

Após os testes realizados com (timestep = 5), não se verificam melhorias no desempenho do modelo, quer na sua capacidade de detecção de anomalias, quer na estabilidade do modelo aquando da reconstrução do erro do sinal de entrada.

6.2 Resultados em dados adquiridos em meio industrial

Por se tratarem de dados adquiridos de motores em meio industrial em constante funcionamento e incorporados em sistemas sensíveis da indústria, os vários conjuntos de dados adquiridos apresentam menores dimensões do que o conjunto de dados IMS-Dataset do qual foram elaborados os testes anteriormente descritos. De seguida, é apresentado um dos conjuntos de dados adquiridos em meio industrial referentes a propagação radial da vibração, Fig.6.6.

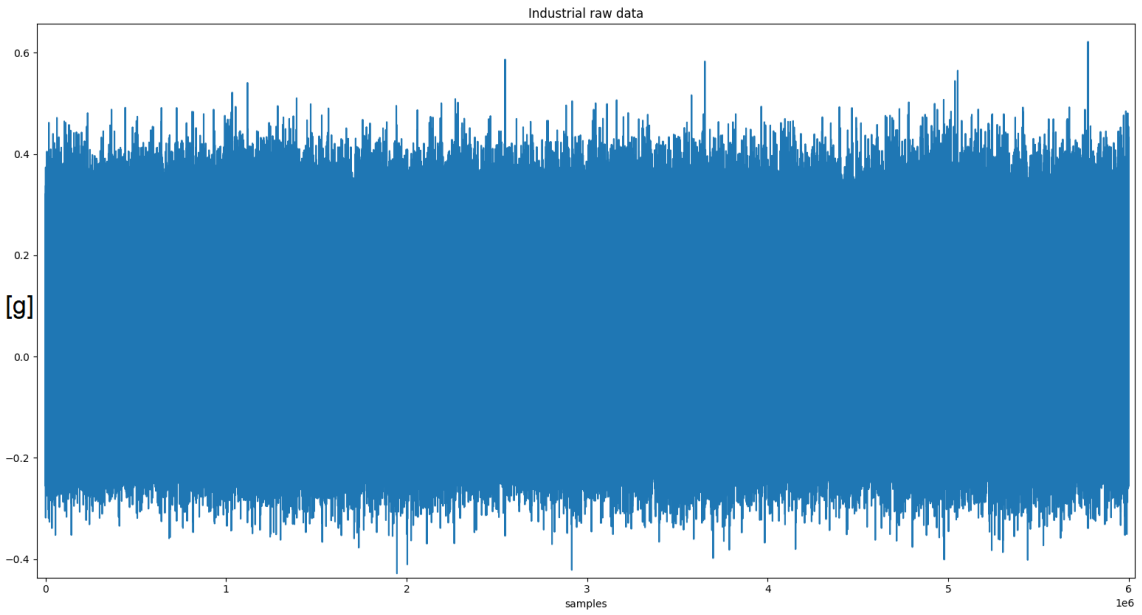


Figura 6.6: Dados em bruto adquiridos em meio industrial.

Tendo em vista a menor dimensão de dados em análise, o processamento MAD é feito em janelas deslizantes de menor dimensão, tendo sido executado em janelas deslizantes de 2048 amostras. O resultado desta operação de pré-processamento é apresentado de seguida na Fig.6.7.

