UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS SOBRAL

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

MAGDIEL CAMPELO ALVES DE SOUSA

PLATAFORMA IOT DE BAIXO CUSTO PARA UTILIZAÇÃO EM MANUTENÇÃO PREDITIVA

SOBRAL

2018

MAGDIEL CAMPELO ALVES DE SOUSA

PLATAFORMA IOT DE BAIXO CUSTO PARA UTILIZAÇÃO EM MANUTENÇÃO PREDITIVA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Orientador: Prof. Me. Wendley S. Silva

SOBRAL

2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S697p Sousa, Magdiel Campelo Alves de.
Plataforma IoT de baixo custo para manutenção preditiva / Magdiel Campelo Alves de Sousa. – 2018.
77 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Sobral, Curso de Engenharia da Computação, Sobral, 2018.
Orientação: Prof. Me. Wendley S. Silva.

1. Internet das Coisas. 2. Manutenção Preditiva. 3. NodeMCU. 4. Redes Neurais Recorrentes. I. Título.

CDD 621.39

MAGDIEL CAMPELO ALVES DE SOUSA

PLATAFORMA IOT DE BAIXO CUSTO PARA UTILIZAÇÃO EM MANUTENÇÃO PREDITIVA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Campus Sobral da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Aprovada em:

BANCA EXAMINADORA

Prof. Me. Wendley S. Silva (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Me. David Nascimento Coelho
Universidade Federal do Ceará (UFC)

Raimundo Farrapo Pinto Júnior
Universidade Federal do Ceará (UFC)

À minha família, por sempre acreditar em mim
e investir na minha formação.

AGRADECIMENTOS

Este trabalho não seria possível sem a ajuda e o apoio de muitas pessoas que me encorajaram, incentivaram e principalmente me inspiraram durante o período da graduação.

Primeiramente gostaria de agradecer aos meus pais Antônio Francisco e Maria da Cruz por todo apoio incondicional que tenho recebido desde o momento que decidi mudar de cidade e sair completamente da minha zona de conforto para correr atrás de um sonho, sem os seus esforços nada disso seria possível. Aos meus irmãos que sempre estiveram do meu lado, me apoiando, dando conselhos e incentivando sempre que precisei, vocês são minha inspiração.

Ao meu orientador, Prof. Me. Wendley S. Silva, com o qual pude aprender bastante como monitor, e pela paciência em minha orientação e ensinamentos que sem dúvidas foram úteis para a concretização deste trabalho.

A banca examinadora por arranjar tempo para participar da apresentação e avaliação deste Trabalho de Conclusão de Curso.

A todos os professores dos cursos de Engenharia de Computação e Engenharia Elétrica da Universidade Federal do Ceará, campus Sobral, que compartilharam um pouco de seu conhecimento comigo e contribuíram para minha formação. Em especial, ao Prof. Dr. Iális Cavalcante de Paula Júnior, com o qual tive o privilégio de passar três anos sob sua tutoria no Programa de Educação Tutorial (PET) e acompanhar seu brilhante trabalho em prol do curso, sem ele o curso não seria o que é hoje.

A todos os meus amigos da universidade com quem tive o prazer de conviver por cinco anos, em especial a minha panelinha Eloise Carvalho, Victória Tomé, Isaac Newton, Syllas Rangel, Laércio Santana e Isaac Ben enfrentamos muitas coisas juntos, muitos momentos de alegrias, tristezas, aperreios, pizzas, sem vocês não teria como passar por tudo isso. Não irei citar o nome de todos pois cultivei muitas amizades que levarei para a vida, e a lista ficaria imensa, cada um sabe a importância que teve e tem na minha vida.

E por fim, mas não menos importante à minha namorada Hillary Medeiros por todo apoio ao longo desse ano.

Muito Obrigado.

"Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível."

(Charles Chaplin)

RESUMO

Este trabalho apresenta uma plataforma para manutenção preditiva de baixo custo, baseando-se em tecnologias de Internet das Coisas, utilizando de um NodeMCU como dispositivo de coleta de dados conectado à sensores de temperatura e corrente e um acelerômetro. O qual é responsável por enviar os dados à um servidor, em que foi desenvolvido uma API para tratar os dados e construir uma Rede Neural Recorrente com o objetivo de identificar o sistema, ou seja, simular matematicamente a chance de falha de um equipamento baseado na coleta de dados intrínsecos a seu funcionamento. Dessa forma, sendo capaz de prever valores futuros, fornecendo uma estimativa de período de falha, e comparar com outros estimadores. A avaliação do sistema será realizada com coleta de dados reais para o sistema, além do uso de um banco de dados com coletas de um longo período de tempo de outro sistema real para análise da eficiência em predição de valores.

Palavras-chave: Internet das Coisas. Manutenção Preditiva. NodeMCU. Rede Neural Recorrente

ABSTRACT

The present work presents a low cost predictive maintenance platform, based on Internet of Things technologies, using a NodeMCU as a data collection device connected to temperature and current sensors and an accelerometer. It will be responsible for sending the data to a server, where an API that was developed to treat the data and build a Recurrent Neural Network with the purpose of identifying the system, that is, mathematically simulating the failure probability of an equipment based on collection of intrinsic data to its operation, being able to predict future values, providing an estimate of failure period, and compare with other estimators. The evaluation of the system will be performed with real data collection for the system, in addition to the use of a database with a long period data collection of another real system to analyze the efficiency in predicting values.

Keywords: Internet of Things. Predictive Maintenance. NodeMCU. Recurrent Neural Network.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

IIoT	*Industrial Internet of Things*, ou Internet das Coisas Industrial
IoT	*Internet of Things*, ou Internet das Coisas
MIT	*Massachusetts institute of technology*, o Instituto de tecnologia de Massachusetts
RFID	*Radio-Frequency Identification*, em tradução livre, identificação por radiofrequência
RNN	*Recurrent Neural Network*, ou Rede Neural Recorrente
WSIS	World Summit on the Information Society
ADC	Conversor Analógico-Digital
API	*Application Programming Interface*, ou Interface de Programação de Aplicação
AR	auto regressivos
ARMAX	auto regressivos com médias móveis e entradas exógenas
ARX	auto regressivos com entradas exógenas
CRUD	*Create, Read, Update and Delete*
I2C	*Inter-Integrated Circuit*
IC	Inteligência Computacional
MAE	*Mean Absolute Error*
MAPE	*Mean Absolute Percentage Error*
MQ	Método dos Mínimos Quadrados
MQTT	*Message Queuing Telemetry Transport*
RMS	*Root Mean Square*
RMSE	*Root Mean Squared Error*
RN	Rede Neural
RNA	Rede Neural Artificial
SPA	*Single Page Application*
UIT	União Internacional de Telecomunicações
USB	*Universal Serial Bus*

LISTA DE SÍMBOLOS

V	`Volt`, unidade de tensão elétrica
A	`Ampere`, unidade de corrente elétrica
b	Bit, unidade mínima de informação computacional
Hz	`Hertz`, unidade de frequência em oscilações por segundo
g	Aceleração da gravidade
^{o}C	Graus Celcius
CV	Cavalo-Vapor

SUMÁRIO

1 INTRODUÇÃO

As pesquisas com *Internet of Things*, ou Internet das Coisas (*IoT*) tem ganhado bastante espaço no cenário acadêmico, fornecendo mais possibilidades para a utilização em larga escala desta tecnologia emergente. A *IoT* terá um grande impacto na economia, melhorando a eficiência e aumentando o engajamento de funcionários e clientes, além de transformar empresas em negócios digitais e facilitar novos modelos de negócios (HUNG, 2017). De acordo com Hung (2017) até 2020 existirão 20 bilhões de coisas conectadas, sem contar com equipamentos de propósitos gerais como *smartphones* e computadores.

A maior barreira para a implantação da IoT, é que a maioria das empresas não sabe o que fazer com a tecnologia (HUNG, 2017). As soluções de *IoT* não se limitam ao uso do público em geral. Visando uma maior produtividade na manufatura, as tecnologias relacionadas a *IoT* proporcionaram a ascensão da *Industrial Internet of Things*, ou Internet das Coisas Industrial (*IIoT*), com aplicações envolvendo desde o controle e comunicação das máquinas entre si, segurança do operador, até a manutenção do equipamento, oferecendo uma solução preditiva utilizando tecnologias estatísticas e cognitivas para extrair informações dos dados coletados pelos sensores.

A *IIoT* é a motivação para a modernização industrial, fazendo parte da Indústria 4.0, a nova revolução industrial. Com a aplicação das tecnologias emergentes, a *IIoT* é capaz de continuamente, em tempo real, capturar dados de vários sensores e objetos, enviar de forma segura as leituras para *data centers* baseados em nuvem, e ajustar parâmetros de manufatura em sistemas de malha fechada (com *feedback*), proporcionando a detecção de falhas de forma efetiva e acionar processos de manutenção, reagindo às alterações inesperadas na produção de forma autônoma. Entretanto, apesar da tecnologia existente, ainda é um desafio analisar, de forma coerente, os dados capturados, pelos dispositivos com sensores, dos mais variados tipos de maquinários e equipamentos. E isso se deve à falta de padronização dos equipamentos de instrumentação, protocolos, e diretrizes de segurança (ZHANG *et al.*, 2016).

1.1 Motivação

Dentro dos principais motivos pelos quais uma organização decide adicionar à sua operação um programa de manutenção preditiva, pode-se mencionar a proteção de ativos financeiros, reconhecimento das capacidades de melhorias pelo nível diretivo, menores taxas de

seguros e maiores possibilidades de obtenção da certificação ISO 9001 (MOBLEY, 2002).

González (2014) cita em seu trabalho, complementando, vantagens como a redução dos custos de manutenção, redução de falhas nas máquinas, consequentemente uma operação mais fluida focada na produtividade, diminuição nos tempos de parada para reparo, aumento da produção, entre outras vantagens. É importante destacar também que as técnicas de manutenção preditiva exigem uma instrumentação sofisticada e pessoas treinadas para o seu manuseio e análise dos dados, que geralmente possuem um custo elevado. Este trabalho propõe uma solução aplicando sensores e dispositivos de baixo custo para uma aplicação em manutenção preditiva.

1.2 Objetivos

O presente trabalho propõe uma aplicação para a manutenção preditiva de equipamentos rotores, tais como motores, ventilação e braços robóticos, baseando-se em leituras de vibração, temperatura e corrente elétrica por dispositivos com sensores conectados à *internet*, e utilizando de tecnologias cognitivas de análise de dados, identificação de sistemas e predição de valores com o objetivo de fornecer uma previsão de uma margem de tempo para a falha do equipamento.

1.2.1 Objetivos específicos

Os seguintes tópicos consistem nos objetivos específicos que são almejados no trabalho realizado:

- Desenvolver um sistema embarcado conectado à *internet* que funciona como *datalogger* para um sistema de manutenção preditiva, que possui sensores e envia dados para a plataforma *web*
- Desenvolver um sistema *web* em que será possível cadastrar sensores e visualizar dados e análises
- Desenvolver uma API (Biblioteca) que será responsável por processar os dados e extrair informações úteis para o sistema de manutenção preditiva
- Simular uma aplicação com dados reais
- Estudar e comparar métodos de predição de valores

Ao final do trabalho, espera-se encontrar resultados pertinentes para a proposta, com uma sugestão de uso de *IoT* que pode ser aplicado à indústria.

1.3 Organização do trabalho

O presente trabalho está organizado como segue. No capítulo 2 são apresentados os conceitos teóricos abordados na construção do trabalho, como o conceito de manutenção preditiva, severidade de vibração em máquinas rotativas, internet das coisas, análise de séries temporais e redes neurais, seguindo pelos trabalhos relacionados já no capítulo 3. E então, no capítulo 4, é apresentada a metodologia utilizada para desenvolver o sistema e as técnicas utilizadas. Na sequência, são expostos no capítulo 5 os resultados e análises obtidas. E, por fim, no capítulo 6 são apresentadas as conclusões e possíveis aprimoramentos.

2 FUNDAMENTAÇÃO TEÓRICA

A predição de falhas nos principais equipamentos mecânicos de uma manufatura é um tópico muito importante na indústria pois possui o objetivo de reduzir os custos com manutenção realizando-a no melhor momento, e o sistema de previsão baseado em *IoT* é a nova maneira de garantir a operação segura desses equipamentos, possibilitando o monitoramento em tempo real reduzindo os custos de operação da análise. Este tipo de sistema combina as características da *IoT* com a computação aplicada, o que possibilita a utilização das mais variadas tecnologias computacionais de análise com segurança e eficiência, a fim de realizar a previsão de falhas.

2.1 Manutenção Preditiva

De acordo com a NBR 5462 (ASSOCIAçãO BRASILEIRA DE NORMAS TéCNICAS - ABNT, 1994), manutenção é a combinação de todas as ações técnicas e administrativas incluindo as de supervisão destinadas a manter ou recolocar um item em um estado no qual possa desempenhar uma função requerida, podendo ser classificada em três tipos de acordo com a forma de intervenção.

Corretiva: Manutenção realizada após a ocorrência de uma falha, com o objetivo de tornar o equipamento operante novamente.

Preventiva: Manutenção realizada em intervalos predeterminados, ou de acordo com critérios prescritos, baseado em estudos estatísticos realizados pelo fabricante ou pesquisadores em que o equipamento foi testado até seus limites, com o objetivo de reduzir a probabilidade de falha ou degradação do equipamento.

Preditiva: Manutenção que permite garantir uma qualidade de serviço desejada, com base na aplicação sistemática de técnicas de análise, ocorre uma vez que o sistema perceba uma tendência do equipamento apresentar falhas, utilizando-se de meios de supervisão centralizados ou de amostragem. Possui o objetivo de reduzir ao mínimo a manutenção preventiva e diminuir a manutenção corretiva.

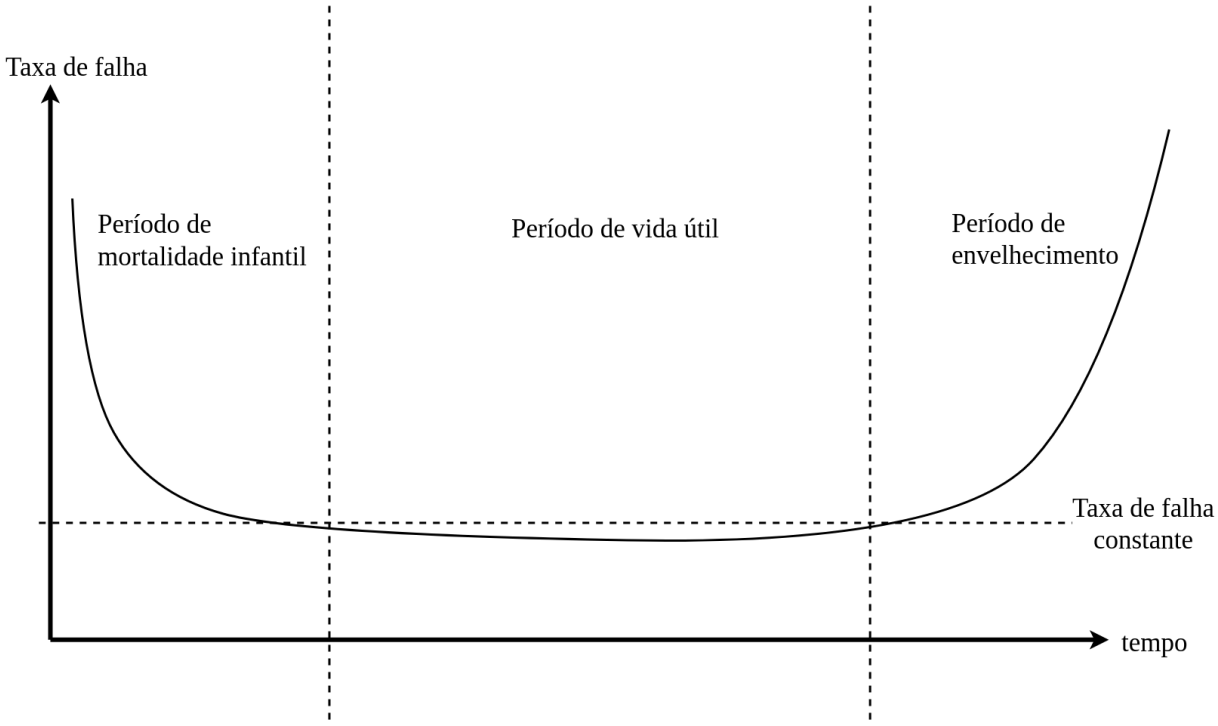
Baseando-se no conhecimento e análise dos fenômenos que ocorrem nas máquinas, é possível obter uma estimativa de eventuais falhas ou defeitos que venham a ocorrer no equipamento. A partir dessa análise dos fenômenos o sistema de manutenção preditiva é capaz de estabelecer diagnósticos, que indicam a origem e gravidade do defeito constatado, e efetuar

análise de tendências, que prediz com uma certa antecedência a avaria ou quebra.

A premissa da manutenção preditiva apoia-se no fato de que o monitoramento das condições mecânicas reais dos equipamentos e máquinas em uso, e do rendimento operacional dos sistemas de processo, garantem o intervalo máximo entre os reparos, minimizando o número e o custo de interrupções não programadas, consequentemente melhorando a disponibilidade geral de plantas operacionais (MOBLEY, 2002).

O comportamento relacionado a taxa de falha de um equipamento durante seu tempo de vida pode ser representado pela curva da banheira mostrada na Figura 1. Essa curva indica as etapas típicas de um sistema: mortalidade infantil, que podem ocorrer falhas por componentes fracos ou mal fabricados, vida útil, em que a taxa de falhas é constante, e envelhecimento, no qual a taxa é crescente (WEBER, 2003).

Figura 1 – Curva da banheira



Fonte: Adaptado de Weber (2003).

Como Weber (2003) explica, as falhas que ocorrem em equipamentos durante o período de mortalidade infantil geralmente são falhas prematuras e decorrem de má especificação do projeto ou por falha na fabricação de componentes e/ou nos processos de montagem e estocagem. Na maturidade já não ocorrem falhas prematuras e é observada uma taxa constante, os defeitos constatados nesse período são aleatórios, causados por incidentes imprevisíveis, tais como catástrofes naturais ou erros de operação. Finalmente, a taxa crescente de falhas no período

senil é decorrente do desgaste dos materiais, tornando-se inevitáveis por perda de resistência, causadas por corrosão, fadiga, trincas, deterioração mecânica, elétrica ou química.

Conforme González (2014) afirma, do ponto de vista da expectativa de vida de um equipamento, a curva da banheira, sendo um gráfico que expressa a expectativa de falha de um item ao longo do tempo considerando que ele ainda não falhou até o momento, pode ser utilizada como ponto de partida em um programa de manutenção preditiva. Então, do modelo da curva da banheira, é possível inferir dos dados coletados uma análise de em quanto tempo o equipamento terá mais chances de falhar, fornecendo um limiar para que uma ação de manutenção preditiva seja realizada em tempo ótimo.

Para que o sistema de manutenção preditiva possa atuar é necessário que observações de características sejam realizadas nos equipamentos. Os parâmetros a serem medidos devem fornecer informações que sejam relevantes para a inspeção de elementos específicos da máquina ou do tipo de falha. Com esses dados é possível utilizar técnicas de análise, em que seus resultados indicam a natureza da falha, que se pode esperar, e estabelecer a partir disso quais são os elementos críticos do sistema (MARÇAL, 2000). Na Tabela 1 estão exemplificados alguns parâmetros passíveis de serem medidos.

Tabela 1 – Parâmetros passíveis de medição em manutenção preditiva.

Parâmetro a ser medido	**Natureza da falha ou defeito a ser detectado**
Amplitude de deslocamento da vibração	Desbalanceamento, desalinhamento, jogo excessivo, falta de rigidez, acoplamento defeituoso, correias frouxas ou vastas, eixos deformados
Amplitude da velocidade da vibração	Mancais ou engrenagens deterioradas
Amplitude da aceleração da vibração	Estado mecânico dos rolamentos, atrito excessivo entre componentes, falta de lubrificação
Frequência da vibração	Dado complementar a medição de qualquer característica da vibração
Fase da vibração	Desbalanceamento dinâmico, folga excessiva, partes frouxas ou soltas
Nível de ruído	Rolamentos ou engrenagens deterioradas, desgastes, cavitação, turbulência, aumento do atrito
Fugas	Deterioramento de selos, juntas e gaxetas
Espessura	Corrosão ou erosão em tanques e tubulações
Temperatura	Lubrificação inadequada, engrenamento, aumento do atrito, sobrecarga, desalinhamento de mancais, produção excessiva de calor em componentes elétricos
Pressão	Deterioramento de rotores, bloqueio de tubulações, válvulas travadas

Fonte: Adaptado de Marçal (2000).

Como Scheffer e Girdhar (2004) afirmam no livro *Practical Machinery Vibration Analysis and Predictive Maintenance*, a análise de vibração é indubitavelmente a melhor técnica para detecção de defeitos mecânicos em máquinas rotacionais, e como é mostrado na Tabela 1, é a que mais fornece informações intrínsecas sobre o estado e saúde da máquina. Também é necessário destacar que essas técnicas de manutenção preditiva exigem instrumentos tecnicamente sofisticados para realizar a detecção e diagnóstico das máquinas, e esses instrumentos geralmente são caros e requerem pessoas qualificadas para operar o sistema (SCHEFFER; GIRDHAR, 2004).

2.1.1 Severidade da vibração

Vibração pode ser simplificadamente definida como o movimento que uma máquina, ou alguma parte sua, faz de uma direção a outra de sua posição de descanso (SCHEFFER; GIRDHAR, 2004).

Como foi mencionado anteriormente a vibração é uma medida que pode ser utilizada para determinar a severidade do defeito de uma máquina rotacional, e um dilema comum que surge com a análise de vibração é determinar se esses valores são aceitáveis para permitir a contínua operação do equipamento de forma segura.