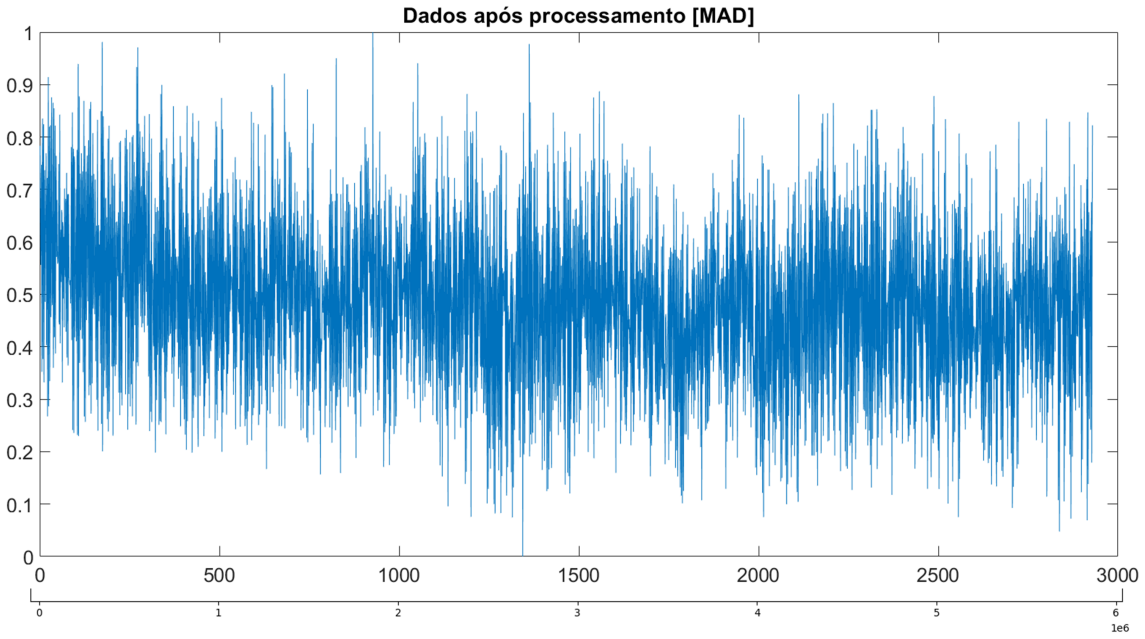


Figura 6.7: Dados adquiridos em meio industrial após processamento mad e normalização

Após o processamento dos dados, estes foram divididos em dados de treino e teste na proporção 50-50, os dados de validação representam 15% dos dados escolhidos como treino. Os dados são transformados num tensor 3D *[batch, timesteps, feature]* necessário para alimentar o modelo AE-LSTM previamente desenvolvido, e assim ser executada a fase de treino do modelo para os novos dados. De seguida, é analisada a perda do modelo na sua fase de treino e calculado o que será o *threshold* para detecção de anomalias na fase de teste.

Com o objectivo da escolha automática do threshold é feita a detecção do maior erro atingido na fase de treino do modelo acrescido de uma tolerância de 15%, como descrito e executado nos testes do IMS-Dataset. Na Fig.6.8 é demonstrada a distribuição da perda para uma melhor percepção do peso dos valores de perda.

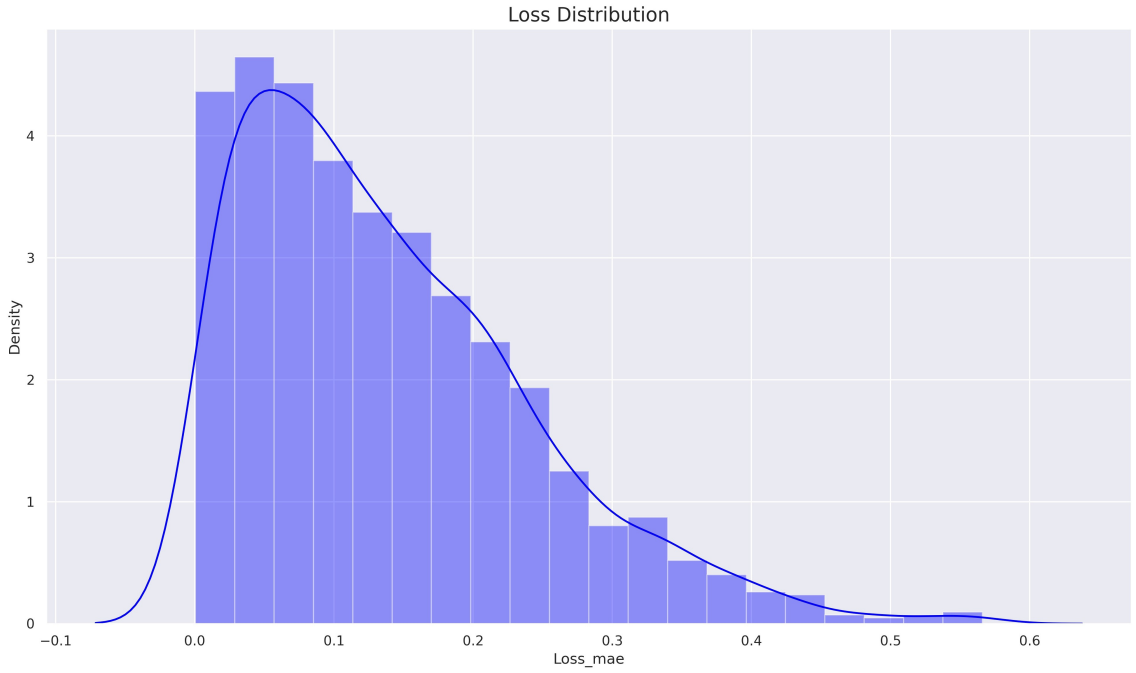


Figura 6.8: Distribuição da perda do modelo na fase de treino.

Por fim os dados selecionados como teste são passados pelo modelo pré-treinado para avaliar o seu erro relativo à operação de reconstrução da entrada. Os dados de erro de reconstrução do modelo na fase de treino e teste são concatenados e demonstrados na Fig.6.9, onde valores superiores ao limite previamente calculado são detectados como anomalia.

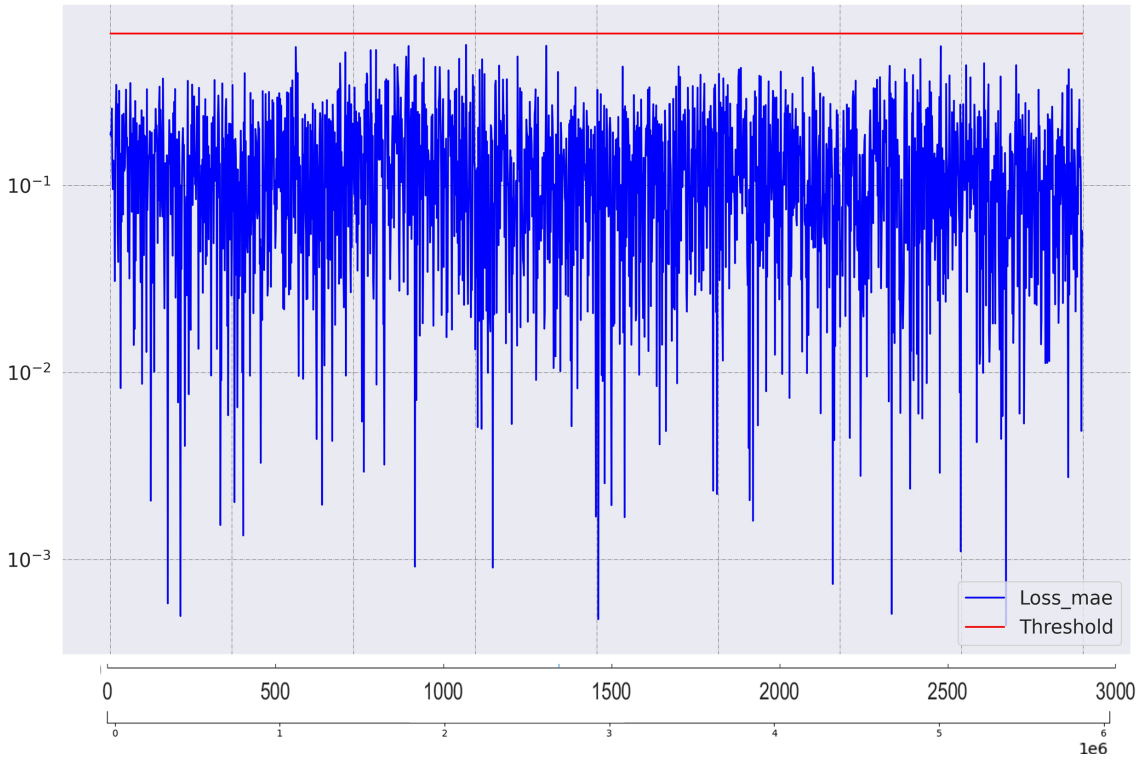


Figura 6.9: Resultados da detecção de anomalia do modelo para dados adquiridos em meio industrial.

Como esperado pelo facto do conjunto de dados em teste proveniente de meio industrial não dispor de anomalias, o modelo não detecta anomalias no conjunto de dados adquirido. Entende-se então que aquando do surgimento de alguma anomalia a reconstrução do erro do modelo irá ultrapassar o limite calculado e assim detectar o valor como anómalo.

Com o objectivo de validar a capacidade do modelo em detectar anomalias em dados provenientes da vibração de máquinas rotativas, procedeu-se à indução de pequenos choques mecânicos na máquina em estudo, pretendendo assim introduzir alterações susceptíveis de ser detectadas pelo sensor de vibração, para assim testar a capacidade do modelo detectar tais alterações como anomalias.