De acordo com Scheffer e Girdhar (2004), o padrão mais utilizado como indicador de severidade é o ISO 2372 (BS 4675), que pode ser visualizado na Figura 2 a seguir, em que as classes são determinadas pela potência do motor, da seguinte forma:

Classe I: Motores elétricos até 15kW (20 CV)

Classe II: Motores elétricos de 15kW até 75kW (20 a 100 CV).

Classe III: Motores elétricos acima de 75kW com base rígida (> 100 CV)

Classe IV: Motores elétricos acima de 75kW com base flexível (> 100 CV)

Porém, como pode ser verificado pelo ISO (2018), esse padrão foi publicado em 1974 e foi revisado pelo ISO 10816-1 em 1995, e este novamente pelo ISO 20816-1 em 2016. Por isso, pode ser observado no Anexo A uma padronização mais recente que é utilizada na indústria, e fornecida pela Azima DLI (2018), que é uma empresa de manutenção preditiva.

A partir da tabela contida no Anexo A, é possível notar que a vibração pode ser utilizada com o mesmo padrão em unidades diferentes para determinar a severidade de utilização, permitindo alternar entre, ou medir em, unidades de deslocamento, velocidade e aceleração.

Figura 2 – Tabela de severidade de vibração ISO 2372

ISO 2372 - ISO Guia para Severidade de Vibração de Maquiinário					
Intervalo de Severidade de Vibração		Exemplos de jugamento de qualidade para diferentes classes de maquinário			
Velocidade in/s - pico	Velocidade mm/s - rms	Classe I	Classe II	Classe III	Classe IV
0,015	0,28				
0,025	0,45				
0,039	0,71				
0,062	1,12				
0,099	1,8				
0,154	2,8				
0,248	4,5				
0,392	7,1				
0,617	11,2				
0,993	18				
1,54	28				
2,48	45				
3,94	71				

A - Bom
B - Aceitável
D - Não aceitável

Fonte: Adaptado de Scheffer e Girdhar (2004).

2.1.2 Captura de dados

A captura de dados é um passo muito importante no sistema de manutenção preditiva, a consistência dos dados é fundamental para que o modelo de análise possua um bom resultado. Dependendo do método de análise e da criticidade do equipamento utilizados a frequência de amostragem pode variar. Métodos mais antigos ainda demandam uma captura dependente de um operador, os *smart devices IoT* podem ser utilizados de forma generalizada para coletar dados no campo com o objetivo de melhorar a produtividade por meio de processos automáticos avançados (CIVERCHIA *et al.*, 2017).

Os sistemas de monitoramento das máquinas, diagnóstico e predição de falhas têm se baseado em três modos, nominalmente: *offline*, em que a informação é capturada no equipamento por vários sensores e então transferida para um computador no qual o sistema irá identificar os dados e realizar o diagnóstico, porém apesar de ser econômico e conveniente é aplicável apenas para a detecção regular; *online* de equipamento único, em que um conjunto de sensores e sistema de análise de falhas é instalado para um ou um tipo de equipamento, possui bom desempenho em

tempo real e alta confiabilidade porém não é econômico e difícil de compartilhar informações entre diferentes sistemas de monitoramento e diagnóstico; e *online* distribuído, que pode superar os problemas de economia e dificuldade de compartilhamento de economia até certo ponto, mas é limitado pelo território e falha em realizar a previsão remota das falhas (XU *et al.*, 2012).

Sistemas de predição de falhas baseados em *IoT* são construídos em cima dos três modos, podendo superar o padrão atual de sistema e monitoramento e diagnóstico com tecnologias emergentes e econômicas, aumentando a eficiência operacional e o nível de inteligência da predição de falhas.

2.2 Internet das Coisas Industrial

O conceito de Internet das Coisas (*IoT*) foi inicialmente derivado de um sistema de rede com *Radio-Frequency Identification*, em tradução livre, identificação por radiofrequência (*RFID*) pelo centro de identificação automática que foi estabelecido no *Massachusetts institute of technology*, o Instituto de tecnologia de Massachusetts (*MIT*), em 1999, mas apenas em 2005 a União Internacional de Telecomunicações (UIT) reconheceu o conceito de *"Internet of Things"* formalmente na conferência World Summit on the Information Society (*WSIS*) (DONG *et al.*, 2017). O *IoT* abriu portas para um mundo conectado, a popularização desse conjunto de tecnologias atingiu uma visão além das casas ou objetos inteligentes.

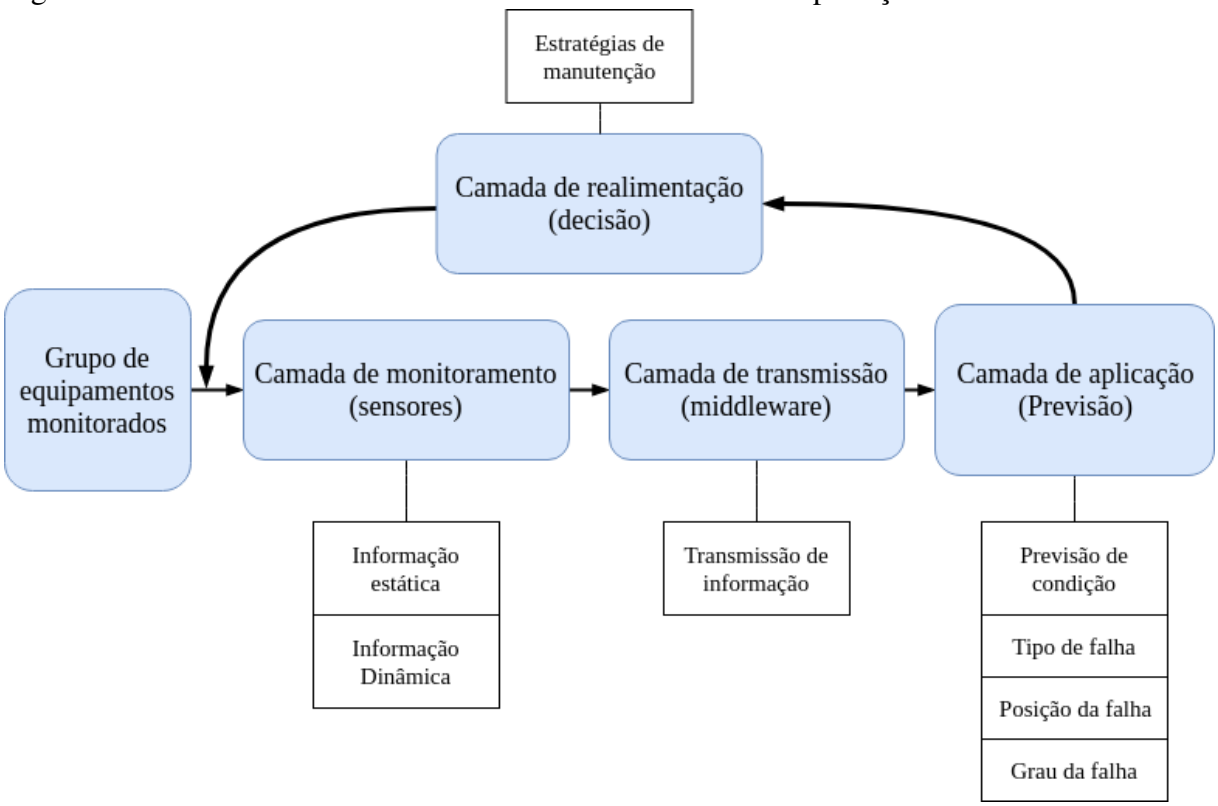
A indústria tem se inserido cada vez mais nesse contexto, o que pode estar sendo a quarta revolução industrial, ou Indústria 4.0 como vem sendo chamada, baseando-se em conceitos de *Business Inteligence*, sistemas ciber-físicos, Internet das Coisas e computação em nuvem (LASI *et al.*, 2014).

Uma forma de aplicar o *IoT* na indústria é demonstrado por Xu *et al.* (2012) com um modelo de sistema de predição de falhas, pois tem tecnologias suficientes e necessárias para uma implementação barata e eficiente, combinando tecnologia de sensores, redes de comunicação e tecnologias de inteligência computacional.

Pode-se destacar características como a percepção abrangente, em que as coisas são coletadas dinamicamente por todos os meios de percepção disponíveis, a transmissão confiável, em que a informação da percepção é transmitida de forma confiável pela rede, e o processamento inteligente, ou seja, a computação inteligente, como a computação em nuvem, que pode ser utilizada para análise e processamento de dados massivos, a fim de implementar um controle inteligente (XU *et al.*, 2012).

O autor ainda propõe uma estrutura funcional de quatro camadas para um sistema de predição de falhas baseado em *IoT*, representado na Figura 3.

Figura 3 – Estrutura funcional de 4 camadas de um sistema de predição de falha



Fonte: Adaptado de Xu *et al.* (2012).

2.2.1 Camada de Monitoramento (Sensores)

A **camada de monitoramento** é a base do sistema, e a principal para monitorar os grupos de equipamentos com informações estáticas e dinâmicas coletadas deles. Essa camada é composta por dispositivos que estarão conectados aos sensores e se encarregarão de, através de protocolos de comunicação sem fio, enviar esses dados para a camada intermediária. Os dados estáticos que compõem a informação podem ser considerados de identificação do equipamento, como nome, modelo, tipo, entre outras a serem determinadas no desenvolvimento, já as informações dinâmicas serão compostas pelos dados coletados dos sensores como vibração, temperatura ou corrente. Com a estrutura do dado montada será passada para a camada intermediária.

2.2.2 Camada de Transmissão (Middleware)

A **camada de transmissão** é uma importante ligação entre a coleta dos dados e o processamento, ela é responsável por transmitir o dado com segurança através da rede, mas além disso nela pode ser aplicado um pré-processamento dos dados antes da transmissão podendo ser utilizado para uma checagem instantânea.

2.2.3 Camada de Aplicação (Previsão)

Na **camada de aplicação** é onde ocorrerá o processamento dos dados coletados, ou seja, poderão ser aplicados os métodos de análise e diagnóstico da falha. Nessa camada vários tipos de algoritmos de processamento podem ser usados para conduzir a predição de falhas, não há um método único que seja útil em qualquer tipo de equipamento, portanto aqui deverá ser aplicado o método que melhor se adequará aos dados coletados e a necessidade do desenvolvedor. Os algoritmos de processamento são classificados na literatura em três tipos principais: os baseados em modelos matemáticos complexos que representam a planta a ser analisada, os que são baseados em processos empíricos e os que são baseados nos dados, que utilizam métodos estatísticos para identificação do sistema e predição de valores ou métodos de inteligência computacional tais como aprendizado de máquina e redes neurais.

2.2.4 Camada de Realimentação (Decisão)

Finalmente, a **camada de realimentação**, é onde o cliente e operador do sistema tem o retorno da predição e toma as decisões sobre o que fazer caso seja necessário uma manutenção, e é por esse motivo que é de realimentação, pois dependendo da ação realizada no equipamento irá alterar os valores coletados mudando a previsão da falha. Ou seja, o sistema se torna de malha fechada, em que o estado ótimo é o de perfeita funcionalidade dos equipamentos.

2.3 Processamento e análise de dados

Como Mobley (2002) explica em seu livro, as técnicas de manutenção preditiva e análise dos dados dependem do tipo de dado a ser coletado, como monitoramento e análise de vibrações, termografia, analise de óleos lubrificantes, ultrassom e testes elétricos.

A predição de valores baseia-se no conceito de que conhecendo o sistema que produz

os dados é possível passar novas entradas e determinar as saídas, nesse caso uma previsão do comportamento futuro da planta em que os dados de entrada do sistema seria o tempo percorrido e a saída os valores medidos e a partir de um limiar definido é possível identificar quando o sistema irá provavelmente ultrapassar esse limiar, constatando uma falha no futuro.

A identificação de sistemas é um procedimento alternativo e a motivação de sua utilização é básica, suponha que estejam disponíveis os sinais de entrada, $u(k)$, e de saída, $y(k)$, de um sistema real qualquer, a identificação de sistemas se propõe a obter um modelo matemático que explique, ao menos em parte e de forma aproximada, a relação existente entre a causa e efeito contida nos dados coletados (AGUIRRE, 2007). A importância da utilização dessas técnicas está relacionado com a necessidade de uma modelagem matemática para o sinal resultante dos dados a serem diagnosticados, pois a partir desse sistema identificado é possível prever dados futuros para inferir uma possível falha. Os modelos utilizados em identificação de sistemas podem ser divididos em três categorias de acordo com suas características.

Caixa-branca: em que são conhecidos, ou previamente determinados, todos os parâmetros e modelagem matemática do sistema, ou seja, é possível controlar todos os parâmetros e determinar matematicamente a saída do sistema.

Caixa-cinza: em que são conhecidos algumas informações ou parâmetros do sistema mas não ele em sua totalidade, o nível de conhecimento pode ser obtido a partir do funcionamento real do sistema, tal como a forma do sinal de saída e através dele escolher um modelo com forma similar para identificação.

Caixa-preta: não utilizam informações conhecidas a priori do sistema a ser identificado, e a identificação é determinada apenas a partir dos dados de entrada e saída do processo. A escolha do método de representação é empírica nos casos mais simples, mas em casos mais complexos, como em sistemas não-lineares, a escolha é crítica.

Determinar o melhor modelo de um conjunto de modelos é normalmente definido a partir dos dados experimentais e pela forma do sinal. Porém, para a aplicabilidade no sistema proposto neste trabalho, o modelo que melhor pode se encaixar nessa situação são os de caixa-preta pois a única informação que pode ser inferida diretamente do sistema de forma automática e predeterminada é a forma dos dados, o sistema se baseia nas incertezas do equipamento e no desgaste do mesmo, portanto não podem ser aplicadas técnicas de caixa-branca pois não é do escopo deste trabalho determinar uma modelagem matemática para a falha de um equipamento.

Para melhor uso das tecnologias relacionadas à *IoT* e computação inteligente, o

modelo de caixa-preta pode ser utilizado com técnicas de inteligência computacional. Uma rede neural é capaz de simular um sistema e prever novos valores a partir de novas entradas e não é necessário conhecer parâmetros do sistema para essa identificação, por esse motivo será utilizado no trabalho a fim de identificar os padrões dos dados e prever valores futuros. Além da rede neural, podem ser aplicadas outras técnicas de identificação de sistemas a fim de prever novos valores.

2.3.1 Estimadores de Séries Temporais

Uma série temporal pode ser definida como uma sequência de observações capturadas ao longo de um período de tempo, na qual a sua principal característica é que valores vizinhos possuem certo grau de dependência (EHLERS, 2005). Ou seja, há características intrínsecas aos valores observados que são relacionadas ao tempo em que foram capturados.

No caso das máquinas rotativas as observações de vibração podem fornecer informações quando ao desgaste com o uso durante o tempo de captura de dados. Como não há como determinar essa relação de maneira analítica a partir dos dados, podem ser utilizados representações como os modelos polinomiais, auto regressivos (AR), auto regressivos com entradas exógenas (ARX) e auto regressivos com médias móveis e entradas exógenas (ARMAX).

O ARX pode ser classificado como modelo com erro na equação por apresentar uma variável exógena representando o erro do modelo, como representado na Equação 2.1 linear nos parâmetros (TAVARES, 2012).

$$[1 - a_1 q^{-1} - \cdots - a_{n_a} q^{-n_a}] y(k) = [b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}] u(k) + e(k) \tag{2.1}$$

Em que $y(k)$ e $u(k)$ são a saída e a entrada, respectivamente, $e(k)$ é a entrada exógena que representa o erro do sistema, n_a e n_b são referente às ordens dos polinômios, no qual a_i, com $i = 1, 2, \cdots, n_a$, e b_k, com $k = 1, 2, \cdots, n_b$, são os parâmetros do modelo, e q^{-1} é o operador de atraso, de que $u(k)q^{-1} = u(k-1)$. Sendo assim a Equação 2.1 pode ser reescrita como segue na Equação 2.2.

$$y(k) = \frac{B(q)}{A(q)} u(k) + \frac{1}{A(q)} e(k) \tag{2.2}$$

Para determinar os parâmetros $A(q)$ e $B(q)$ são utilizados estimadores estatísticos. Aguirre (2007) cita em seu livro que, dentre os mais diversos algoritmos, o estimador de

parâmetros pelo Método dos Mínimos Quadrados (MQ) é o mais conhecido e utilizados por engenheiros e cientistas, servindo de base para muitas técnicas de identificação. O MQ é uma técnica de otimização que determina o melhor ajuste do conjunto de dados, a partir da minimização da soma do quadrado do erro (TAVARES, 2012).

O modelo ARMAX é similar ao ARX, a diferença é que possui uma componente $C(q) = [1 + c_1 q^{-1} + \cdots + a_{n_c} q^{-n_c}]$ em $e(k)$, sendo n_c a ordem do polinômio, este modelo pode ser representado como segue na Equação 2.3.

$$y(k) = \frac{B(q)}{A(q)} u(k) + \frac{C(q)}{A(q)} e(k) \tag{2.3}$$

Os parâmetros do ARMAX podem ser determinados utilizando o estimador preditor baseado no filtro de Kalman, que é um método matemático capaz de executar estimativas a partir do um modelo dinâmico de um sistema sob influência de ruído branco (MARTINS, 2014).

Como o modelo leva em consideração o sinal de erro $e(k)$, este pode ser considerado um sinal ruído branco aleatório e aplicado o filtro de Kalman para estimar os parâmetros do modelo e prever observações futuras.

2.3.2 Método dos Mínimos Quadrados

Uma vez determinado o modelo matemático a ser estimado, o algoritmo do método dos mínimos quadrados traduz-se por um conjunto de funções matemáticas ou equações (FONTE, 1994).

Para aplicação do método Aguirre (2007) supõe que se conhece o valor estimado do vetor de parâmetros $\boldsymbol{\theta}$, e que é cometido um erro ξ ao tentar explicar a observação y a partir do vetor de regressores $\mathbf{x}$ e de $\boldsymbol{\theta}$, na forma da Equação 2.4

$$y = \boldsymbol{x}^T \boldsymbol{\theta} + \xi \tag{2.4}$$

Após N aplicações é produzida a matriz de regressores ϕ e o vetor de erros $\boldsymbol{\xi}$, sendo descrito a forma matricial na Equação 2.5 (AGUIRRE, 2007).

$$\boldsymbol{y} = \boldsymbol{\phi\theta} + \boldsymbol{\xi} \tag{2.5}$$

É interessante que $\boldsymbol{\theta}$ seja de tal forma que reduza $\boldsymbol{\xi}$, para isso pode-se utilizar o somatório do quadrado dos erros, que é um índice que quantifica a qualidade de ajuste de $\boldsymbol{\phi\theta}$ ao vetor $\boldsymbol{y}$ (AGUIRRE, 2007). A fim de minimizar J_{MQ} para $\boldsymbol{\theta}$ deve-se resolver $(\partial J_{MQ}/\partial\boldsymbol{\theta}) = 0$, para isso deve-se substituir a Equação 2.5 em J_{MQ}, este processo está descrito nas Equações 2.6, 2.7 e 2.8.

$$\begin{aligned} J_{MQ} &= \textstyle\sum_{i=1}^{N} \xi(i)^2 = \boldsymbol{\xi}^T\boldsymbol{\xi} = \|\boldsymbol{\xi}\|^2 \\ &= (\boldsymbol{y}-\boldsymbol{\phi\theta})^T(\boldsymbol{y}-\boldsymbol{\phi\theta}) \end{aligned} \tag{2.6}$$

$$\begin{aligned} \tfrac{\partial J_{MQ}}{\partial\boldsymbol{\theta}} &= -(\boldsymbol{y}^T\boldsymbol{\phi})^T - \boldsymbol{\phi}^T\boldsymbol{y} + (\boldsymbol{\phi}^T\boldsymbol{\phi} + \boldsymbol{\phi}^T\boldsymbol{\phi})\boldsymbol{\theta} \\ &= -\boldsymbol{\phi}^T\boldsymbol{y} - \boldsymbol{\phi}^T\boldsymbol{y} + 2\boldsymbol{\phi}^T\boldsymbol{\phi\theta} \end{aligned} \tag{2.7}$$

Igualando a Equação 2.7 a 0, tem-se:

$$\boldsymbol{\theta} = [\boldsymbol{\phi}^T\boldsymbol{\phi}]^{-1}\boldsymbol{\phi}^T\boldsymbol{y} \tag{2.8}$$

É possível utilizar a Equação 2.8 para determinar o vetor de parâmetros a partir do vetor da construção da matriz de regressões $\boldsymbol{\phi}$ inferido do modelo matemático escolhido e do vetor de observações $\boldsymbol{y}$.

2.3.3 Filtro de Kalman

O estimador baseado no filtro de Kalman depende de um conjunto de equações recursivas, que são utilizadas para estimar os estados de um sistema dinâmico, ou seja, os parâmetros do modelo pré-determinado (AGUIRRE, 2007). Primeiro, assume-se que o sistema pode ser descrito pelo modelo linear no espaço de estados descrito na Equação 2.9 que segue.

$$\begin{cases} \boldsymbol{x}_k = & \boldsymbol{\Phi x}_{k-1} + \boldsymbol{\Gamma u}_{k-1} + \boldsymbol{w}_{k-1} \\ \boldsymbol{y}_{k-1} = & \boldsymbol{H x}_{k-1} + \boldsymbol{v}_{k-1} \end{cases} \tag{2.9}$$

Sendo que $\boldsymbol{w}$ e $\boldsymbol{v}$ são variáveis aleatórias independentes, de média nula, e que satisfazem as seguintes relações de covariância $E[\boldsymbol{w}_k\boldsymbol{w}_k^T] = Q_k$, $E[\boldsymbol{v}_k\boldsymbol{v}_k^T] = R_k$ e $\boldsymbol{E[v_i w_j^T]} = 0, \forall i, j$. Nesse contexto, pode-se referir à variável **w** como ruído de processo e à variável **v** como ruído de medição. No qual, $\boldsymbol{\Phi}$ é o modelo de transição de estados, $\boldsymbol{x}$ o vetor de parâmetros a ser estimado, $\boldsymbol{y}$ a saída estimada, $\boldsymbol{H}$ a matriz de leituras, $\boldsymbol{u}$ a sequência de entradas e $\boldsymbol{\Gamma}$ o modelo das entradas de controle.