Os dados relativos à leitura do sensor de vibração com adição de choques mecânicos são mostrados na Fig.6.10.

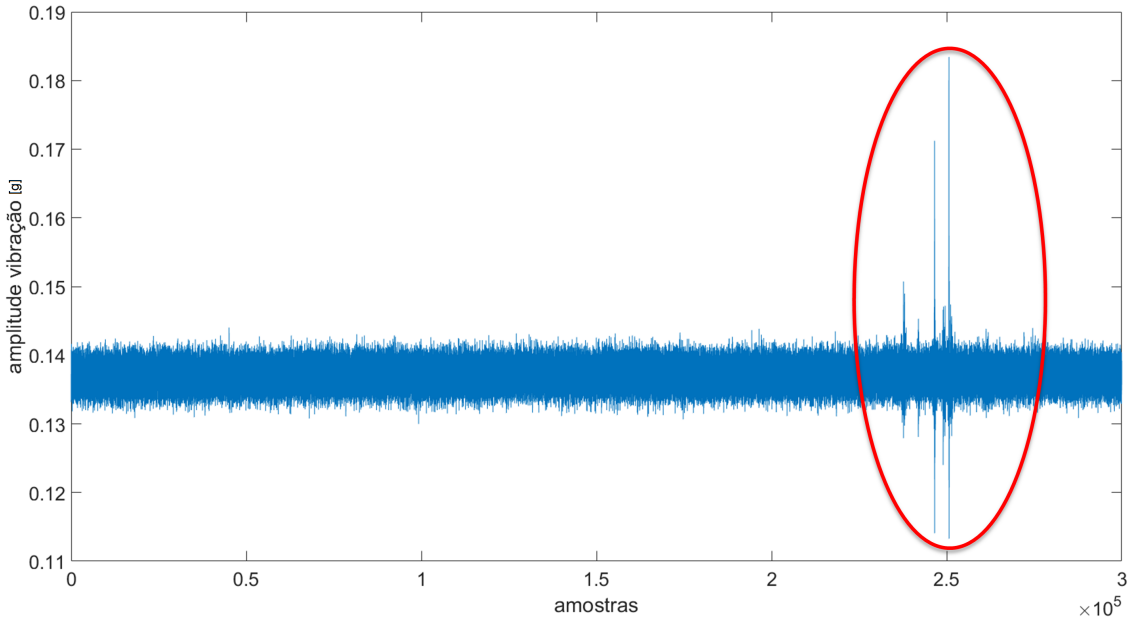


Figura 6.10: Dados em bruto de meio industrial com adição de choques mecânicos realçados (vermelho).

O processamento dos dados é feito de forma similar ao descrito nos testes anteriores, sendo que devido ao tamanho reduzido do conjunto de dados agora em análise o processamento é executado em janelas deslizantes de 250 amostras, os resultados desta operação são apresentados na Fig.6.11.

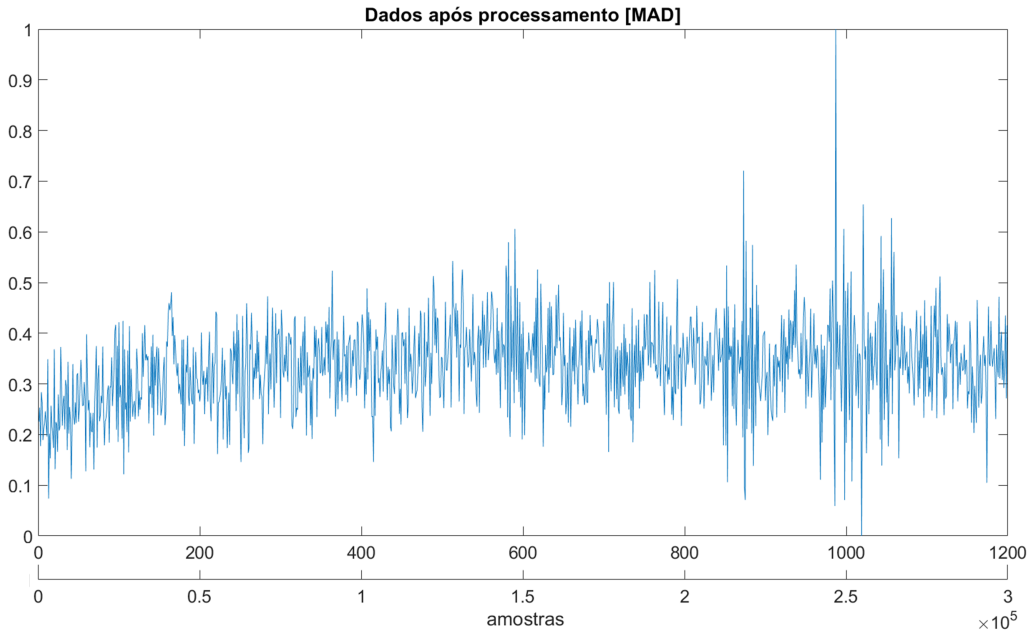


Figura 6.11: Dados de meio industrial com adição de choques mecânicos após processamento mad

Procedendo com a fase de treino e teste semelhante ao descrito em testes anteriores o resultado do modelo no que diz respeito à capacidade de detecção de anomalias é apresentado na Fig.6.12.

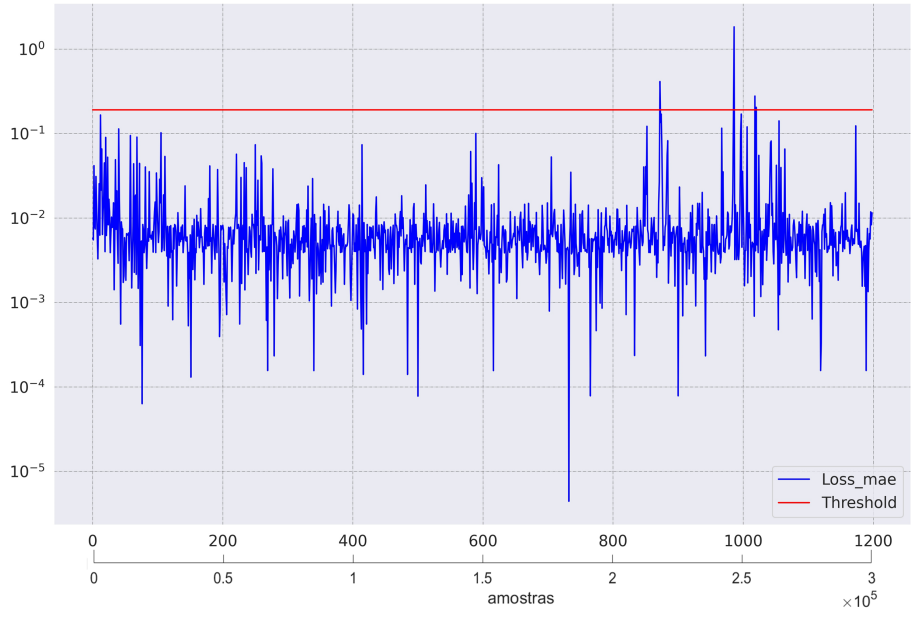


Figura 6.12: Detecção de anomalia dos dados de meio industrial com adição de choques mecânicos

Observa-se assim uma boa capacidade do modelo desenvolvido na tarefa de detecção de anomalias provenientes de variações nas medições de vibração em máquinas rotativas, tendo sido detectadas as ocorrências dos choques induzidos na estrutura do motor em análise. Os diversos resultados obtidos em dados adquiridos de meio industrial, validam a boa capacidade do modelo ser generalizado a tarefas similares para detecção de anomalias.

Capítulo 7

Conclusão

O objectivo principal deste trabalho de dissertação foi desenvolver, treinar e testar um modelo baseado em arquitectura AE computacionalmente apto a ser implementado em dispositivos de borda. O modelo demonstrou capacidade de detectar falhas em máquinas rotativas através da análise de vibração, sem dependência de dados previamente catalogados. A abordagem apresentada e por análise dos resultados obtidos permite a generalização do modelo para outras máquinas rotativas semelhantes. Esta generalização torna a abordagem interessante quando comparada com métodos como IF e SVM que têm a necessidade de ser configurados singularmente para cada contexto de aplicação. Os resultados demonstram ainda a capacidade do modelo mostrar bons resultados de detecção apesar das suas dimensões reduzidas, mesmo em volume de dados de treino consideravelmente baixo.