Com o vetor de estado no instante $k-1$ e os sinais de entrada, é possível ter uma estimativa do estado no instante K. Aguirre (2007) segue o processo de atualização recursiva dos parâmetros, que passa pelas fases de correção, atualização e previsão, no filtro de Kalman, essas fases são descritas pelo sistema de equações que segue em 2.10.

$$\begin{cases} \boldsymbol{K}_k = & \boldsymbol{\Phi P}_{k-1}\boldsymbol{\Phi}^T\boldsymbol{H}^T[\boldsymbol{H\Phi P}_{k-1}\boldsymbol{\Phi}^T\boldsymbol{H}^T + R_k]^{-1} \\ \hat{\boldsymbol{x}}_k = & \boldsymbol{\Phi}\hat{\boldsymbol{x}}_{k-1} + \boldsymbol{\Gamma u}_{k-1} + \boldsymbol{K}_k[y_k - \boldsymbol{H}(\boldsymbol{\Phi}\hat{\boldsymbol{x}}_{k-1} + \boldsymbol{\Gamma u}_{k-1})] \\ \boldsymbol{P}_k = & \boldsymbol{\Phi P}_{k-1}\boldsymbol{\Phi}^T - \boldsymbol{K}_k\boldsymbol{H\Phi P}_{k-1} \end{cases} \tag{2.10}$$

Em que $\boldsymbol{K}$ é o ganho do filtro de Kalman, $\hat{\boldsymbol{x}}$ é o vetor de parâmetros estimados e $\boldsymbol{P}$ é a matriz de covariância. Para simplificar, pode-se usar a mesma nomenclatura do estimador descrito na seção anterior, que pode ser representado como segue em 2.11.

$$\begin{aligned} \boldsymbol{\theta}_k = & \quad \boldsymbol{\theta}_{k-1} \\ y_k = & \quad \boldsymbol{\phi}_k^T\boldsymbol{\theta}_{k-1} + e_k \end{aligned} \tag{2.11}$$

Que corresponde a Equação 2.9 com $\boldsymbol{\Phi} = I$, $\boldsymbol{\Gamma} = 0$, $\boldsymbol{H} = \boldsymbol{\phi_k^T}$ e $\boldsymbol{\theta}_k = \boldsymbol{x}_k$. Aplicando-se o filtro de Kalman 2.10 ao modelo 2.11, chega-se às equações recursivas descritas em 2.12.

$$\begin{cases} \boldsymbol{K}_k = & \boldsymbol{P}_{k-1}\boldsymbol{\phi}_k[\boldsymbol{\phi}_k^T\boldsymbol{P}_{k-1}\boldsymbol{\phi}_k + R_k]^{-1} \\ \hat{\boldsymbol{\theta}}_k = & \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{K}_k[y(k) - \boldsymbol{\phi}_k^T\hat{\boldsymbol{\theta}}_{k-1}] \\ \boldsymbol{P}_k = & \boldsymbol{P}_{k-1} - \boldsymbol{K}_k\boldsymbol{\phi}_k^T\boldsymbol{P}_{k-1} \end{cases} \tag{2.12}$$

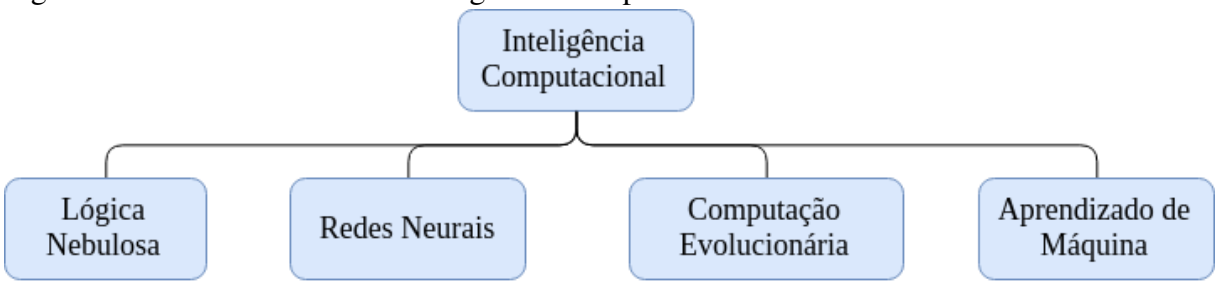
Ao aplicar as equações recursivas à cada iteração, percorrendo a série de observações, é possível estimar o vetor de parâmetros $\boldsymbol{\theta}$.

2.3.4 Inteligência Computacional

Goldschmidt (2010) define Inteligência Computacional (IC) como uma ciência multidisciplinar que busca desenvolver e aplicar técnicas computacionais que simulem o comportamento humano em atividades específicas. Na Figura 4 são apresentados os principais paradigmas da IC, pode-se notar que IC envolve estudos e aplicações inspirados nos fenômenos naturais.

No livro "Uma Introdução à Inteligêcia Computacional: fundamentos, ferramentas e aplicações", de Goldschmidt (2010), é explicado que a **Lógica Nebulosa** se trata de um

Figura 4 – Uma taxonomia de Inteligência Computacional



Fonte: Adaptado de (GOLDSCHMIDT, 2010).

paradigma que modela o raciocínio incerto do pensamento humano, tal como conceitos de muito, pouco, quente, frio, alto, médio, baixo, com mecanismos para manipular e tratar esse tipo de informação imprecisa e subjetiva, fornecendo respostas aproximadas para questões baseadas nesses conhecimentos inexatos. Já a **Rede Neural (RN)** se trata de modelos computacionais inspirados no funcionamento e na estrutura de um cérebro e suas conexões sinápticas, e buscam reproduzir características humanas como aprendizado, associação, generalização e abstração. A **Computação Evolucionária** trata de uma área interdisciplinar com algoritmos inspirados no princípio da evolução natural das espécies proposto por Charles Darwin e recombinação genética e mutação. O **Aprendizado de Máquina** possui o mesmo propósito que as redes neurais, porém não se baseia necessariamente no comportamento humano, e sim em modelos matemáticos e estatísticos para reconhecimento de padrões e classificação de dados.

A aplicação de métodos de IC normalmente resultará em soluções do tipo caixa-preta para os problemas, exatamente por não ser uma solução analítica e sim baseada na aprendizagem inferida dos dados, embora seja eficiente, nem sempre é aceitável por alguns usuários devido à falta de clareza do sistema gerado.

Um ponto forte para a utilização de soluções em IC é sua capacidade superior em modelar sistemas complexos do mundo real, que muitas vezes são intratáveis por meio de métodos clássicos, o que justifica sua popularidade e ampla utilização por profissionais e pesquisadores (FULCHER, 2008).

2.3.5 Redes Neurais Artificiais

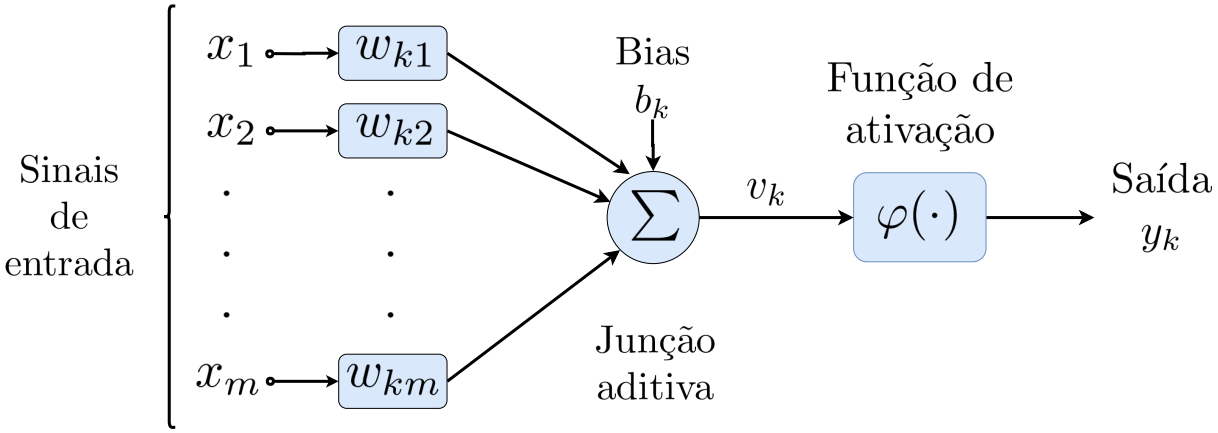
Uma rede neural pode ser definida como um processador paralelamente distribuído, constituído de unidades de processamento simples nomeadas de neurônios, que tem a capacidade de armazenar conhecimento experimental e torná-lo disponível para uso, em que o conhecimento é adquirido pela rede através de um processo de aprendizagem e as conexões entre os neurônios

são utilizadas para armazenar o conhecimento adquirido (HAYKIN, 2007). O algoritmo de aprendizagem possui a função de adaptar os pesos das conexões entre as unidades de processamento da rede de uma forma ordenada.

Haykin (2007) mostra em seu livro algumas vantagens da utilização de uma rede neural, dentre elas pode-se citar: a **não-linearidade** inerente da rede, por ser baseada em unidade de processamento não lineares, que é útil no caso de o sistema físico que gerou os dados não seja um sistema linear; a **capacidade de mapeamento de entrada-saída**, em um treinamento supervisionado a rede aprende os padrões do sistema a partir de um conjunto de entradas inseridas no sistema e a saída dele; a **adaptabilidade**, em que a rede é capaz de se adaptar a mudanças no ambiente; o **contexto**, onde cada neurônio da rede é potencialmente afetado pela atividade de todos os outros neurônios, portanto a informação contextual intrínseca nos dados é tratada pela rede neural; o **paralelismo**, a sua natureza maciçamente paralela a fez ser potencialmente rápida na computação de certas tarefas.

A unidade de processamento de uma rede neural é fundamental para a operação da mesma, o neurônio artificial segue a base de uma modelagem matemática de um neurônio biológico, representado na Figura 5 (HAYKIN, 2007).

Figura 5 – Modelo não-linear de um neurônio



Fonte: Adaptado de Haykin (2007).

Podem ser identificadas três partes principais da formação de um neurônio artificial, primeiro um conjunto de sinapses caracterizado por um peso w_{km} próprio de cada ligação de um neurônio k, esse peso irá ponderar o sinal de entrada x_m no neurônio e é ele que armazena a informação aprendida pela rede, que passa por um somador para integrar todas as entradas ponderadas pelas sinapses. Juntamente a isso, a segunda parte é um bias b_k, que é responsável por aumentar os graus de liberdade permitindo uma melhor adaptação, e, por fim, o sinal v_k,

saído do somatório, passa por uma função de ativação $\varphi(\cdot)$ qualquer, usualmente uma sigmoide, responsável por determinar a saída y_k do neurônio. Matematicamente, pode ser representado pela Equação 2.13.

$$y_k = \varphi(\sum_{j=1}^{m} w_{kj} x_j + b_k) \tag{2.13}$$

A função de ativação $\varphi(\cdot)$ tem a saída comumente entre $[0,1]$ ou $[-1,1]$, dependendo do padrão adotado; os dados de entrada são normalizados para o funcionamento independente da amplitude do sinal, a forma da função pode variar sendo geralmente utilizadas as funções degrau, rampa ou sigmoide, podendo ativar a sinapse ou mandar um sinal atenuado, no caso de uma sigmoide ou rampa.

2.3.6 Arquitetura de Rede Neural

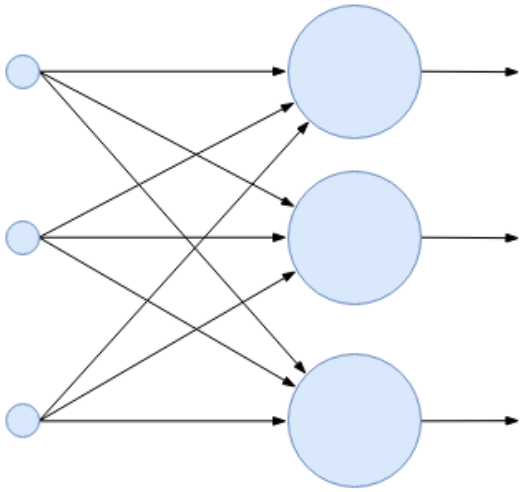
A maneira como os neurônios de uma rede neural está organizada está intimamente ligado com o algoritmo de aprendizagem utilizado para o treinamento da rede (HAYKIN, 2007). Em geral, é possível identificar três topologias fundamentalmente diferentes, as redes **alimentadas adiante com camada única**, **alimentadas diretamente com múltiplas camadas** e finalmente as **redes recorrentes**.

O tipo de rede neural de uma camada pode ser considerada uma simplificação da rede de múltiplas camadas, em que a rede neural é composta apenas de uma camada em que todos os neurônios que recebem as entradas são também saídas do sistema, um exemplo dessa topologia pode ser visto na Figura 6.

Já na topologia de múltiplas camadas, a rede neural segue um padrão de uma camada final para saídas e um conjunto de camadas intermediárias denominadas de camada de neurônios oculta. Um exemplo pode ser visualizado na Figura 7, nele está representado uma RN com duas camadas intermediárias e uma camada de saída, é dita que essa camada está totalmente conectada, pois todos os nós de uma camada estão conectados à todos os nós da camada adjacente, caso alguma das conexões sinápticas esteja faltando é dita que ela está parcialmente conectada (HAYKIN, 2007).

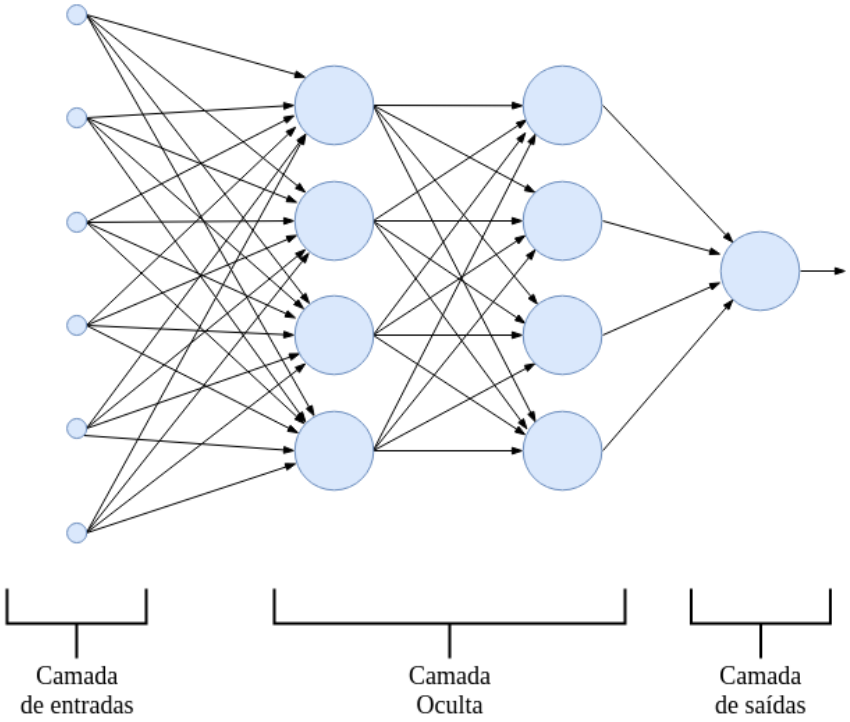
A habilidade de os neurônios ocultos extraírem informações de ordem elevada é valiosa quando a quantidade de entradas é grande, e quando mais neurônios na camada oculta maior a capacidade de a rede armazenar informações sobre o padrão dos dados, porém aumenta

Figura 6 – Rede alimentada adiante com única camada



Fonte: Elaborado pelo autor

Figura 7 – Rede alimentada diretamente com múltiplas camadas



Fonte: Elaborado pelo autor

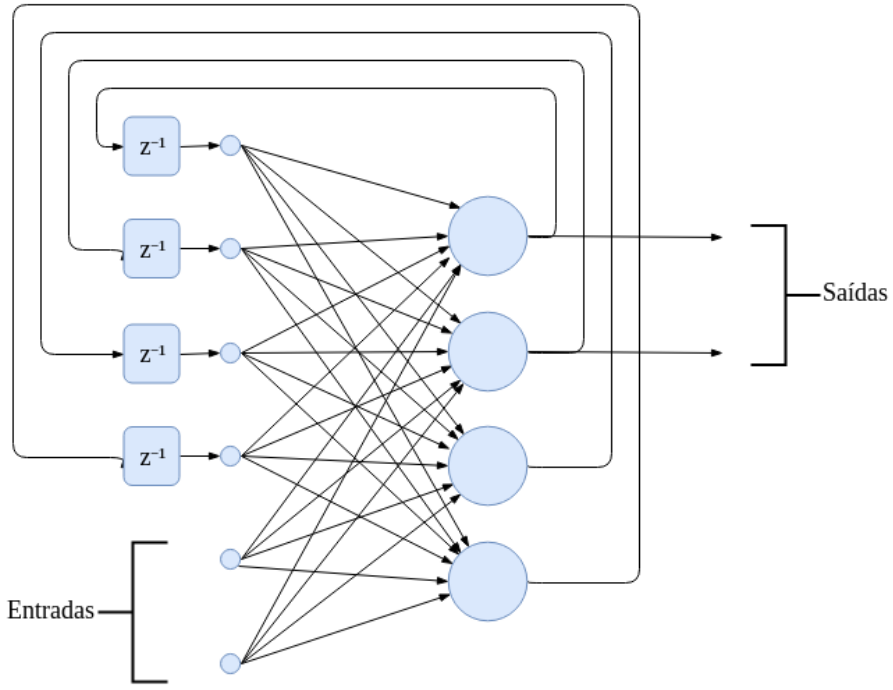
a complexidade computacional do treinamento na medida que aumenta as sinapses (HAYKIN, 2007).

Existem informações e dados que são sequenciais por natureza, como por exemplo palavras, frases e enredos, em uma frase palavras soltas desordenadas não fazem sentido, porém quando organizadas de forma que venham a ter um contexto possuem todo o significado. E esse tipo de informação sequencial não se limita apenas a frases e textos, mas também a séries temporais como preços de ações, taxas de juros ou dados capturados de um sensor durante um período de tempo, o contexto desses dados está relacionado com o momento que foi capturado e isso diz muito sobre as informações que esses dados representam.

Para trabalhar com esse tipo de dados é utilizado o terceiro tipo de Rede Neural Artificial (RNA), a recorrente. As Redes Neurais Recorrentes podem ter uma ou mais camadas, mas a sua particularidade reside no fato de que se tem conexões que partem da saída de uma unidade de processamento em direção a outra unidade da mesma camada ou de uma camada anterior a esta. Estes tipos de conexões permitem a criação de modelos que levam em consideração aspectos temporais e comportamentos dinâmicos, em que a saída de uma unidade depende de seu estado em um tempo anterior (OSÓRIO; VIEIRA, 1999).

Na Figura 8 está representada uma rede neural recorrente com uma camada oculta, em que não há entrada conectada diretamente a ela, a presença de laços de realimentação tem um impacto profundo na capacidade de aprendizagem da rede e no seu desempenho, além disso nas sinapses de realimentação há a presença de elementos de atraso unitário, representados por z^{-1}, que resulta em um comportamento dinâmico não-linear, admitindo-se que a rede contenha unidades não-lineares (HAYKIN, 2007).

Figura 8 – Rede Neural Recorrente com Neurônios Ocultos



Fonte: Adaptado de Haykin (2007)

3 TRABALHOS RELACIONADOS

Com o objetivo de entender e levantar diversos problemas de análise de vibração e manutenção preditiva foram analisados trabalhos que serviram de inspiração para o desenvolvimento deste. Existem alguns trabalhos que utilizam componentes de baixo custo a fim de fornecer uma alternativa barata para manutenção preditiva em equipamentos rotacionais.

Mourão e Junior (2011) apresentam em seu projeto um modelo de sistema de manutenção preditiva para sistemas de refrigeração utilizando dados experimentais e simulação de entradas e saídas de sensores. O autor utiliza leis físicas e métodos de estatística para modelar o sistema e prever valores futuros, a fim de determinar uma possível falha. O trabalho serviu de inspiração por comparar métodos de diagnóstico baseados em modelagem do sistema com métodos estatísticos e de aprendizagem de máquina.

A diferença do projeto de Mourão e Junior (2011) com o presente trabalho está na falta de conhecimento do modelo do sistema, por esse motivo serão utilizadas técnicas de identificação de sistemas caixa-preta. Além disso, foi desenvolvido um protótipo de *data logger* para captura dos dados.

Soares (2015) desenvolveu em seu trabalho um *data logger*, utilizando tecnologias como Arduino[1] e acelerômetro, responsável por capturar uma sequência de observações de vibração durante um período de tempo pré-determinado, e armazenar essa leitura de vibração em um cartão de memória. Após a captura dos dados, eles são repassados pelo cartão para um computador, onde é realizada a análise do sinal.

González (2014) mostra em sua teste, "Desenvolvimento de um protótipo analisador de vibração de baixo custo para uso em manutenção preditiva", um protótipo de um analisador de vibração utilizando como microcontrolador o Arduino Due, versão com maior poder de processamento e memória do Arduino, juntamente com um acelerômetro para capturar as observações e realizar um pré-processamento do sinal. Diferente de Soares (2015), o protótipo de González (2014) se comunica com o computador via *Universal Serial Bus* (USB), no qual os dados são repassados para o MATLAB, onde é realizada a análise final dos sinais.

Ribeiro (2017) desenvolveu um protótipo de baixo custo, para captura de dados de vibração de máquinas rotacionais, utilizando o sensor ótico de um *mouse* de computador. O *mouse* é conectado à um computador, que faz todo o processamento do sinal, e a vibração é referente ao deslocamento capturado pelo sensor ótico direcionado para uma fita refletora no

[1] https://www.arduino.cc/

motor. O *software* no computador captura o movimento do mouse e realiza a análise do sinal para diagnosticar o motor.

Porém, estes protótipos não fornecem dados em tempo real ou sem auxílio de um operador para capturar os dados e passar para um *software* de análise. Xu *et al.* (2012) propõe uma abordagem para a utilização de tecnologias *IoT* aplicada à indústria, a taxonomia proposta é um modelo de sistema *IIoT* para manutenção preditiva, e serviu de inspiração na montagem da plataforma desenvolvida neste trabalho.

Diferente de Mourão e Junior (2011), González (2014) e Ribeiro (2017), a plataforma desenvolvida segue a ideia de *IIoT* e realiza o processamento do sinal em nuvem, fornecendo ao operador uma *dashboard* para visualização dos dados e tomada de decisões, juntamente com um protótipo de *data logger* conectado à internet que captura os dados em tempo real e manda para a plataforma. Este tipo de sistema fornece mais segurança ao operador, por distanciá-lo do maquinário pesado, evitando acidentes.

4 METODOLOGIA

A proposta deste trabalho é desenvolver uma plataforma *IoT* para captura de dados e processamento na nuvem em uma aplicação de manutenção preditiva, para esse propósito, foi utilizada uma arquitetura semelhante a proposta por Xu *et al.* (2012) mostrada na Figura 3. O desenvolvimento foi dividido em partes, primeiro a captura dos dados, após isso o pré-processamento dos dados e transmissão para servidor em nuvem, o terceiro momento será o processamento dos dados para identificação dos padrões e predição de valores futuros, e por fim a parte de aplicação em que os dados poderão ser visualizados em forma de gráficos e extraídas informações úteis do processamento.

Para validar o modelo de sistema a ser utilizado, e estudar as técnicas de estimadores e preditores, foram realizadas simulações em *MATLAB*, utilizando um banco de dados de leituras contínuas de vibração, temperatura e corrente de um motor com caixa de engrenagens disponibilizado *online* por Martin *et al.* (2018), e posteriormente este mesmo banco de dados foi inserido no sistema desenvolvido, a fim de validar os estimadores.

O banco de dados utilizado faz parte do projeto GOTIX[1], que está localizado no GIPSA-lab, e possui como objetivo caracterizar os defeitos em sistemas mecânicos acionados eletricamente. O experimento atual é à longo prazo com desgaste natural de engrenagens pelo uso (MARTIN *et al.*, 2018). Todos os dados produzidos são compartilhados livremente sob demanda.

Este capítulo vai seguir a seguinte estrutura: primeiro será apresentado como foram realizadas as simulações referentes ao trabalho, em seguida, será exposto como a plataforma foi construída seguindo o modelo baseado na Figura 3, mostrando as camadas de monitoramento, transmissão, aplicação e decisão.

Uma lista completa de todas as tecnologias de *software* que foram utilizadas para desenvolver este projeto podem ser encontradas na Tabela 8 no Apêndice A.

4.1 Simulação

O *software* utilizado para realizar as simulações foi o *MATLAB* R2015a. O banco de dados consiste em uma sequência de arquivos organizados temporalmente, na qual cada um contém uma gravação de 80 segundos da vibração por um acelerômetro espaçadas de 25 horas

[1] http://www.gipsa-lab.grenoble-inp.fr/projet/gotix/

cada feitas com o acelerômetro B&K 4391 e usando o sistema RACAL, tendo a captura deste banco de vibração iniciado em janeiro de 2003 com 2795 horas de funcionamento contínuo do equipamento e parado em 2007 com 4865 horas de funcionamento (MARTIN *et al.*, 2018), o que resulta em um total de 2070 horas de experimento.

Para que fosse possível analisar o aumento gradual do desgaste, foi então calculado o valor RMS de cada uma das observações, que pode ser descrito utilizando a Equação 4.1 abaixo (CAESARENDRA; TJAHJOWIDODO, 2017).

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \tag{4.1}$$

Em que N é o número total de amostras e x é a amostra na posição i. Obtendo então 85 amostras de valor RMS da vibração com separação de 25 horas entre cada uma. Esses valores foram normalizados de acordo com a sensibilidade e as informações do sensor utilizado.

O equipamento funcionou por 3500 horas quando o acoplamento de engrenagem que liga o eixo do motor ao eixo da caixa de engrenagem quebrou repentinamente, criando uma fissura helicoidal no eixo de engrenagem na sua posição de acoplamento (COMBET *et al.*, 2004). Combet *et al.* (2004) explicam que isso ocorreu enquanto estavam testando novas engrenagens sob carga em condições normais por 300 horas.

Por esse motivo, para este estudo de caso, foi selecionada a primeira falha e utilizados os dados de captura a partir de 15 dias antes da falha, até 4 dias antes da falha, a fim de utilizar os estimadores para prever os valores dessa semana e verificar se o tempo estimado para ocorrer a falha condiz com o tempo real em que a falha aconteceu.

Além do valor RMS, um parâmetro chamado *delta* RMS pode ser utilizado para analisar a tendência do sinal de vibração, e pode ser determinado pela diferença de dois valores RMS consecutivos. A suposição por trás desse parâmetro é que se acontece um dano na engrenagem ele vai aumentar mais rapidamente se comparado com os níveis normais de vibração sem danos (IGBA *et al.*, 2016).

Para estimar os valores RMS, a fim de prever novos valores e determinar o grau de severidade baseado no gráfico do Anexo A, foram utilizados três métodos: estimador de função **polinomial** pelo método dos mínimos quadrados, estimador dos parâmetros do **modelo ARX** pelo método dos mínimos quadrados, estimador dos parâmetros do **modelo ARMAX** pelo método do filtro de Kalman, e uma **rede neural recorrente**.

Estes métodos foram escolhidos pois era necessário um modelo que seguisse o formato da curva de valores RMS, pois quando um equipamento está saudável o gráfico tende a seguir uma linha reta ou uma curva suave, mas quando o equipamento começa a dar sinais de falha as leituras começam a ficar instáveis e tendem a subir bastante em uma direção, dependendo da natureza da falha (IGBA *et al.*, 2016).

O modelo polinomial foi utilizado por seguir de forma suave essa tendência, porém atua como uma interpolação podendo divergir bastante caso a previsão seja de muitos valores à frente, mas é uma forma de se lidar com dados não sazonais que contenham uma tendência (EHLERS, 2005), segue na Equação 4.2 o modelo utilizado, esse formato foi determinado de forma empírica por tentativa e erro. O método dos mínimos quadrados estima os coeficientes a_i.

$$y = a_1 x^2 + a_2 x + a_3 1 + a_4 \cos x \tag{4.2}$$

O modelo ARX foi escolhido por ele ser capaz de levar em consideração valores de entradas e saídas passadas para prever uma nova leitura, absorvendo características intrínsecas das séries temporais. Além de ser uma técnica de identificação de sistemas bastante utilizada para determinar sistemas de controle do tipo caixa-preta (AGUIRRE, 2007), o modelo utilizado possui $n_a = 3$ e $n_b = 3$ (ver Equação 2.1), também determinado de forma empírica.

A rede neural recorrente foi escolhida por assimilar funções com comportamento dinâmico e não-linear, levando em consideração a memória de observações passadas e o tempo que foi observado, tendo um desempenho eficiente quando os dados utilizados para treinar a rede são séries temporais (HAYKIN, 2007).

4.1.1 Avaliação do desempenho dos estimadores

Para avaliação do desempenho das previsões utilizando os modelos polinomial, ARX, ARMAX e *RNN*, foram consideradas as seguintes métricas, sendo N o número de observações de treinamento, y a série de observações e $\hat{y}$ a série estimada.

***Mean Absolute Error* (MAE):** O erro médio absoluto é um parâmetro que mede o valor médio do erro entre as séries observadas e estimadas, cuja equação pode ser determinada como segue em 4.3 (CAMELO *et al.*, 2017).

$$\text{MAE} = \frac{1}{N} \sum_{t=1}^{N} |y(t) - \hat{y}(t)| \tag{4.3}$$

***Root Mean Squared Error* (RMSE):** A raiz quadrada da média do erro ao quadrado é um parâmetro que determina o erro de previsão da curva estimada. Pode ser determinada pela Equação 4.4 que segue, essa métrica penaliza mais os maiores erros (CAMELO *et al.*, 2017).

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y(t)-\hat{y}(t))^2} \tag{4.4}$$

***Mean Absolute Percentage Error* (MAPE):** O erro médio absoluto em porcentagem tem a vantagem de ter um resultado de rápido entendimento, retornando a porcentagem de erro em relação ao valor esperado da série estimada. Este parâmetro pode ser determinado na forma da Equação 4.5 a seguir.

$$\text{MAPE} = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{\hat{y}(t)-y(t)}{y(t)}\right| \cdot 100 \tag{4.5}$$

Com esses parâmetros é possível determinar qual dos estimadores possuiu melhor desempenho durante os experimentos realizados. A acurácia do modelo é complementar ao erro.

Mais detalhes sobre os métodos de estatística de erro que serão usadas podem ser vistas nos trabalhos de Montgomery *et al.* (2015) e Cochran (1977).

4.2 Camada de monitoramento

O *hardware* utilizado para a finalidade de captura de dados foi pesquisado a fim de obter o menor custo para uma aplicação não-intrusiva, ou seja, a instalação do dispositivo no equipamento não exige a necessidade de modificar o circuito operacional. Os sensores utilizados no sistema estão listados na Tabela 2 com o preço médio de mercado.

Tabela 2 – Tabela de sensores utilizados.

Sensor	**Referência**	**Preço**
Acelerômetro de 3 eixos	MMA8452Q	R$ 29,00
Sensor de corrente não invasivo	SCT-013	R$ 49,00
Sensor de temperatura	DS18B20	R$ 17,00

Fonte: Adaptado de Robocore (2018).

Para efeitos comparativos, foram pesquisados alternativas já utilizadas em meios convencionais de manutenção preditiva, para a análise de vibrações é comum utilizar equipamentos como o medidor de vibrações SDL800 da Extech[2], que custa cerca de R$ 7100,00 pela

[2] http://www.extech.com/sdl800/

TecnoFerramentas[3], e ainda depende de outro equipamento de análise pois apenas grava os dados em um cartão de memória, uma outra solução já encapsulada para análise e a medição dos dados de vibração é o analisador de vibrações VIB100 da Hottec, que custa cerca de R$ 28.000,00 pela Hottec[4].

O Acelerômetro MMA8452Q é um sensor digital de baixo consumo energético e $3.3V$ de tensão de alimentação, com leitura de 3 graus de liberdade, possui 12 bits de resolução, o que resulta em até 4096 níveis de quantização, e escalas selecionáveis de $\pm 2g / \pm 4g / \pm 8g$ com dados filtrados por um filtro passa alta, e se comunica através do protocolo *Inter-Integrated Circuit* (I2C) (ROBOCORE, 2018).

O sensor de corrente alternada não invasivo SCT-013 possui um limite de corrente de entrada de 0 a $100A$ e corrente de saída de 0 a $50mA$, como geralmente os dispositivos leem valores de tensão é necessário um circuito complementar, em que uma carga de resistência deve ser dimensionada, e a partir dela é possível controlar a precisão de leitura (ROBOCORE, 2018). Para o dimensionamento da carga pode ser realizado o cálculo descrito nas Equações 4.6 e 4.7.

$$i_s = \frac{\sqrt{2} \cdot i_{max}}{N} \tag{4.6}$$

$$R_c = \frac{U_s}{i_s} \tag{4.7}$$

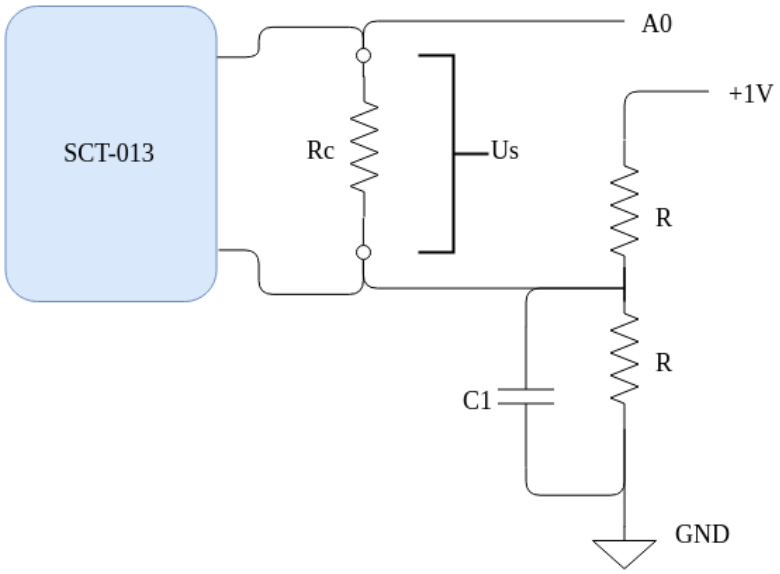
Em que i_{max} é a corrente máxima a ser medida pelo sistema, N é o número de espiras do sensor, que no caso do SCT-013 é 2000, R_c é o valor da resistência de carga a ser colocada no circuito e U_s é a tensão máxima de saída definida, o circuito utilizado para medir a corrente pode ser visualizado na Figura 9.

O sensor de temperatura DS18B20 possui um limite de leitura de -55°C à 125°C, com precisão 0.5°C para valores entre -10°C e 85°C, tensão de alimentação $3.3V$ e comunicação pelo protocolo I2C, com resolução de 9 ou 12 bits, dando de 512 a 4096 níveis de quantização (ROBOCORE, 2018).

[3] https://www.tecnoferramentas.com.br/medidor-de-vibracaoregistrador-de-dados-capacidade-aceleracao-05-a-1999-ms2-velocidade-05-a-1999-mms-deslocamento-0003-a-1999mm-extech-sdl800-500083/p

[4] http://www.hottec.com.br/analisadordevibracaoportatil.php

Figura 9 – Circuito complementar.



Fonte: Elaborado pelo autor.

O dispositivo a ser utilizado para enviar os dados coletados dos sensores será um NodeMCU modelo ESP12E, que possui um custo de cerca de R$ 25,00[5]. O NodeMCU é uma solução encapsulada para *IoT*, que mesmo tendo o propósito de prototipagem é bastante utilizado em soluções finais por pequenas empresas. Ele possui um ESP8266 embutido, que pode ser programado nativamente em Lua, mas já existem *frameworks* que permitem a utilização da popular IDE de desenvolvimento do Arduino para programar o NodeMCU em Wiring (GROKHOTKOV, 2018), que é a linguagem de programação utilizada pelo Arduino. Porém, com a utilização dessa IDE a pinagem do NodeMCU precisa ser remapeada, e pode ser verificada na Figura 10.

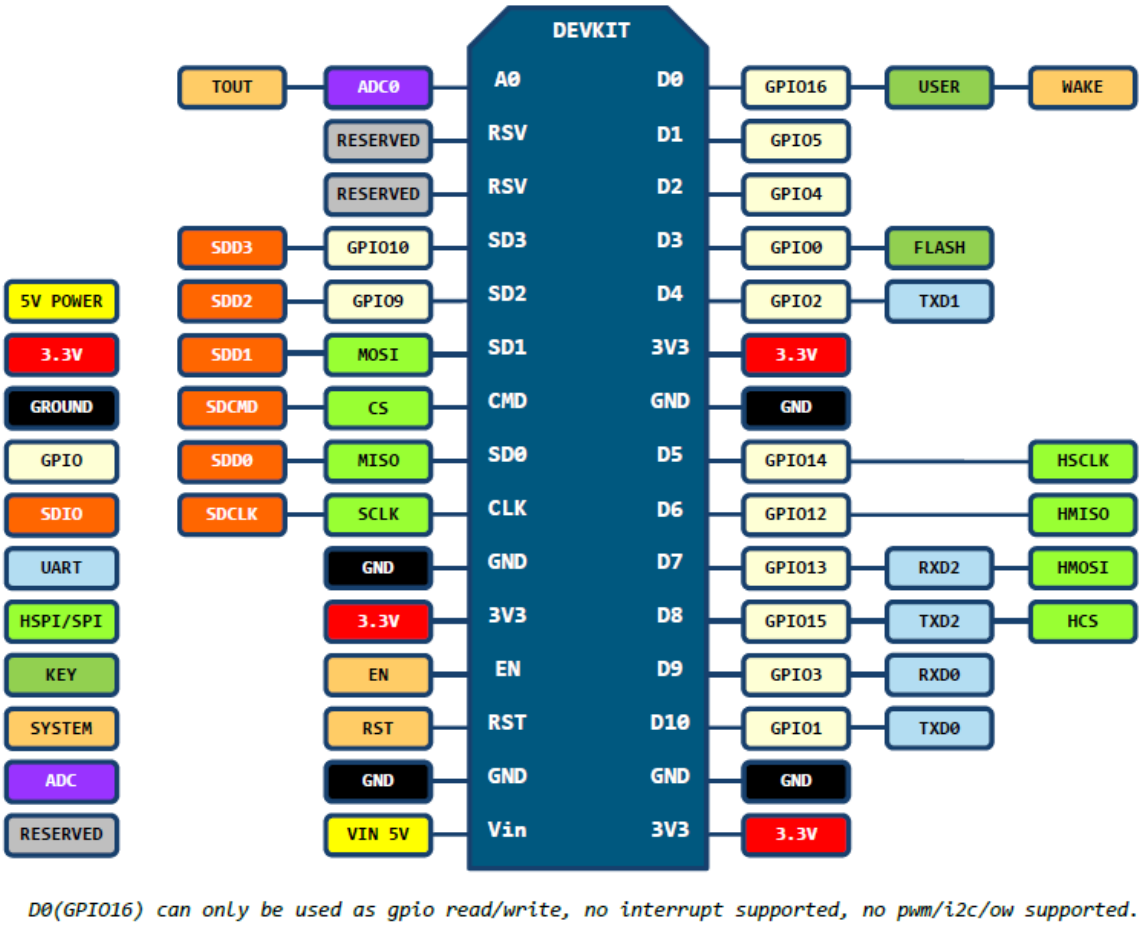
O NodeMCU possui uma configuração de pinos específica que vem referenciada na placa, na Figura 10 a placa está representada pelo retângulo azul.

Como a IDE do Arduino foi construída para outra placa, a configuração de pinos é diferente, os pinos referentes ao Arduino que são utilizados no código como sendo do NodeMCU estão representados na Figura 10 pelos nomes GPIO.

Exemplificando, onde no NodeMCU está referenciado como D0, no código escrito na IDE utilizada pelo Arduino é necessário usar 16, e assim segue por todas os outros pinos. A única exceção é a porta analógica, como só existe uma, a referência é a mesma, A0.

[5] https://lista.mercadolivre.com.br/nodemcu

Figura 10 – Pinagem do NodeMCU remapeada.



Fonte: (MUKHERJEE, 2016).

O ESP8266 embutido no NodeMCU possui suporte à *wireless* padrão IEEE 802.11 b/g/n/, o WiFi, com antena interna, possibilitando o envio dos dados coletados sem necessidade de cabos para comunicação com a unidade que irá transmitir os dados para a nuvem. Além disso possui um Conversor Analógico-Digital (ADC) de 10 bits de precisão, ou seja, com uma saída de 0 a 1024 níveis de quantização com entrada de 0 a $1V$, que é suficiente para o sensor de corrente, a tensão de operação é de 3.3V, com clock de operação de $80MHz$ com suporte até $160MHz$, e uma taxa de amostragem máxima de $2.5KHz$ no ADC e uma memória flash máxima de 16Mb (ELECTRODRAGON, 2017).

Pode ser verificado na Figura 10 que o NodeMCU possui uma entrada analógica e 13 pinos digitais, que serão suficientes para conectar todos os sensores, pois o sensor de corrente irá utilizar a entrada analógica, o acelerômetro e o sensor de temperatura são sensores digitais que se comunicam através do protocolo I2C, que permite a utilização de vários sensores pelo mesmo barramento, diminuindo a quantidade de fios e conexões necessárias.

O dispositivo foi projetado de forma modular, ou seja, cada sensor é independente e

pode ser adicionado ou removido sem que pare de funcionar. Portanto, o custo final do dispositivo é equivalente à soma do NodeMCU com o(s) sensor(es) utilizado(s). Para gerar os resultados do Capítulo 5, foi utilizado apenas o acelerômetro descrito na Tabela 2, a fim de capturar as observações de vibração. Fotos do hardware montado podem ser vistas no Apêndice B.

De forma periódica, o dispositivo envia as leituras de todos os sensores conectados, pré-programado para à cada 25 horas enviar o valores de 1 segundo de amostragem. Podendo ser configurado pela plataforma WEB. Também é possível realizar as leituras individuais sob demanda, sendo requisitado pela plataforma.

4.3 Camada de transmissão

O dispositivo é o responsável por transmitir os dados capturados utilizando o protocolo *Message Queuing Telemetry Transport* (MQTT) para o servidor. Como Yuan (2018) define, o protocolo MQTT segue um padrão de publicação e assinatura. Nele são definidas duas entidades: o *message broker* e os clientes. O *broker* é responsável por receber as mensagens enviadas pelos clientes e redirecionar essas mensagens aos clientes interessados.