7.1 Trabalho Futuro

De forma a melhorar ou a acrescentar valor ao trabalho desenvolvido, vários pontos podem ser motivo de interesse, como:

- A integração e teste do modelo no microcontrolador de forma a analisar a sua performance quando executado em dispositivos de borda;
- A integração do modelo em meio industrial por um longo período para melhor caracterização das fases de funcionamento do equipamento em análise;
- Possibilidade de ser criado e catalogado um dataset em meio industrial assim que seja perceptível o surgimento de anomalias;

Bibliografia

[1] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang Suh, Ikkyun Kim, and Kuinam Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22, 01 2019.

[2] Xuan Hien Le, Hung Ho, Giha Lee, and Sungho Jung. Application of long short-term memory (lstm) neural network for flood forecasting. *Water*, 11:1387, 07 2019.

[3] B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.

[4] BS ISO and BRITISH STANDARD. Mechanical vibration—evaluation of machine vibration by measurements on non-rotating parts. 2009.

[5] IMI Sensors. Motor vibration: Detect mechanical electrical motor faults with vibration monitoring instrumentation. `https://pdf.directindustry.com/pdf/pcb-piezotronics/imi-sensors-motor-vibration/111589-378803.html`.

[6] Redhwan Al-amri, R.K. Murugesan, Mustafa Man, Alaa Fareed, Mohammed A. Al-Sharafi, and Ammar Alkahtani. A review of machine learning and deep learning techniques for anomaly detection in iot data. *Applied Sciences*, 11, 06 2021.

[7] Hai Qiu, Jay Lee, Jing Lin, and Gang Yu. Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. *Journal of Sound and Vibration*, 289:1066–1090, 02 2006.

[8] Renu Khandelwal. A basic introduction to tensorflow lite. *Medium*, Jun 2020.

[9] FAO. The state of world fisheries and aquaculture 2020. sustainability in action. *Rome, Italy*, 2020.

[10] Yuan Ai, Mugen Peng, and Kecheng Zhang. Edge computing technologies for internet of things: a primer. *Digital Communications and Networks*, 4(2):77 – 86, 2018.

[11] J. L. Kohler, J. Sottile, and F. C. Trutt. Condition-based maintenance of electrical machines. In *Conference Record of the 1999 IEEE Industry Applications Conference. Thirty-Forth IAS Annual Meeting (Cat. No.99CH36370)*, volume 1, 1999.

[12] STMicroelectronics. Condition monitoring / predictive maintenance. `https://www.st.com/en/applications/factory-automation/condition-monitoring-predictive-maintenance.html`.

[13] Alessandro Faulisi Alessandra Di Pietro, Giuseppe Rotondo. Capacitive mems accelerometer for condition monitoring. https://www.st.com/content/ccc/resource/technical/document/white_paper/group0/c0/30/46/2f/00/24/42/1c/Capacitive_MEMS_accelerometer_for_condition_monitoring/files/MEMS_Condition_monitoring.pdf/jcr:content/translations/en.MEMS_Condition_monitoring.pdf.

[14] G.K Singh and Saad Al kazzaz. Induction machine drive condition monitoring and diagnostic research - a survey. *Electric Power Systems Research*, 64:145–158, 02 2003.

[15] Paresh Girdhar and C. Scheffer. 5 - machinery fault diagnosis using vibration analysis. In Paresh Girdhar and C. Scheffer, editors, *Practical Machinery Vibration Analysis and Predictive Maintenance*, pages 89 – 133. Newnes, Oxford, 2004.

[16] Y. Han and Y. H. Song. Condition monitoring techniques for electrical equipment-a literature survey. *IEEE Transactions on Power Delivery*, 18(1):4–13, 2003.

[17] Goutam Senapaty and U. Rao. Vibration based condition monitoring of rotating machinery. *MATEC Web of Conferences*, 144:01021, 01 2018.

[18] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41, 07 2009.

[19] Bouchra Lamrini, Augustin Gjini, Simon Daudin, François Armando, Pascal Pratmarty, and Louise Travé-Massuyès. Anomaly detection using similarity-based one-class svm for network traffic characterization. 08 2018.

[20] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. volume 12, pages 582–588, 01 1999.

[21] Fei Tony Liu, Kai Ting, and Zhi-Hua Zhou. Isolation forest. pages 413 – 422, 01 2009.

[22] José Machado, Filomena Soares, Justyna Trojanowska, and Sahin YILDIRIM. *Innovations in Mechatronics Engineering*. 06 2021.

[23] Maad Mijwil, Adam Esen, and Aysar Alsaadi. Overview of neural networks. 1:2, 04 2019.

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[25] Klaus Greff, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2017.