O MQTT segue a estrutura de tópicos, como se fossem diretórios, os clientes podem assinar e publicar nos tópicos, quando um cliente está assinado a um tópico ele recebe todas as mensagens publicadas neste tópico, um cliente não precisa estar assinado a um tópico para publicar nele (YUAN, 2018).

Essa estrutura de tópicos permite uma liberdade para o desenvolvedor separar e associar cada leitura a um único tópico significativo para aquela leitura.

No servidor se encontra o *broker* que é responsável por receber as mensagens enviadas pelo dispositivo e retransmitir para a API, que fará o processamento dos dados, o *broker* também transmite mensagens que são enviadas pela API para o dispositivo.

As mensagens transmitidas pela API são informações de configuração, para que o *datalogger* (dispositivo que captura os dados) possa operar periodicamente como determinado pelo usuário na plataforma WEB, e requisição de leituras, caso haja necessidade de uma nova leitura fora do intervalo determinado.

Antes dos dados serem enviados para o *broker*, o dispositivo faz um pré-processamento calculando o valor RMS da leitura de vibração. Devido as limitações de memória do NodeMCU, só é possível capturar até 1.5 segundos. O qual é suficiente para capturar todo o ciclo de operação da máquina sem perder informações na amostragem.

4.4 Camada de aplicação

Foi desenvolvida uma *Application Programming Interface*, ou Interface de Programação de Aplicação (API) com *NodeJS*, que é responsável por todo o processamento e comunicação com o banco de dados. A sua construção foi feita seguindo o padrão *RESTful*, em que as requisições são do modelo *Create, Read, Update and Delete* (CRUD). O projeto é modular, fazendo com que seja possível a integração de novas funcionalidades sem quebrar a execução atual.

Para que o *datalogger* consiga enviar as leituras para o banco de dados, é necessário que o usuário cadastre as informações na plataforma, criando, se já não houver, o motor referente ao equipamento da planta, e no objeto do motor cadastrar os sensores acoplados a ele. Cada sensor possui uma referência individual. Feito isso, ficará disponível na plataforma um código de identificação do sensor, que deve ser colocado no código do *datalogger* como primeiro diretório do tópico MQTT a ser enviado seguido pelo tipo de leitura, por exemplo: *"123abc/vibration"*. Isso permite o sistema identificar a qual motor da planta e a qual sensor a leitura pertence.

A API está assinada à todos os tópicos do *broker*, recebendo assim todas as mensagens enviadas pelo *datalogger*, ou qualquer outro que venha a ser adicionado ao sistema. O projeto permite escalabilidade da planta podendo ser adicionados inúmeros sensores, sendo limitado apenas ao poder de processamento do servidor.

O processamento dos dados é feito sob demanda, ou seja, quando a aplicação WEB requisita ao servidor visualizar os dados de previsão, é então criado uma instância que realiza os cálculos e retorna os resultados para a aplicação.

Como *Javascript*, que é a linguagem de programação utilizada para desenvolver com NodeJS, não é otimizada para cálculos científicos, foram utilizados scripts feitos com *Python* utilizando as bibliotecas *Numpy*, para operações matemáticas com matrizes, e *Pyrenn*, para criação da rede neural recorrente.

A biblioteca *Numpy* possui algumas similaridades com *Matlab*, o que possibilitou portar o código realizado nas simulações para a API. Portanto, os mesmo métodos utilizados nas simulações estão disponíveis na plataforma.

Na API existem hoje 6 *endpoints*, que são os caminhos a que são realizados as requisições CRUD, porém apenas 3 são CRUD, nas 3 responsáveis pelos cálculos apenas realiza requisição do tipo POST (cria). Como está descrito abaixo:

***/motors*:** Cria, atualiza, deleta e retorna os objetos que armazenam informações do

motor da plata.

***/sensors*:** Cria, atualiza, deleta e retorna os objetos que armazenam informações do sensor, está ligado com os motores de forma N sensores para 1 motor. Para criar um novo sensor é necessário passar a referência do motor.

***/readings*:** Cria, atualiza, deleta e retorna os objetos contendo informações de captura. Esta ligado de forma N leituras para 1 sensor, é necessário passar a referência do sensor para adicionar um valor a lista de valores capturados por aquele sensor.

***/polinomial*:** Executa o *script* que realiza a estimação polinomial e retorna a lista de valores estimados, ao efetuar uma requisição do tipo POST (cria), passando como parâmetros o sensor e o tempo de horas para prever.

***/arx*:** Executa o *script* que realiza a estimação ARX e retorna a lista de valores estimados, ao efetuar uma requisição do tipo POST (cria), passando como parâmetros o sensor e o tempo de horas para prever.

***/rnn*:** Executa o *script* que realiza a estimação *RNN* e retorna a lista de valores estimados, ao efetuar uma requisição do tipo POST (cria), passando como parâmetros o sensor e o tempo de horas para prever.

4.5 Camada de decisão

Para visualização dos dados, foi desenvolvida uma página *WEB* do tipo *Single Page Application* (SPA), utilizando a *framework* de *Javascript VueJS*, por possuir bibliotecas com componentes prontos, como botões, cartões e outros elementos que compõem uma página.

Nela é possível cadastrar novos motores à planta industrial, adicionar sensores aos motores, configurar a periodicidade de cada uma das leituras e visualizar as leituras.

Além disso, pela página é onde se faz as requisições para a API processar os dados das leituras passando pelos estimadores e visualizar os gráficos com a previsão das leituras futuras e uma previsão de em quantas horas o motor tem chance de falhar baseando-se na tabela de severidade de vibração do Anexo A. Esses limites de tempo de previsão e severidade são configuráveis. Imagens da plataforma em funcionamento podem ser visualizadas no Apêndice C.

5 RESULTADOS

Neste capítulo serão expostos os resultados obtidos com os estimadores escolhidos. Para efeitos de validação e comparação com um banco de dados de controle, os testes e resultados experimentais obtidos foram realizados com a base de dados fornecida por Martin *et al.* (2018), como foi descrita no capítulo 4, seguindo para a análise de operação do sistema, com dados reais coletados pelo *data logger*.

Fotos do *hardware* construído para este trabalho podem ser vistas no Apêndice B e as telas projetadas para a *dashboard* podem ser vistas no Apêndice C.

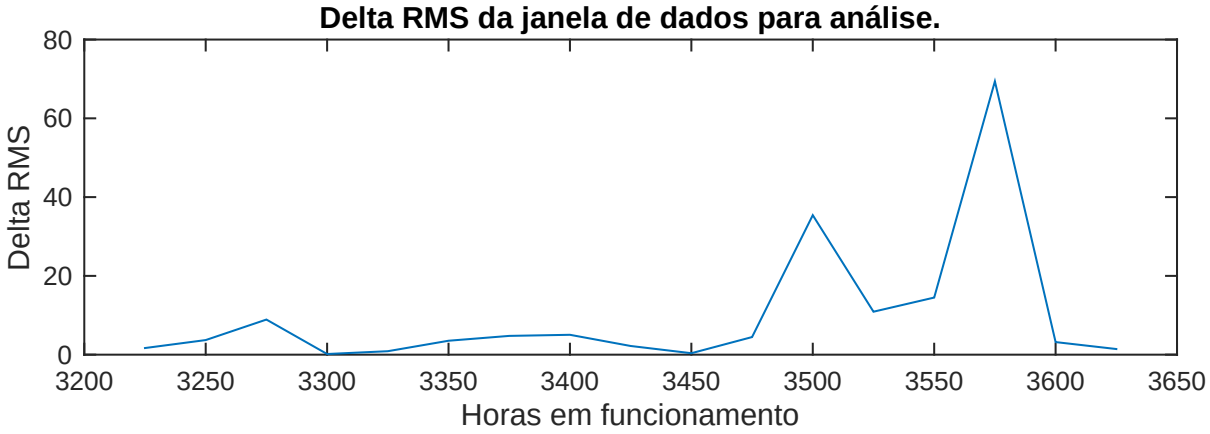
5.1 Análise do Banco de Dados

Apesar do banco de dados possuir uma grande quantidade de observações, foi selecionada uma janela com 18 observações, sendo uma à cada 25 horas, referente ao mês em que houve a primeira falha, sendo as 12 primeiras para treinamento dos estimadores e as 6 últimas para validação, que é onde a falha acontece.

Os dados foram normalizados de acordo com a sensibilidade do acelerômetro utilizado para captação de dados, mais informações podem ser obtidas no site do projeto GOTIX (MARTIN *et al.*, 2018).

Na Figura 11 a seguir, está representado o gráfico do *delta* RMS na janela de dados selecionada. Pode-se notar que a partir das 3500 horas de funcionamento a variação do *delta* RMS é maior, que foi onde ocorreu a primeira falha pois este parâmetro é sensível a mudanças bruscas.

Figura 11 – Gráfico *delta* RMS



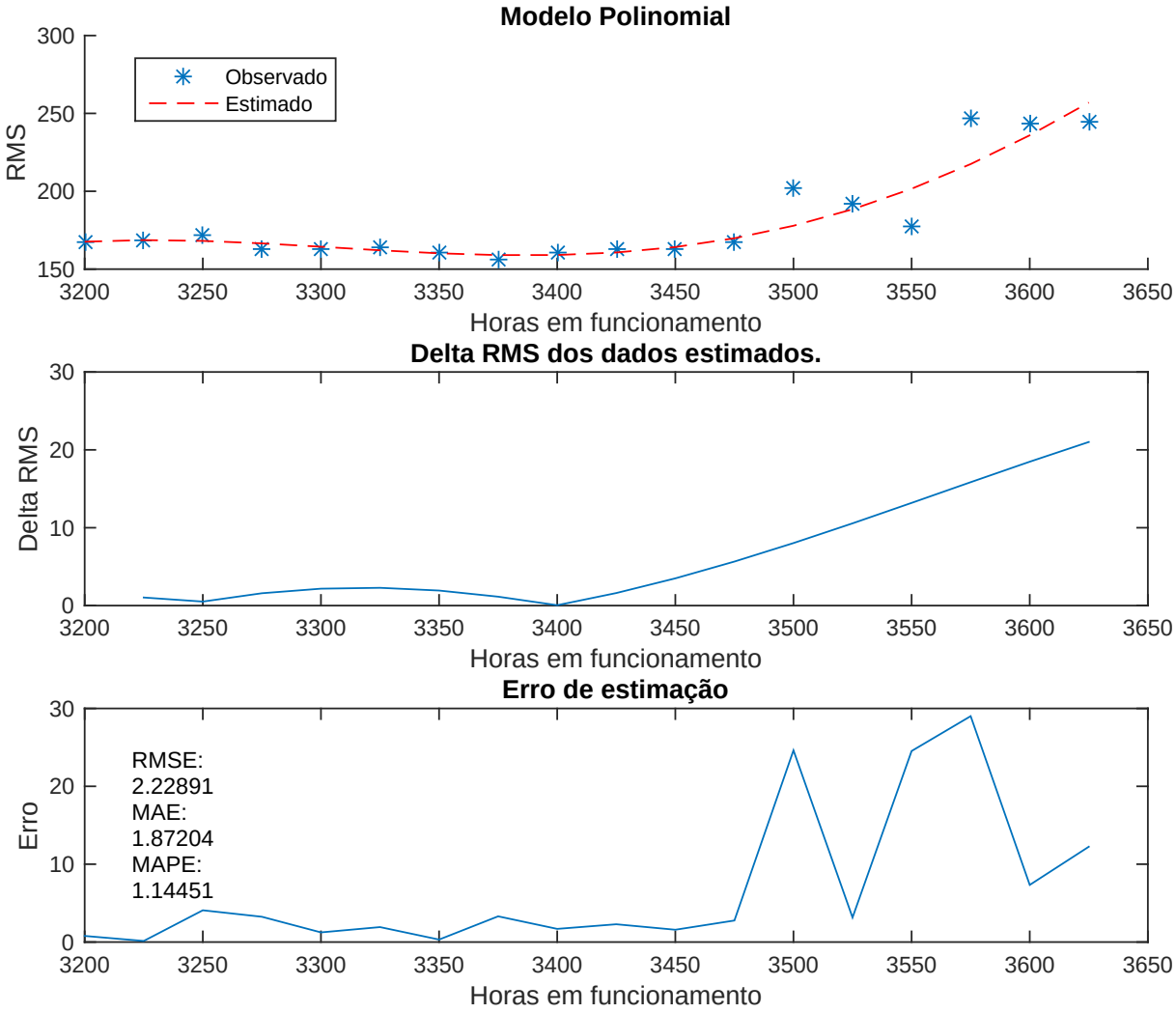
Fonte: Produzido pelo autor.

Este pode ser entendido e adicionado à plataforma como uma análise imediata, já que ele não é um estimador, com as mudanças bruscas nota-se a necessidade de uma manutenção no equipamento.

5.2 Modelo Polinomial

O modelo polinomial utilizado tanto na simulação, quanto na plataforma, foi o descrito na Equação 4.2. Para estimar os coeficientes do modelo, foi utilizado o método dos mínimos quadrados descrito no Capítulo 2. O resultado da estimação com este modelo pode ser visto na Figura 12, que esta representado na linha tracejada como a série estimada, e os pontos são os valores observados, seguindo pelo gráfico de *delta* RMS e erro de estimação com os parâmetros de avaliação do estimador.

Figura 12 – Estimação polinomial



Fonte: Produzido pelo autor.

Na estimação foram usadas as leituras de 3200 à 3425 para treinar, e de 3500 à 3625 para previsão. Pode-se notar que o modelo polinomial atua como um interpolador nos dados observados, assimilando a forma do sinal. Para calcular os parâmetros de erro para avaliação do estimador foram utilizados os dados de treinamento.

Os coeficientes estimados a_i (ver Equação 4.2) podem ser observados na Equação 5.1 a seguir.

$$a_1 = -7.9 \cdot 10^{-4}; \quad a_2 = 6.2; \quad a_3 = -11622.75; \quad a_4 = -207.4; \tag{5.1}$$

Dos dados de acurácia apresentados na Tabela 3 a seguir, pode-se inferir que, apesar ter um erro de 1.145%, possui uma variação de cerca de 2 m/s^2 do valor RMS esperado para o estimado, a diferença entre o RMSE e MAE mostra que há variabilidade entre os valores, ou seja, a diferença entre o valor observado e o valor predito é relevante. Mas isso se dá ao fato de que o modelo polinomial não captura as informações históricas dos dados, apesar de se adaptar a tendência crescente do sinal, indicando que possivelmente ocorrerão falhas em um futuro próximo.

Tabela 3 – Acurácia do modelo polinomial.

Erro	**Valor**
RMSE	2.229 m/s^2
MAE	1.872 m/s^2
MAPE	1.145%

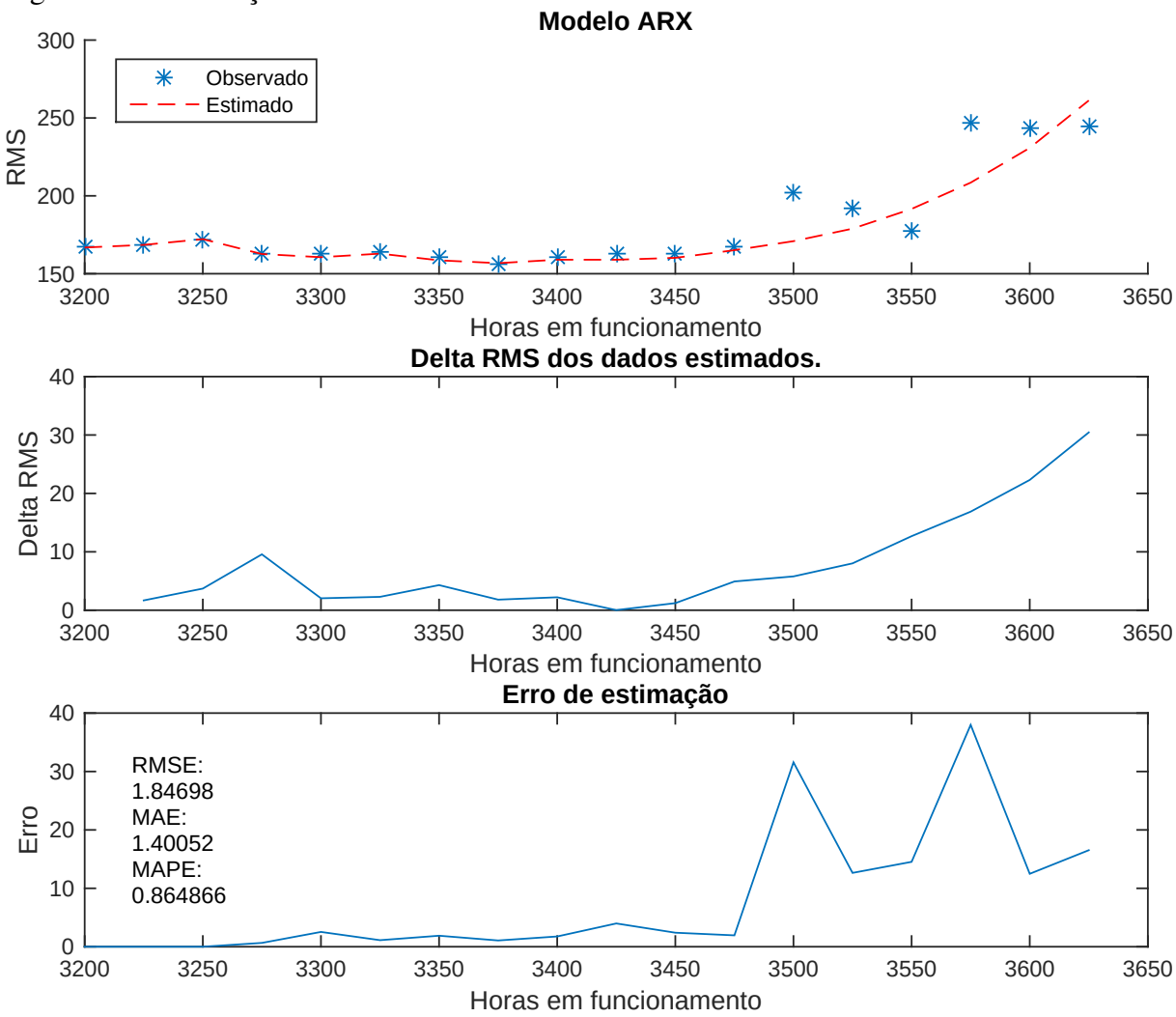
Fonte: Produzido pelo autor.

5.3 Modelo ARX

Assim como na seção anterior, a janela de dados é a mesma e o método estimador de parâmetros utilizado foi o mínimos quadrados. Na Figura 13 a seguir está o gráfico do resultado da previsão utilizando o modelo ARX.

Diferentemente do modelo polinomial, o ARX assimilou melhor os dados de observação, absorvendo suas características temporais. O estimador utilizando o modelo ARX obteve um desempenho melhor, como pode ser observado na Tabela 4, com uma menor variabilidade dos dados. O RMSE é sensível a grandes mudanças, o que indica que a diferença entre o valor observado e o valor previsto para a janela de testes foi menor do que utilizando o modelo polinomial.

Figura 13 – Estimação ARX



Fonte: Produzido pelo autor.

Além de possuir uma taxa de erro MAPE de 0.8%, esse modelo segue a tendência crescente do sinal, mostrando que possivelmente ocorrerá uma falha.

Tabela 4 – Acurácia do modelo ARX.

Erro	Valor
RMSE	1.85 m/s^2
MAE	1.4 m/s^2
MAPE	0.86%

Fonte: Produzido pelo autor.

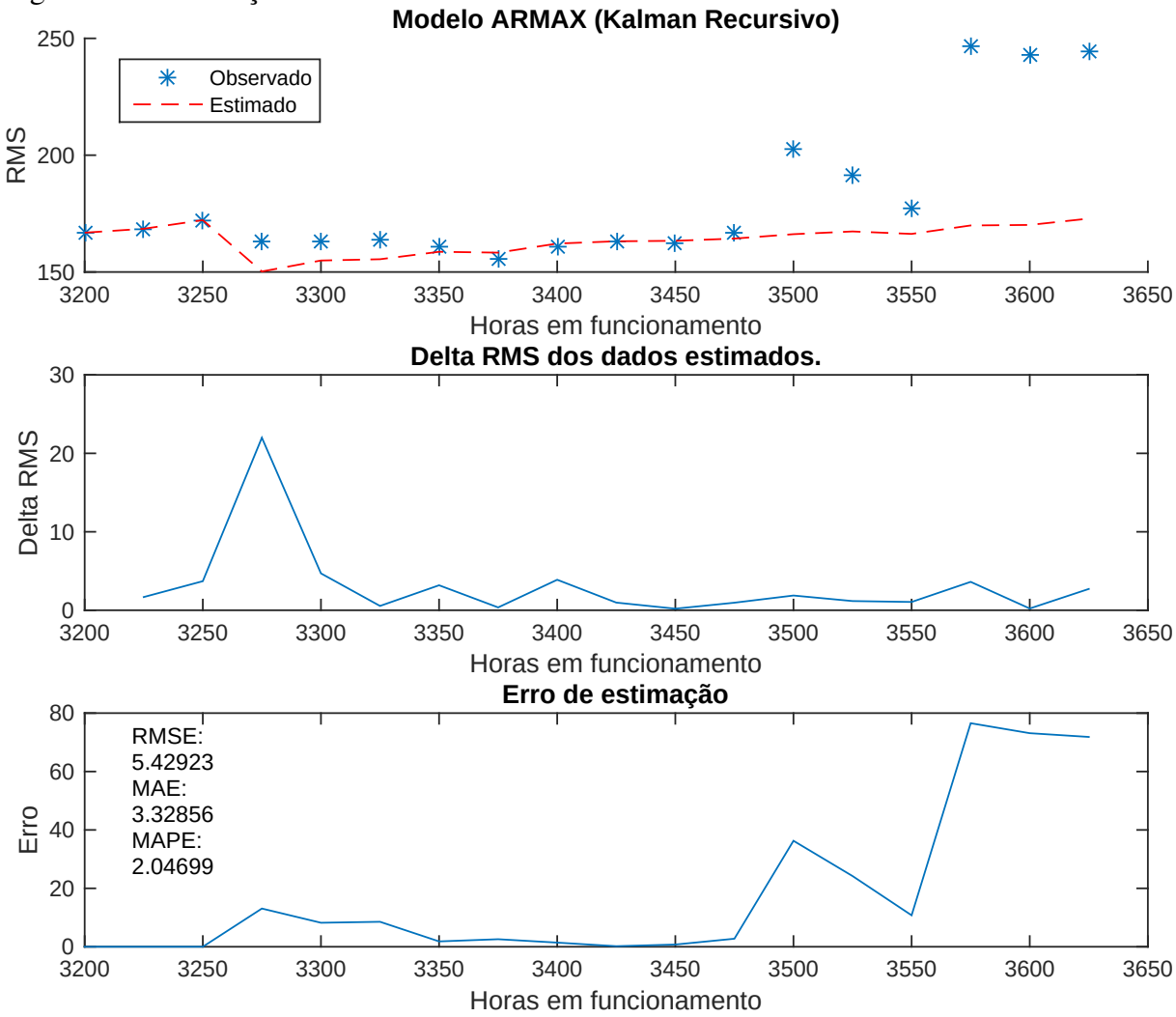
O vetor de parâmetros $\boldsymbol{\theta}$ (ver Equação 2.1) do modelo ARX estimado pelo método dos mínimos quadrados pode ser visualizado na Equação 5.2 a seguir.

$$\boldsymbol{\theta} = [-0.6663 \quad -0.1752 \quad -0.8133 \quad -16.4131 \quad 16.254 \quad 0.1890 \quad 0.0619] \tag{5.2}$$

5.4 Modelo ARMAX com filtro de Kalman

Na Figura 14 a seguir, está representado a estimação utilizando o modelo ARMAX com o estimador recursivo baseado no filtro de Kalman. Pode-se perceber que, ao comparar com os resultados anteriores, este modelo obteve um desempenho bem inferior, tanto para assimilar as características da curva quanto para prever a falha. A variação do delta RMS no final não é suficiente para indicar a falha no espectro previsto, apesar da inconsistência ela diminui a medida que avança.

Figura 14 – Estimação ARMAX



Fonte: Produzido pelo autor.

O vetor de parâmetros $\boldsymbol{\theta}$ do modelo ARMAX estimado pelo método recursivo baseado no filtro de Kalman pode ser visualizado na Equação 5.3 a seguir.

$$\boldsymbol{\theta} = [0.0219 \quad -0.0110 \quad 0.0592 \quad -0.1503 \quad -0.0409 \quad 0.0685 \quad 0.1779 \quad -1.2065] \tag{5.3}$$

Da Tabela 5 a seguir, pode-se inferir que, a variabilidade entre os valores observados e os valores reais é alta, gerando um RMSE de 5.4 m/s^2. A taxa de erro MAPE de 2% mostra que o modelo é capaz de acertar uma parte dos valores, porém pela curva na Figura 14 é possível perceber que o modelo não assimila a tendência crescente do sinal, ou seja, não consegue identificar a falhar que ocorre na janela de testes.

Tabela 5 – Acurácia do modelo ARMAX.

Erro	**Valor**
RMSE	5.43 m/s^2
MAE	3.33 m/s^2
MAPE	2%

Fonte: Produzido pelo autor.

5.5 Previsão com Rede Neural Recorrente

Para criação da *RNN*, foi utilizada uma biblioteca feita em *Python*, que possui suporte à *Matlab*, chamada *Pyrenn*. O modelo utilizado, determinado empiricamente, pois não há método analítico para modelar uma *RNN*, esta possui apenas uma entrada, cinco camadas internas, cada uma com cinco neurônios, e apenas uma saída. O resultado da previsão após um dos treinamentos realizados está representado na Figura 15 a seguir.

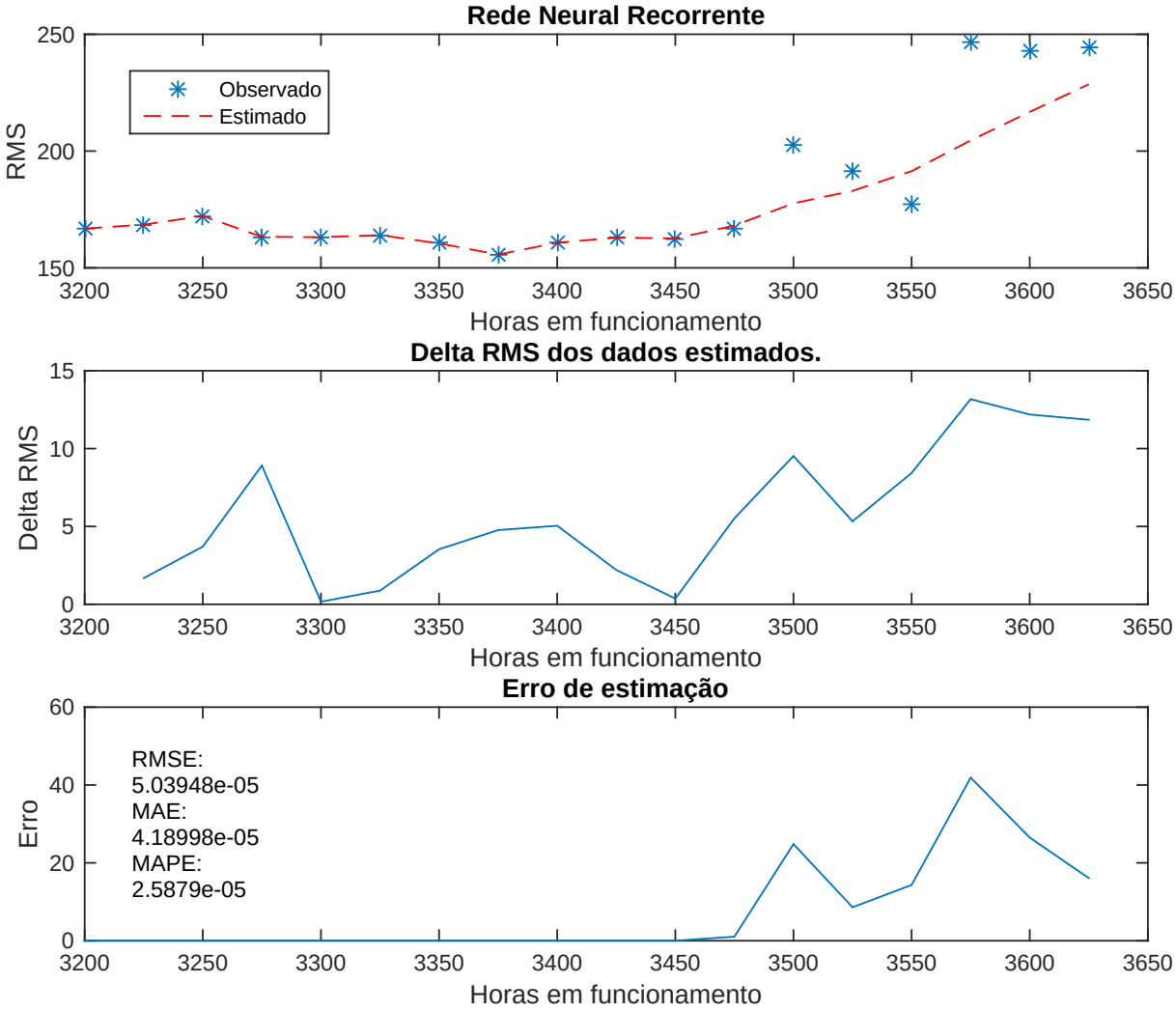
Como pode ser visto na Tabela de acurácia 6, dos métodos utilizados para estimação, a *RNN* foi a que melhor assimilou os dados de treino, tendo também sempre conseguido captar a tendência crescente dos gráficos de RMS. Os parâmetros de erro RMSE e MAE sempre quase zero, que significa uma boa precisão ao prever os valores de treinamento.

Tabela 6 – Acurácia da *RNN*.

Erro	**Valor**
RMSE	$5 \cdot 10^{-5}$ m/s^2
MAE	$4.2 \cdot 10^{-5}$ m/s^2
MAPE	$2.6 \cdot 10^{-5}$%

Fonte: Produzido pelo autor.

Figura 15 – Estimação *RNN*



Fonte: Produzido pelo autor.

A maior desvantagem de se utilizar uma rede neural é a inconsistência dos pesos entre cada treinamento, não há como garantir que sempre que treinar terá um resultado bom. Porém, como é possível salvar o modelo da rede e reutilizá-la, podem ser aplicados algoritmos para escolher o melhor modelo e salvar no banco de dados, como amostras de Monte Carlo ou algoritmos genéticos.

Além disso, como a rede neural recorrente possui uma memória, é possível retreinar o melhor modelo à medida que novos dados de observação são adicionados, podendo essa atualização ser feita de forma automática, pois o dispositivo não necessita da interferência humana para enviar os dados à plataforma, e o servidor pode executar esse comando ao adicionar uma nova leitura.

Diferente dos outros dois estimadores, que sempre irão retornar o mesmo vetor de parâmetros pois são métodos determinísticos, a rede neural sempre se adapta de forma diferente,

podendo pegar características intrínsecas diferentes a cada treinamento.

5.6 Plataforma IoT para Manutenção Preditiva

Como foram utilizados os mesmos métodos, o banco de dados utilizado na simulação teve o mesmo resultado na plataforma, e a rede neural teve resultado similar, por ser utilizada a mesma biblioteca.

Devido às limitações de memória do NodeMCU, o dispositivo é capaz de capturar pouco mais de 1 segundo de dados de vibração, à uma frequência de 800 Hz, e sensibilidade mínima 2G, limitando a gama de motores em que pode ser utilizado pela velocidade de rotação devido a taxa de amostragem.

Como a plataforma foi projetada de forma modular, ou seja, é uma plataforma escalável que não se limita só ao *hardware* que foi desenvolvido, sensores mais caros e precisos podem ser adicionados ao sistema fornecendo uma gama maior de motores.

Os parâmetros de severidade de vibração podem ser ajustados de acordo com a padronização utilizada pelo usuário. Possibilitando uma previsão mais justa com cada usuário.

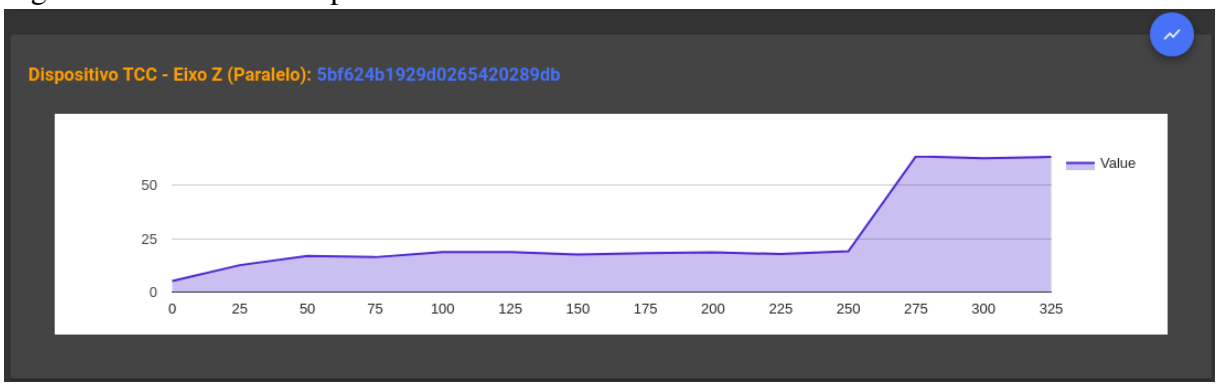
Porém, como o propósito desse trabalho é fornecer uma opção a baixo custo, o *hardware* utilizado foi suficiente para os testes com a plataforma. Para os testes o dispositivo foi acoplado ao eixo de um motor de ventilador com rotação de 60Hz obedecendo a condição da taxa de amostragem Nyquist, em que não há perda de informação.

O teorema da amostragem de Nyquist diz que para não haver perda de informações por amostragem, em que o sinal se sobrepõe ou está muito espaçado, a frequência de amostragem deve ser pelo menos duas vezes maior que a máxima frequência do sinal (OPPENHEIM, 1999).

Com o dispositivo acoplado ao motor de testes, foi utilizado um método similar ao de Khabarov (2015) para simular um aumento na vibração do maquinário, já que não foram realizados testes prolongados como os feitos pelo GIPSA-lab (MARTIN *et al.*, 2018). Primeiro foram capturados uma série de dados com utilização normal do equipamento, para aumentar o nível de vibração da máquina anexou-se uma fita na ponta de uma das hélices e foram capturados mais três observações.

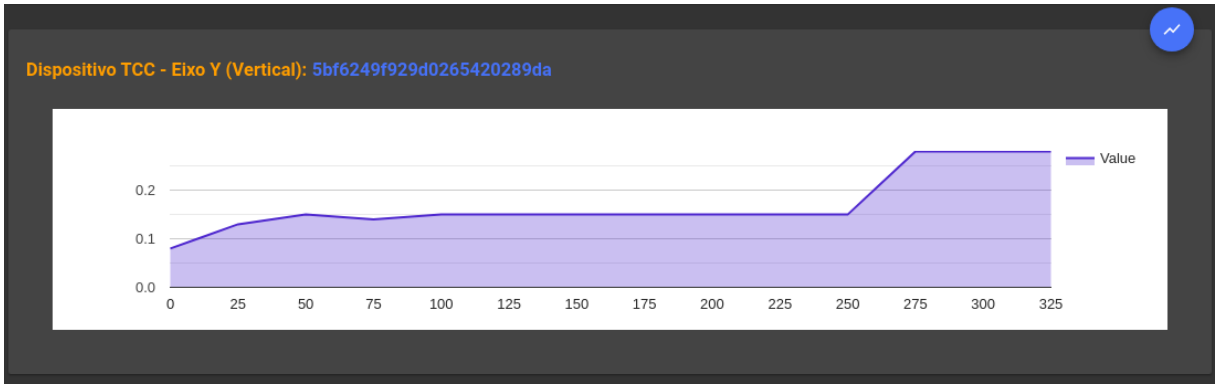
Assim, nas leituras observa-se um aumento do valor RMS, como pode ser visto nas Figuras 16, 17 e 18 a seguir.

Figura 16 – Leitura eixo paralelo ao eixo do motor



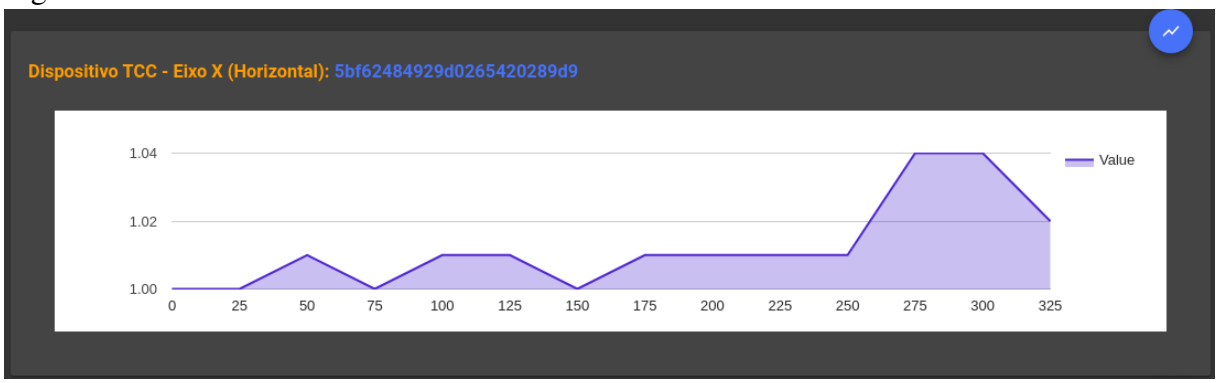
Fonte: Produzido pelo autor.

Figura 17 – Leitura eixo vertical referente ao motor



Fonte: Produzido pelo autor.

Figura 18 – Leitura eixo horizontal referente ao motor



Fonte: Produzido pelo autor.

Assim, com o pico de vibração no final, espera-se que o sistema entenda que a curva RMS possua tendência a aumentar e gerar uma falha. Para verificar isso, foram escolhidas as leituras do eixo que possuiu maior deslocamento, o eixo paralelo ao eixo no motor.

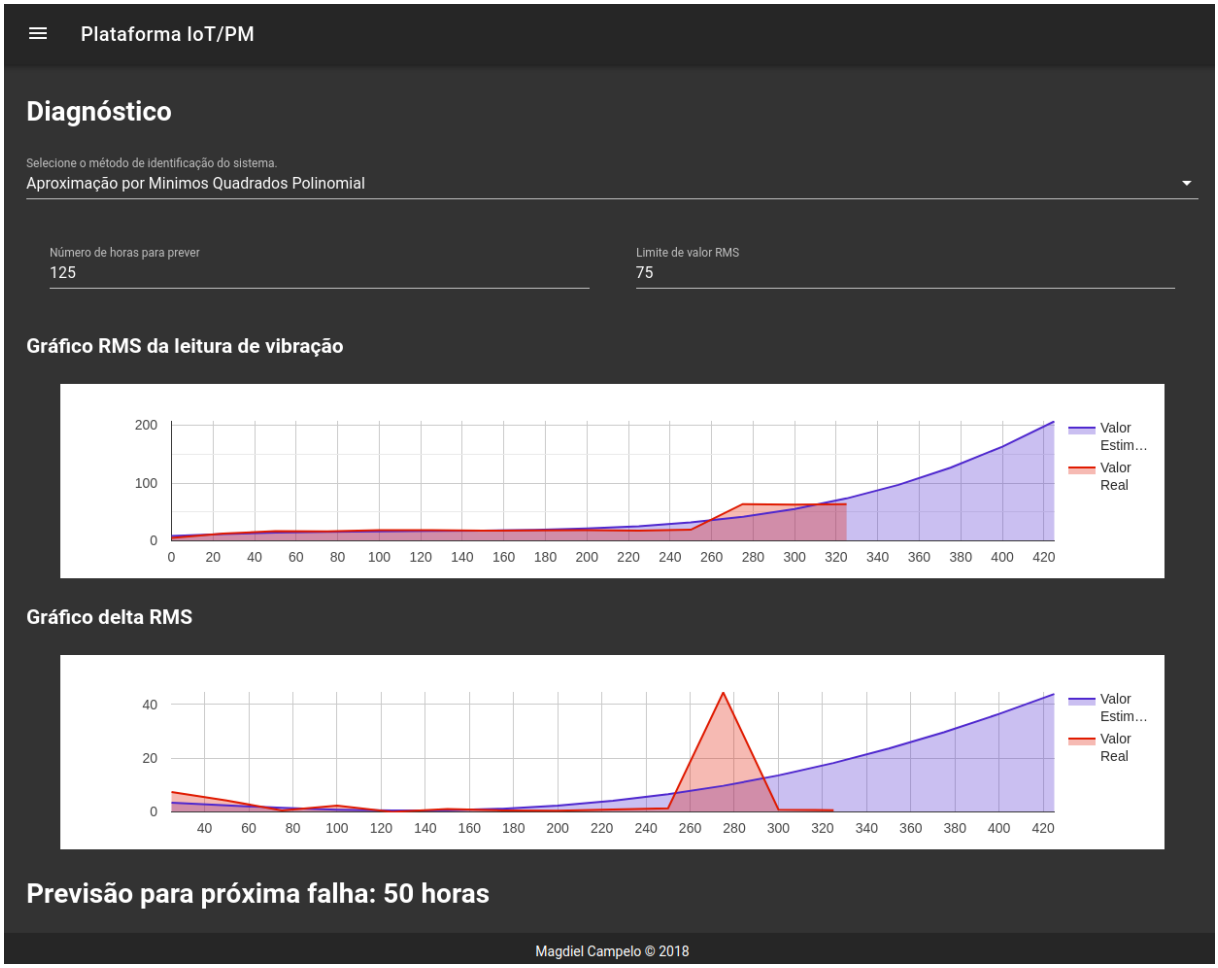
Para as análises foram determinadas a previsão das próximas 125 horas, espaçadas de 25 horas cada. O limiar de severidade de vibração foi definido em 75, caso ultrapasse esse valor o sistema detecta uma possível falha.

Pode ser observado na Figura 19 que o estimador polinomial, tenta interpolar o

gráfico de observações, e segue a tendência crescente. Já na Figura 19, em que está representado o estimador utilizando o modelo ARX, é possível observar que neste caso a curva segue uma tendência linear, mostrando-se não muito eficaz, demonstrando a importância da comparação entre os métodos. E por fim, na Figura 21 está representada a *RNN*, que obteve o melhor resultado de todos os métodos, assimilando bem as características dos valores observados e prevendo valores futuros dentro da tendência da curva de RMS.