[26] Jaime Zabalza, Jinchang Ren, Jiangbin Zheng, Huimin Zhao, Chunmei Qing, Zhijing Yang, Peijun Du, and Stephen Marshall. Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging. *Neurocomputing*, 185:1–10, 2016.

[27] Arif Ahmed and Ejaz Ahmed. A survey on mobile edge computing. 01 2016.

[28] Glenn H. Bate. Vibration diagnostics for industrial electric motor drives. 2007.

[29] R.I. JOHNSTON. Equipment breakdown responsible for nearly one-third of all property-related losses in 2018, reports fm global, Jul 2019.

[30] P. J. Tavner. Review of condition monitoring of rotating electrical machines. *IET Electric Power Applications*, 2(4):215–247, 2008.

[31] Tan Junbo, Lu Weining, An Juneng, and Wan Xueqian. Fault diagnosis method study in roller bearing based on wavelet transform and stacked auto-encoder. In *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pages 4608–4613, 2015.

[32] M. He and D. He. Deep learning based approach for bearing fault diagnosis. *IEEE Transactions on Industry Applications*, 53(3):3057–3065, 2017.

[33] F. R. Barbosa, O. M. Almeida, A. P. S. Braga, M. A. B. Amora, and S. J. M. Cartaxo. Application of an artificial neural network in the use of physicochemical properties as a low cost proxy of power transformers dga data. *IEEE Transactions on Dielectrics and Electrical Insulation*, 19(1):239–246, 2012.

[34] E. A. Rietman and M. Beachy. A study on failure prediction in a plasma reactor. *IEEE Transactions on Semiconductor Manufacturing*, 11(4):670–680, 1998.

[35] Yixing Wang, Meiqin Liu, and Zhejing Bao. Deep learning neural network for power system fault diagnosis. pages 6678–6683, 07 2016.

[36] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science (New York, N.Y.)*, 313:504–7, 08 2006.

[37] Peng Chen, Lifen Yuan, Yigang He, and Shuai Luo. An improved svm classifier based on double chains quantum genetic algorithm and its application in analogue circuit diagnosis. *Neurocomputing*, 211:202–211, 2016.

[38] Besma Bessam, Arezki Menacer, Mohamed Boumehraz, and Hakima Cherif. Detection of broken rotor bar faults in induction motor at low load using neural network. *ISA transactions*, 64, 06 2016.

[39] Mohamed Amine ATOUI, Sylvain VERRON, and Abdessamad KOBI. Conditional gaussian network as pca for fault detection. *IFAC Proceedings Volumes*, 47(3):1935–

1940, 2014. 19th IFAC World Congress.

[40] Meng Luo, Chaoshun Li, Xiaoyuan Zhang, Ruhai Li, and Xueli An. Compound feature selection and parameter optimization of elm for fault diagnosis of rolling element bearings. *ISA Transactions*, 65:556–566, 2016.

[41] Chen Lu, Zhen-Ya Wang, Wei-Li Qin, and Jian Ma. Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification. *Signal Processing*, 130, 07 2016.

[42] Feng Jia, Yaguo Lei, Jing Lin, Xin Zhou, and Na Lu. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, 72-73, 11 2015.

[43] Jiedi Sun, Changhong Yan, and Jiangtao Wen. Intelligent bearing fault diagnosis method combining compressed data acquisition and deep learning. *IEEE Transactions on Instrumentation and Measurement*, 67(1):185–195, 2018.

[44] Wenjun Sun, Siyu Shao, Rui Zhao, Ruqiang Yan, Xingwu Zhang, and Xuefeng Chen. A sparse auto-encoder-based deep neural network approach for induction motor faults classification. *Measurement*, 89, 04 2016.

[45] Zong Meng, Xuyang Zhan, Jing Li, and Zuozhou Pan. An enhancement denoising autoencoder for rolling bearing fault diagnosis. *Measurement*, 130, 08 2018.

[46] X. Ding and Q. He. Energy-fluctuated multiscale feature learning with deep convnet for intelligent spindle bearing fault diagnosis. *IEEE Transactions on Instrumentation and Measurement*, 66(8):1926–1935, 2017.

[47] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Transactions on Industrial Electronics*, 65(2):1539–1548, 2018.

[48] Chuan Li, René Sánchez, Grover Zurita, Mariela Cerrada, Diego Cabrera, and Rafael Vasquez. Multimodal deep support vector classification with homologous features and its application to gearbox fault diagnosis. *Neurocomputing*, 168, 06 2015.