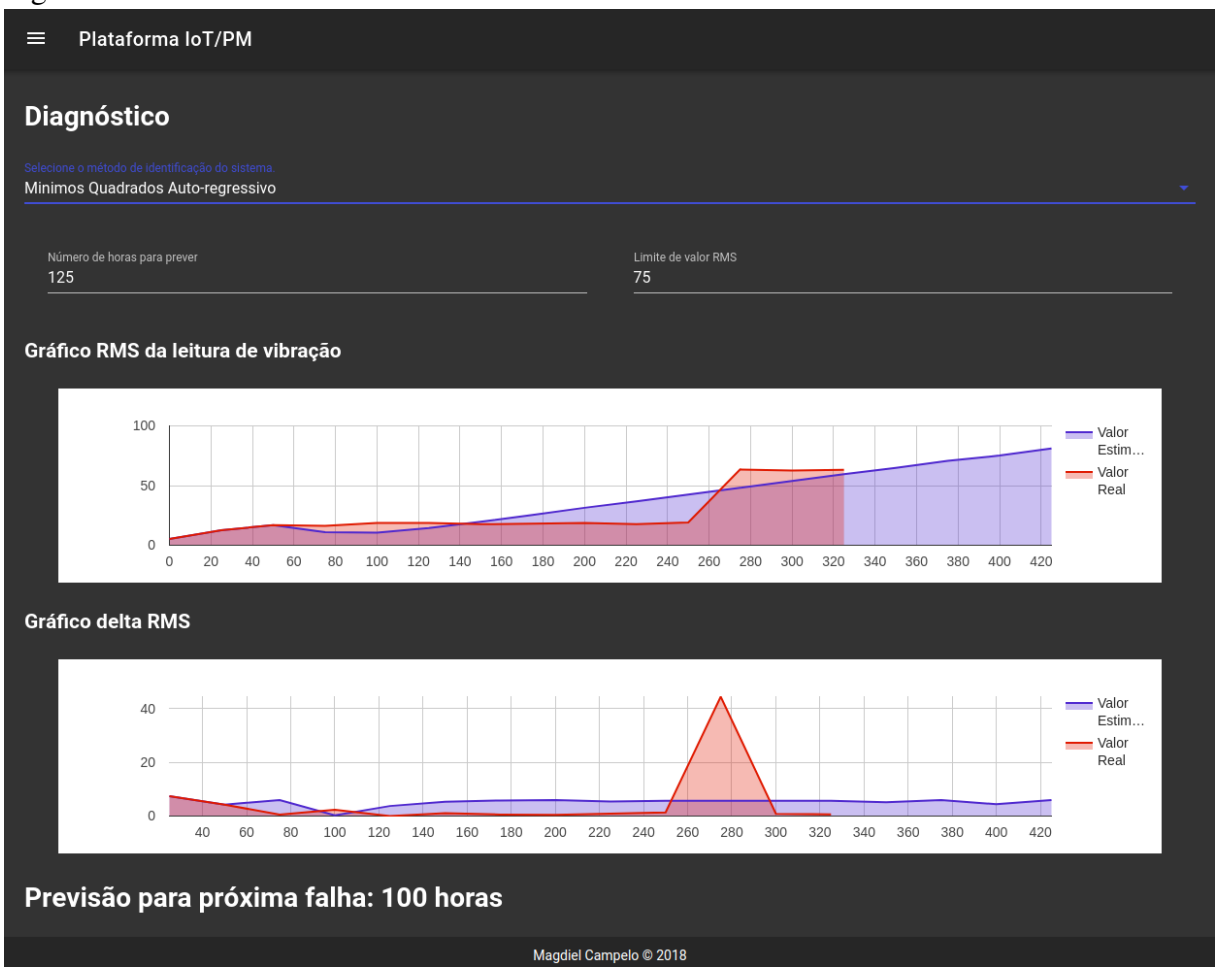
O estimador ARMAX não foi implementado na plataforma pois não foi capaz de assimilar a tendência crescente do sinal, ou seja, não é útil para prever uma possível falha.

Figura 19 – Previsão da curva de RMS usando o modelo polinomial



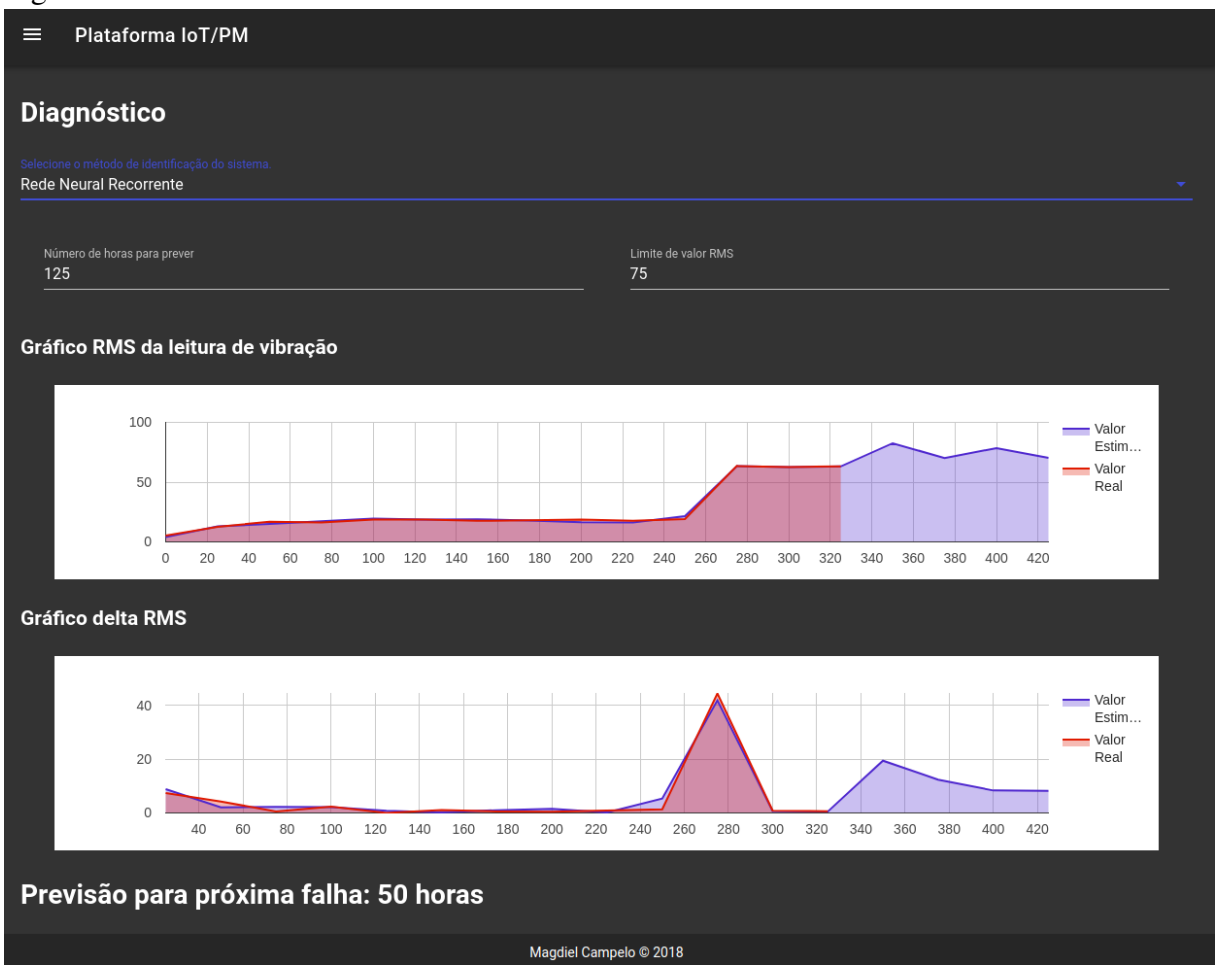
Fonte: Produzido pelo autor.

Figura 20 – Previsão da curva de RMS usando o modelo ARX



Fonte: Produzido pelo autor.

Figura 21 – Previsão da curva de RMS usando a *RNN*



Fonte: Produzido pelo autor.

O desempenho do algoritmo nos outros eixos pode ser observado na Tabela 7 a seguir. Dependendo do eixo a amplitude de variação é muito baixa, o que significa que os parâmetros de erro na mesma unidade que o RMS terão valores baixos, sendo o parâmetro MAPE o mais intuitivo para comparação.

Mesmo com baixa variação, ou em casos praticamente invariáveis (como o eixo horizontal), o estimador utilizando *RNN* foi o que obteve melhor desempenho, tanto em estimar os dados de treino como prever a tendência do sinal de vibração.

Tabela 7 – Comparação de acurácia entre os eixos de teste.

Estimador	**RMSE**	**MAE**	**MAPE**
Eixo paralelo ao motor			
Polinomial	7.9 m/s^2	5.3 m/s^2	21.2 %
ARX	10.6 m/s^2	7.9 m/s^2	35.3 %
RNN	0.7 m/s^2	0.5 m/s^2	4.7 %
Eixo vertical ao motor			
Polinomial	0.02 m/s^2	0.01 m/s^2	9.8 %
ARX	0.04 m/s^2	0.03 m/s^2	19.2 %
RNN	0.01 m/s^2	0.008 m/s^2	4.7 %
Eixo horizontal ao motor			
Polinomial	0.008 m/s^2	0.006 m/s^2	0.6 %
ARX	0.008 m/s^2	0.006 m/s^2	0.6 %
RNN	$7 \cdot 10^{-8}$ m/s^2	$4 \cdot 10^{-8}$ m/s^2	$4.7 \cdot 10^{-6}$%

Fonte: Produzido pelo autor.

5.7 Discussões

A partir dos resultados obtidos, ao comparar o banco de dados utilizado na simulação e as leituras feitas pelo dispositivo construído, é possível concluir que os métodos possuem desempenho diferente dependendo da forma da curva de RMS, quando há mais variações, mais não-linearidade, os modelos polinomial e ARX tem bom desempenho, porém quando a curva é mais estável eles tentam seguir a tendência mas adicionando as variações ao valores estimados, o ARX no geral estima melhor que o polinomial por ter características temporais nos parâmetros.

Apesar disso, a rede neural recorrente acaba sempre tendo um melhor desempenho para inferir a tendência do sinal e prever resultados mais próximos a realidade, pois ela se adapta à curva e e leva em consideração entradas e saídas passadas.

6 CONCLUSÕES E TRABALHOS FUTUROS

No presente Trabalho de Conclusão de Curso foi apresentada uma plataforma *IoT* de baixo custo para manutenção preditiva, utilizando tecnologias emergentes e de fácil acesso, além de demonstrar um dispositivo de baixo custo para captura e pré-processamento de dados úteis para o diagnóstico de equipamentos rotores e manutenção preditiva, oferecendo um intervalo de tempo em que o equipamento pode ocorrer uma falha, seguindo padrões internacionais de severidade de vibração. Mostrou também o desenvolvimento de uma biblioteca para processamento das leituras capturadas pelo dispositivo construído, ou qualquer outro que seja cadastrado no sistema, e uma *dashboard* para visualização dos dados e controle dos sensores. Sendo o trabalho validado utilizando um banco de dados com observações reais de um equipamento de funcionamento contínuo.

O trabalho também valida a taxonomia para um sistema de predição de falhas baseado em *IoT* proposta por Xu *et al.* (2012), pois foi desenvolvido seguindo o padrão proposto.

6.1 Trabalhos futuros

Para trabalhos futuros algumas melhorias podem ser aplicadas, como adicionar um sistema de autenticação para criar níveis de segurança ao sistema, para melhorar o desempenho e as predições podem ser aplicados algoritmos para selecionar o melhor estimador, como algoritmos genéticos, amostras de Monte Carlo e o pré-processamento automático desses modelos, ou até mesmo aplicar lógica *fuzzy* na previsão para determinar o nível de severidade futura baseando-se em padrões internacionais.

REFERÊNCIAS

AGUIRRE, L. A. **Introdução à Identificação de Sistemas – Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais**. 3. ed. Belo Horizonte: Editora UFMG, 2007. ISBN 9788570415844. Disponível em: <https://books.google.com.br/books?id=f9IwE7Ph0fYC>.

ASSOCIAçãO BRASILEIRA DE NORMAS TéCNICAS - ABNT. **NBR 5462: confiabilidade e mantenabilidade**. [S.l.], 1994. Disponível em: <https://www.abntcatalogo.com.br/norma.aspx?ID=004086>.

AZIMA DLI. **Severity Chart**. AZIMA, 2018. Disponível em: <http://www.azimadli.com>. Acesso em: 30 Nov. 2018.

CAESARENDRA, W.; TJAHJOWIDODO, T. A review of feature extraction methods in vibration-based condition monitoring and its application for degradation trend estimation of low-speed slew bearing. **Machines**, Multidisciplinary Digital Publishing Institute, v. 5, n. 4, p. 21, 2017.

CAMELO, H. d. N.; LUCIO, P. S.; JUNIOR, J. B. V. L.; CARVALHO, P. C. M. de. Time series forecasting methods and hybrid modeling both applied on monthly average wind speed for regions of northeastern brazil. **Revista Brasileira de Meteorologia**, SciELO Brasil, v. 32, n. 4, p. 565–574, 2017. Disponível em: <http://www.scielo.br/pdf/rbmet/v32n4/0102-7786-rbmet-32-04-0565.pdf>.

CIVERCHIA, F.; BOCCHINO, S.; SALVADORI, C.; ROSSI, E.; MAGGIANI, L.; PETRACCA, M. Industrial internet of things monitoring solution for advanced predictive maintenance applications. **Journal of Industrial Information Integration**, v. 7, p. 4 – 12, 2017. ISSN 2452-414X. Enterprise modelling and system integration for smart manufacturing. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2452414X16300954>.

COCHRAN, W. G. **Sampling Techniques: 3d Ed**. [S.l.]: Wiley New York, 1977.

COMBET, F.; MARTIN, N.; JAUSSAUD, P.; LÉONARD, F. Detection of a gear coupling misalignment in a gear testing device. In: **Eleventh International Congress on Sound and Vibration, ICSV11**. [S.l.: s.n.], 2004.

DONG, L.; MINGYUE, R.; GUOYING, M. Application of internet of things technology on predictive maintenance system of coal equipment. **Procedia Engineering**, v. 174, p. 885 – 889, 2017. ISSN 1877-7058. 13th Global Congress on Manufacturing and Management Zhengzhou, China 28-30 November, 2016. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877705817302370>.

EHLERS, R. Análise de séries temporais. **Departamento de Estatística, Universidade Federal do Paraná**, 2005. Disponível em: <http://www.each.usp.br/rvicente/AnaliseDeSeriesTemporais.pdf>. Acesso em: 30 Nov. 2018.

ELECTRODRAGON. **ESP-12F ESP8266 Wifi Board**. 2017. Publicado Online. Disponível em: <http://www.electrodragon.com/w/ESP-12F_ESP8266_Wifi_Board>. Acesso em: 30 Nov. 2018.

FONTE, C. **Ajustamento de observações utilizando o método dos mínimos quadrados**. Coimbra, 1994. Disponível em: <http://www.mat.uc.pt/~cfonte/docencia/Topografia\%20Aplicada/MMQnovo.pdf>. Acesso em: 30 Nov. 2018.

FULCHER, J. Computational intelligence: An introduction. In: ____. **Computational Intelligence: A Compendium**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 3–78. ISBN 978-3-540-78293-3. Disponível em: <https://doi.org/10.1007/978-3-540-78293-3_1>.

GOLDSCHMIDT, R. R. **Uma Introdução à Inteligência Computacional: fundamentos, ferramentas e aplicações**. 1. ed. Rio de Janeiro: IST - Rio, 2010. Disponível em: <http://www.boente.eti.br/boente2012/fuzzy/ebook/ebook-fuzzy-goldschmidt.pdf>.

GONZÁLEZ, R. C. D. **Desenvolvimento de um protótipo analisador de vibração de baixo custo para uso em manutenção preditiva**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, 2014. Disponível em: <https://repositorio.ufsc.br/xmlui/handle/123456789/129200>.

GROKHOTKOV, I. **Arduino core for ESP8266 WiFi chip**. 2018. Publicado Online. Disponível em: <https://github.com/esp8266/Arduino>. Acesso em: 30 Nov. 2018.

HAYKIN, S. **Redes Neurais: Princípios e Prática**. Artmed, 2007. ISBN 9788577800865. Disponível em: <https://books.google.com.br/books?id=bhMwDwAAQBAJ>.

HUNG, M. (Ed.). **Leading the IoT - Gartner Insights on How to Lead in a Connected World**. Gartner, 2017. Disponível em: <https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf>.

IGBA, J.; ALEMZADEH, K.; DURUGBO, C.; EIRIKSSON, E. T. Analysing rms and peak values of vibration signals for condition monitoring of wind turbine gearboxes. **Renewable Energy**, v. 91, p. 90 – 106, 2016. ISSN 0960-1481. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0960148116300064>.

ISO. **International Organization for Standardization**. 2018. Disponível em: <https://www.iso.org/>. Acesso em: 30 Nov. 2018.

KHABAROV, N. **Predictive Maintenance**. 2015. Disponível em: <http://orange.dataart.com/projects/predictive-maintenance/>. Acesso em: 30 Nov. 2018.

LASI, H.; FETTKE, P.; KEMPER, H.-G.; FELD, T.; HOFFMANN, M. Industry 4.0. **Business & Information Systems Engineering**, v. 6, n. 4, p. 239–242, Aug 2014. ISSN 1867-0202. Disponível em: <https://doi.org/10.1007/s12599-014-0334-4>.

MARÇAL, R. F. M. **Um método para detectar falhas incipientes em máquinas rotativas baseado em análise de vibração e lógica Fuzzy**. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 12 2000.

MARTIN, N.; GRANJON, P.; SERVIERE, C. **GOTIX Project**. 2018. Disponível em: <http://www.gipsa-lab.grenoble-inp.fr/projet/gotix/>. Acesso em: 30 Nov. 2018.

MARTINS, G. F. **Implemento e Análise do Comportamento do Algoritmo de Kalman como Estimador de Estados de um Servo Motor**. Monografia (Trabalho de Conclusão de Curso) — Universidade de Estadual de Londrina, Londrina, 11 2014. Disponível em: <http://www.uel.br/ctu/deel/TCC/TCC2014_GabrielFelipeMartins.pdf>.

MOBLEY, R. **An Introduction to Predictive Maintenance**. Elsevier Science, 2002. (Plant Engineering). ISBN 9780080478692. Disponível em: <https://books.google.com.br/books?id=SjqXzxpAzSQC>.

MONTGOMERY, D. C.; JENNINGS, C. L.; KULAHCI, M. **Introduction to time series analysis and forecasting**. [S.l.]: John Wiley & Sons, 2015.

MOURÃO, M. S.; JUNIOR, G. L. P. **Desenvolvimento de um sistema de detecção e diagnóstico de falha aplicado a um sistema de refrigeração**. Monografia (Trabalho de Conclusão de Curso) — Universidade de Brasília, Brasília, 6 2011. Disponível em: <http://bdm.unb.br/bitstream/10483/2902/1/2011_MarceloMourao_GeorgePalmeiraJunior.pdf>.

MUKHERJEE, A. **Getting started with ESP NodeMcu using ArduinoIDE**. 2016. Published Online. Disponível em: <https://www.hackster.io/Aritro/getting-started-with-esp-nodemcu-using-arduinoide-aa7267>.

OPPENHEIM, A. V. **Discrete-time signal processing**. [S.l.]: Pearson Education India, 1999.

OSÓRIO, F. S.; VIEIRA, R. Sistema híbridos inteligentes - tutorial. **ENIA'99 - ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL**, Rio de Janeiro, 1999.

RIBEIRO, D. A. **Análise de vibração em motores elétricos com mouse óptico**. Dissertação (Mestrado) — Universidade Federal de Lavras, Lavras, 2017. Disponível em: <http://repositorio.ufla.br/jspui/handle/1/12177>.

ROBOCORE. **Loja Virtual**. 2018. Publicado Online. Disponível em: <https://www.robocore.net/>. Acesso em: 30 Nov. 2018.

SCHEFFER, C.; GIRDHAR, P. **Practical Machinery Vibration Analysis and Predictive Maintenance**. Elsevier Science, 2004. (Practical Machinery Vibration Analysis and Predictive Maintenance). ISBN 9780080480220. Disponível em: <https://books.google.com.br/books?id=tAvTO1t2mwkC>.

SOARES, F. G. **Desenvolvimento de um data logger de vibração de baixo custo para uso em manutenção preditiva**. Monografia (Trabalho de Conclusão de Curso) — Universidade Federal do Pampa, Alegrete, 12 2015. Disponível em: <http://dspace.unipampa.edu.br:8080/jspui/handle/riu/1513>.

TAVARES, M. F. **Utilização dos modelos ARX e ARMAX em plantas industriais ruidosas**. Tese (Doutorado) — Universidade de São Paulo, 2012.

WEBER, T. S. Tolerância a falhas: conceitos e exemplos. **Intech Brasil**, v. 52, p. 32 – 42, 2003. Disponível em: <http://www.inf.ufrgs.br/~taisy/disciplinas/textos/>.

XU, X.; CHEN, T.; MINAMI, M. Intelligent fault prediction system based on internet of things. **Computers Mathematics with Applications**, v. 64, n. 5, p. 833 – 839, 2012. ISSN 0898-1221. Advanced Technologies in Computer, Consumer and Control. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0898122111011059>.

YUAN, M. **Conhecendo o mqtt**. 2018. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html>. Acesso em: 30 Nov. 2018.

ZHANG, D.; WAN, J.; HSU, C.-H. R.; RAYES, A. Industrial technologies and applications for the internet of things. **Computer Networks**, v. 101, p. 1 – 4, 2016. ISSN 1389-1286. Industrial Technologies and Applications for the Internet of Things. Disponível em: <http://www.sciencedirect.com/science/article/pii/S138912861630041X>.

APÊNDICE A – TECNOLOGIAS UTILIZADAS PARA DESENVOLVER A PLATAFORMA

Na tabela abaixo encontra-se um resumo das tecnologias e seus propósitos neste projeto.

Tabela 8 – Tabela de tecnologias utilizadas.