[49] B. SAMANTA and K.R. AL-BALUSHI. Artificial neural network based fault diagnostics of rolling element bearings using time-domain features. *Mechanical Systems and Signal Processing*, 17(2):317 – 328, 2003.

[50] Donghyun Park, Seulgi Kim, Yelin An, and Jae-Yoon Jung. Lired: A light-weight real-time fault detection system for edge computing using lstm recurrent neural networks.

Sensors, 18:2110, 06 2018.

[51] G. Qian, S. Lu, D. Pan, H. Tang, Y. Liu, and Q. Wang. Edge computing: A promising framework for real-time fault diagnosis and dynamic control of rotating machines using multi-sensor data. *IEEE Sensors Journal*, 19(11):4211–4220, 2019.

[52] Pier Orrù, Andrea Zoccheddu, Lorenzo Sassu, Carmine Mattia, Riccardo Cozza, and Simone Arena. Machine learning approach using mlp and svm algorithms for the fault prediction of a centrifugal pump in the oil and gas industry. *Sustainability*, 12:4776, 06 2020.

[53] Yi Gu, Jiawei Cao, Xin Song, and Jian Yao. A denoising autoencoder-based bearing fault diagnosis system for time-domain vibration signals. *Wireless Communications and Mobile Computing*, 2021:9790053, May 2021.

[54] Rui Yang, Mengjie Huang, Qidong Lu, and Maiying Zhong. Rotating machinery fault diagnosis using long-short-term memory recurrent neural network. *IFAC-PapersOnLine*, 51:228–232, 01 2018.

[55] Mohammad Kaji, Jamshid Parvizian, and Hans Wernher van de Venn. Constructing a reliable health indicator for bearings using convolutional autoencoder and continuous wavelet transform, 12 2020.

[56] Sabtain Ahmad, Kevin Styp-Rekowski, Sasho Nedelkoski, and Odej Kao. Autoencoder-based condition monitoring and anomaly detection method for rotating machines. 01 2021.

[57] Stwin sensortile wireless industrial node development kit and reference design for industrial iot applications. `https://www.st.com/resource/en/data_brief/steval-stwinkt1b.pdf`. Acessado a 08-04-2021.

[58] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, and Xiaoqiang Zhang. Tensorflow: A system for large-scale machine learning. 05 2016.

[59] Monica Ramchandani, Hrishikesh Khandare, Priyanshi Singh, Prince Rajak, Nidhi Suryawanshi, Anjali Jangde, Laxmi Arya, Prithviraj Kumar, and Mridu Sahu. Survey: Tensorflow in machine learning. *Journal of Physics: Conference Series*, 2273:012008, 05 2022.

[60] Wazir Muhammad, Irfan Ullah, and Mohammad Ashfaq. *An Introduction to Deep*

Convolutional Neural Networks With Keras, pages 231–272. 01 2020.

[61] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[62] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[63] Julien Herzen, Francesco Lässig, Samuele Giuliano Piazzetta, Thomas Neuer, Léo Tafti, Guillaume Raille, Tomas Van Pottelbergh, Marek Pasieka, Andrzej Skrodzki, Nicolas Huguenin, Maxime Dumonal, Jan Kościsz, Dennis Bader, Frédérick Gusset, Mounir Benheddi, Camila Williamson, Michal Kosinski, Matej Petrik, and Gaël Grosch. Darts: User-friendly modern machine learning for time series, 2021.

[64] J.Scott Armstrong and Fred Collopy. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, 8(1):69–80, 1992.

[65] Spyros Makridakis and Michèle Hibon. The m3-competition: results, conclusions and implications. *International Journal of Forecasting*, 16(4):451–476, 2000. The M3-Competition.

[66] Spyros Makridakis. Accuracy measures: theoretical and practical concerns. *International Journal of Forecasting*, 9(4):527–529, 1993.

[67] J. Scott Armstrong and Robert Fildes. Correspondence on the selection of error measures for comparisons among forecasting methods. *Journal of Forecasting*, 14(1):67–71, 1995.

[68] Boris Shishov. Deepforecasting. `https://github.com/bshishov/DeepForecasting/blob/master/metrics.py`, 2018.

[69] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. `https://github.com/keras-team/keras-tuner`, 2019.

[70] Introduction to the keras tuner. `https://www.tensorflow.org/tutorials/keras/keras_tuner`. Acessado a 23-05-2022.

[71] Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5(1):18, May 2016.