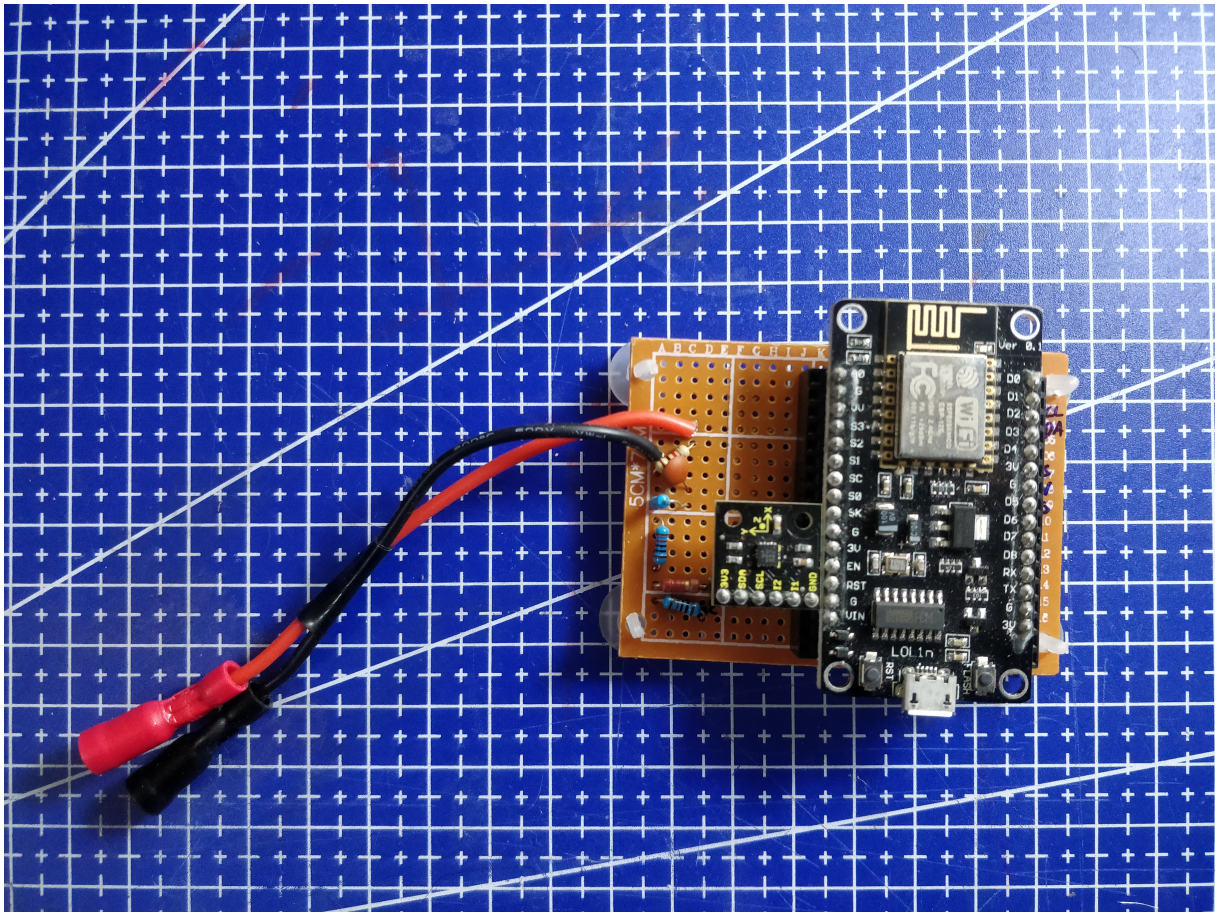
Tecnologia	**Propósito**
MATLAB R2015a	Simulação dos métodos a serem utilizados e pré-processamento do banco de dados GOTIX
Javascript	Principal linguagem utilizada no projeto
Arduino	IDE utilizada para programar o NodeMCU
NodeJS	Interpretador de Javascript para servidor
Express.js	Framework para desenvolver API RESTful
MongoDB e Mongoose	Banco de dados não relacional (NoSQL) utilizado para armazenar os dados da aplicação
Python	Utilizado para desenvolver os nós resposáveis por processar os dados
Numpy	Biblioteca para manipulações de matrizes e operações matemáticas complexas
Pyrenn	Biblioteca para criação e utilização de redes neurais recorrentes em python e matlab
MQTT.js	Biblioteca para a API inscrever e publicar no broker MqTT
Node-RED e Mosca	Utilizado para rodar o broker MqTT
VueJS, Vuetify e Vue-resource	Desenvolvimento da dashboard responsável por visualizar e cadastrar informações
Git e Github	Utilizado para controle de versionamento e publicação do código fonte

Fonte: Produzido pelo autor.

APÊNDICE B – *DATALOGGER* MQTT

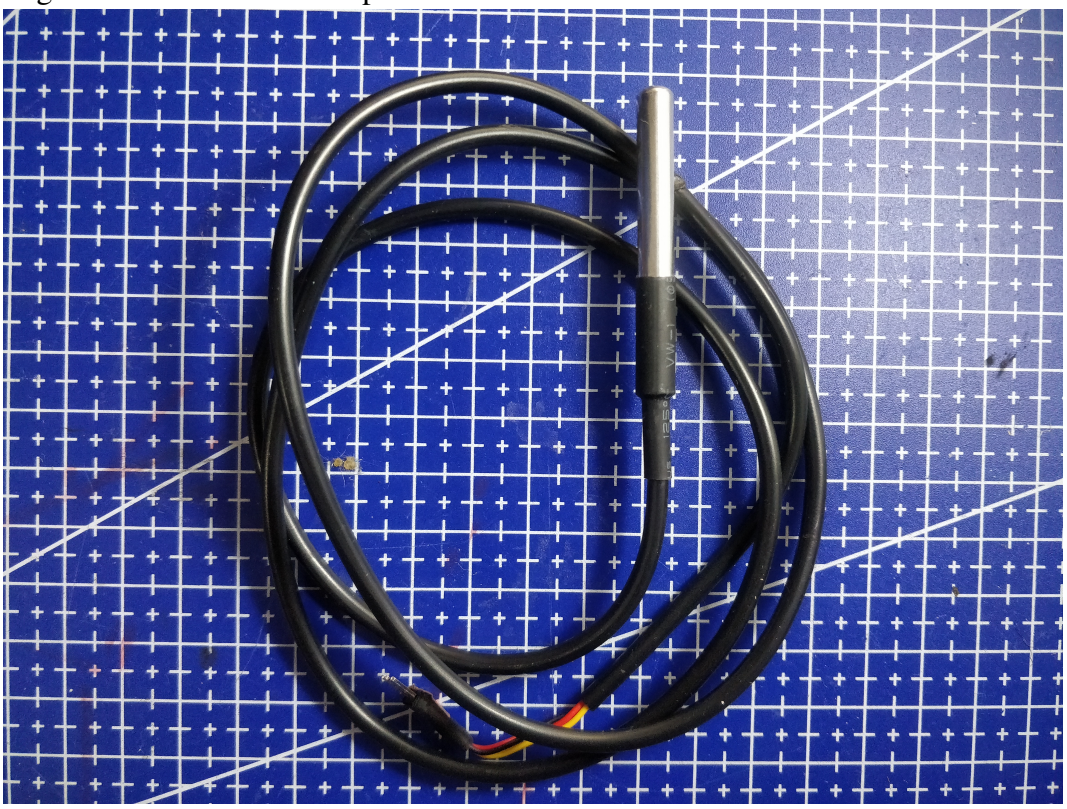
O hardware foi construído de forma modularizada, com o propósito de ser possível adicionar ou remover sensores, dependendo da necessidade.

Figura 22 – *NodeMCU* com acelerômetro MMA8452Q.



Fonte: Produzido pelo Autor.

Figura 23 – Sensor de Temperatura DS18B20.

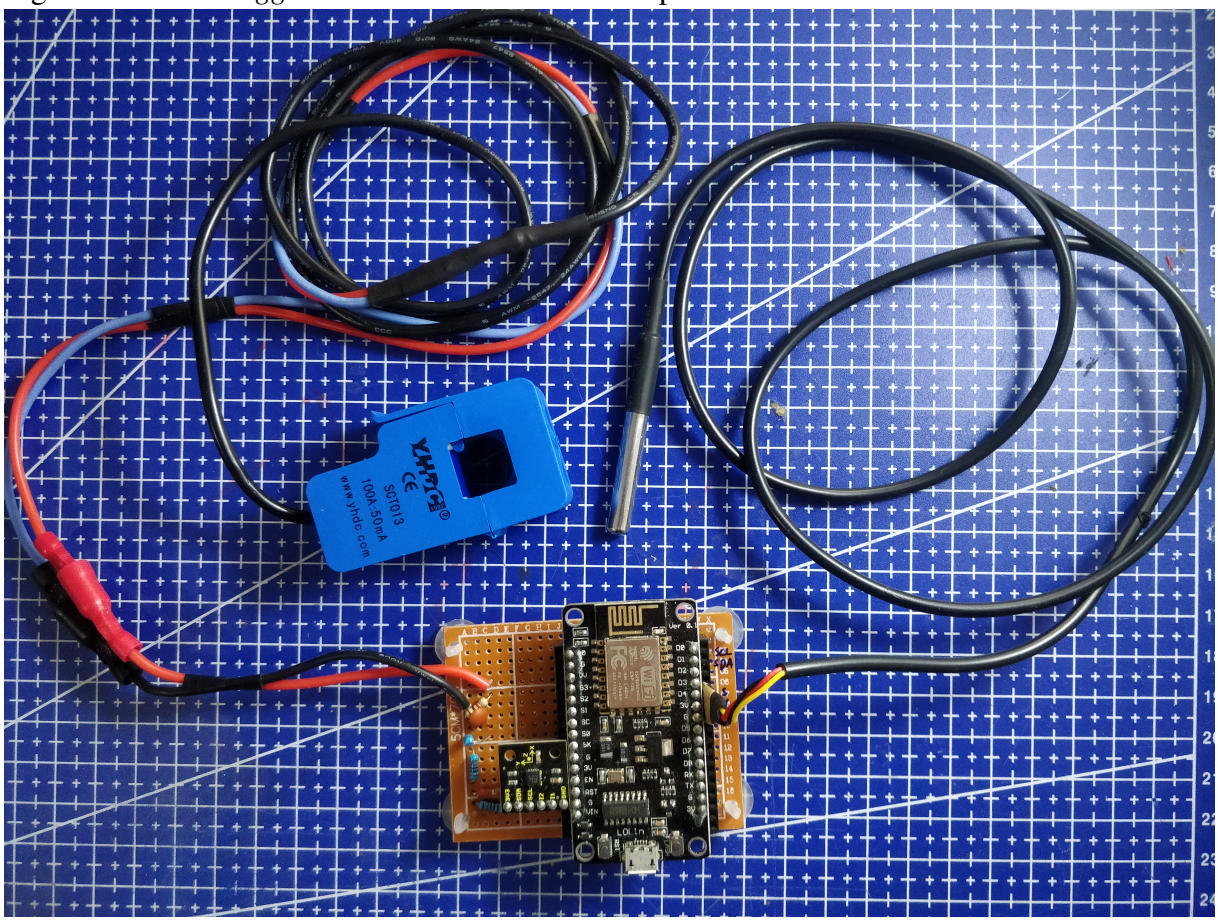


Fonte: Produzido pelo Autor.

Figura 24 – Sensor de Corrente SCT-013.

Fonte: Produzido pelo Autor.

Figura 25 – *Datalogger* com todos os sensores acoplados

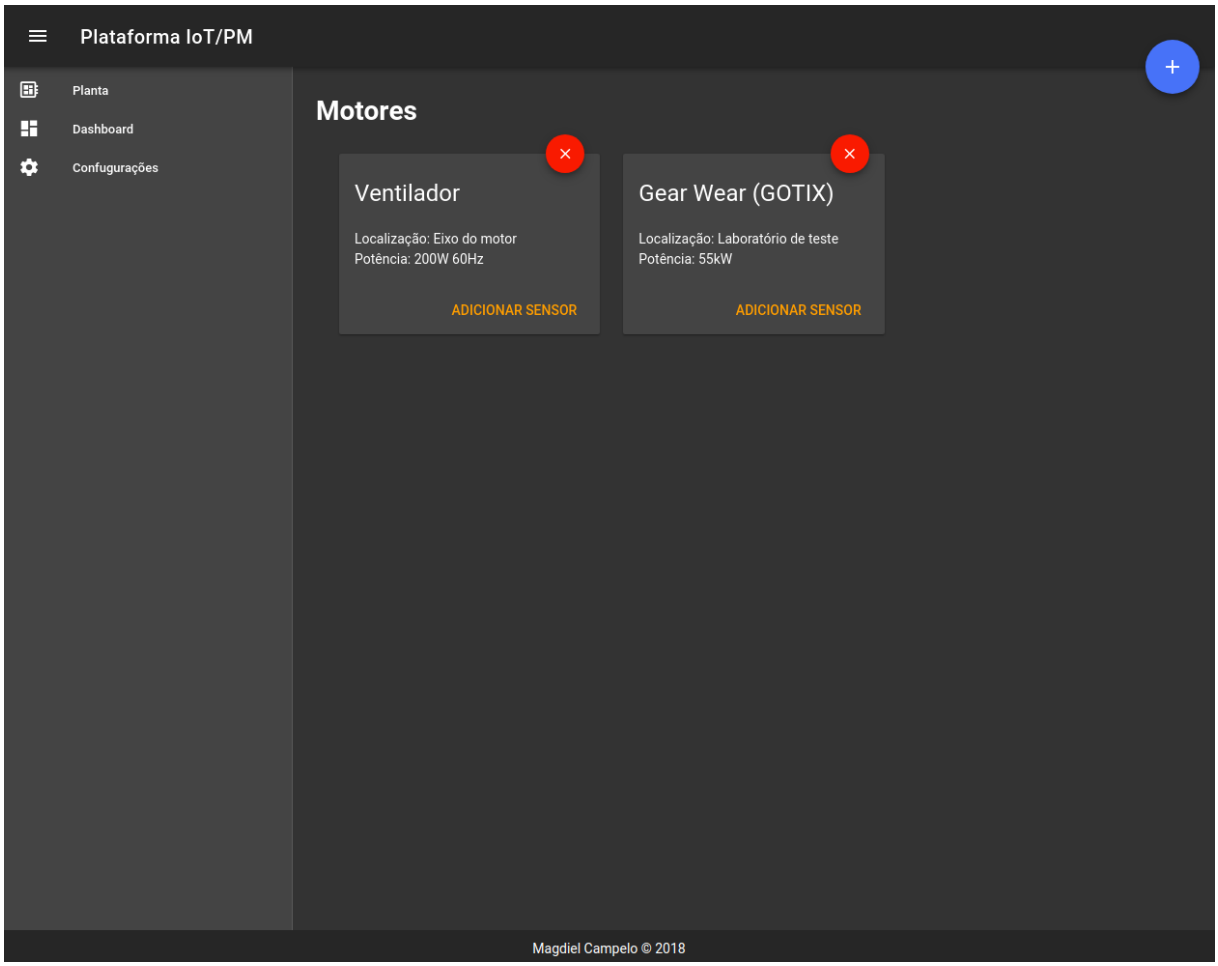


Fonte: Produzido pelo Autor.

APÊNDICE C – PLATAFORMA IOT PARA MANUTENÇÃO PREDITIVA

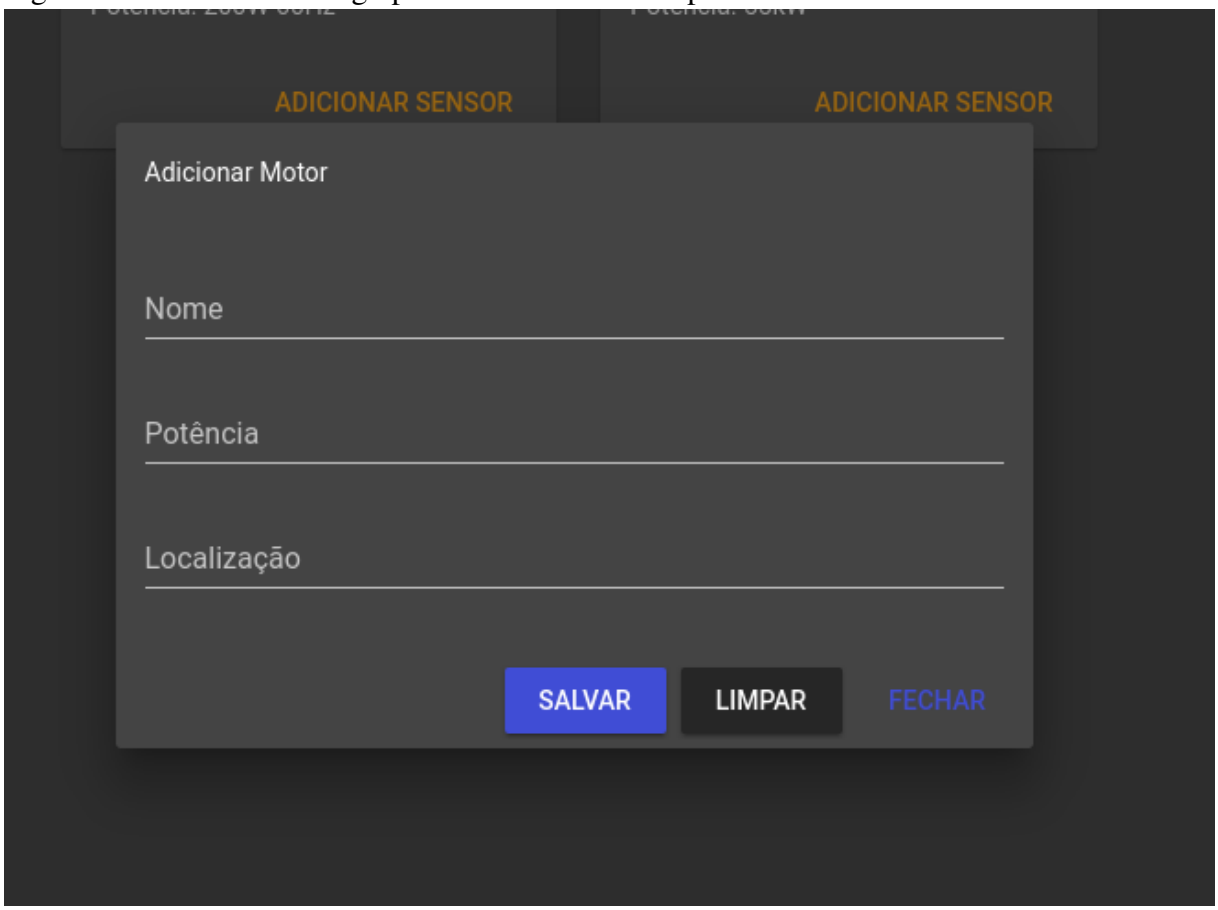
Seguem abaixo as telas desenvolvidas para a plataforma, em que é possível visualizar dados, cadastrar novos motores e sensores, configurar a periodicidade de captura de dados e realizar estimações e previsões de falha.

Figura 26 – Página inicial que contém a planta do sistema.



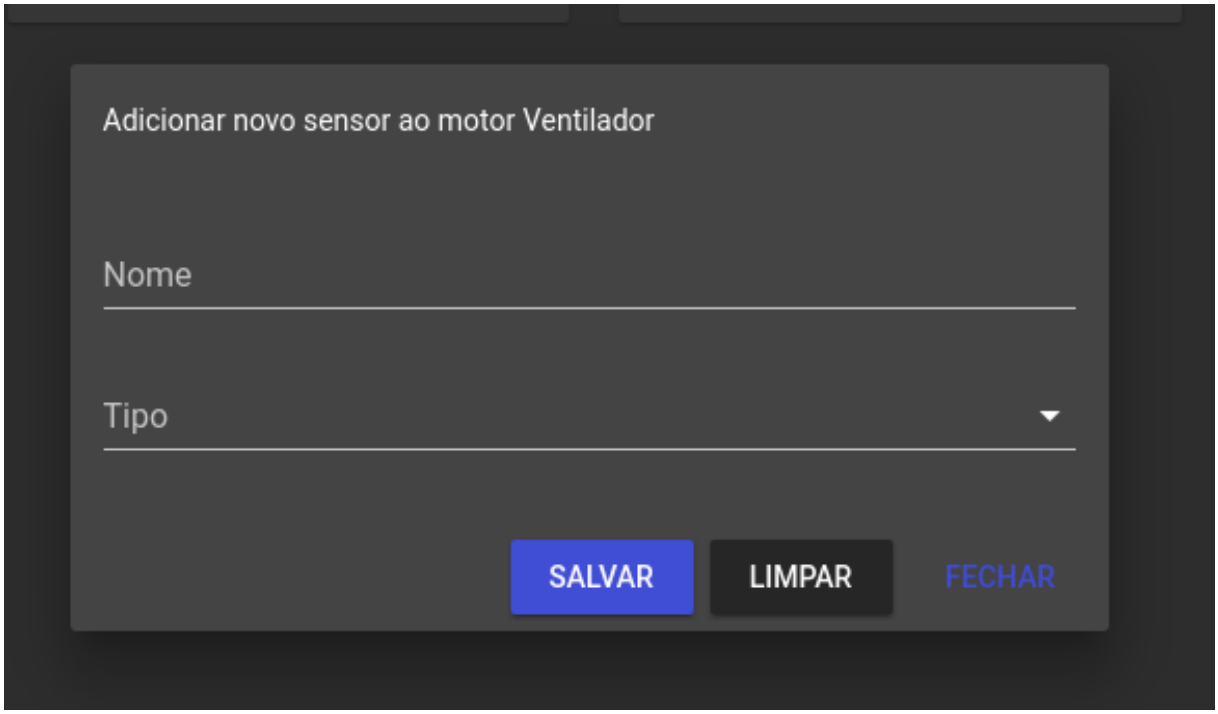
Fonte: Produzido pelo Autor.

Figura 27 – Caixa de dialogo para adicionar motor à planta.



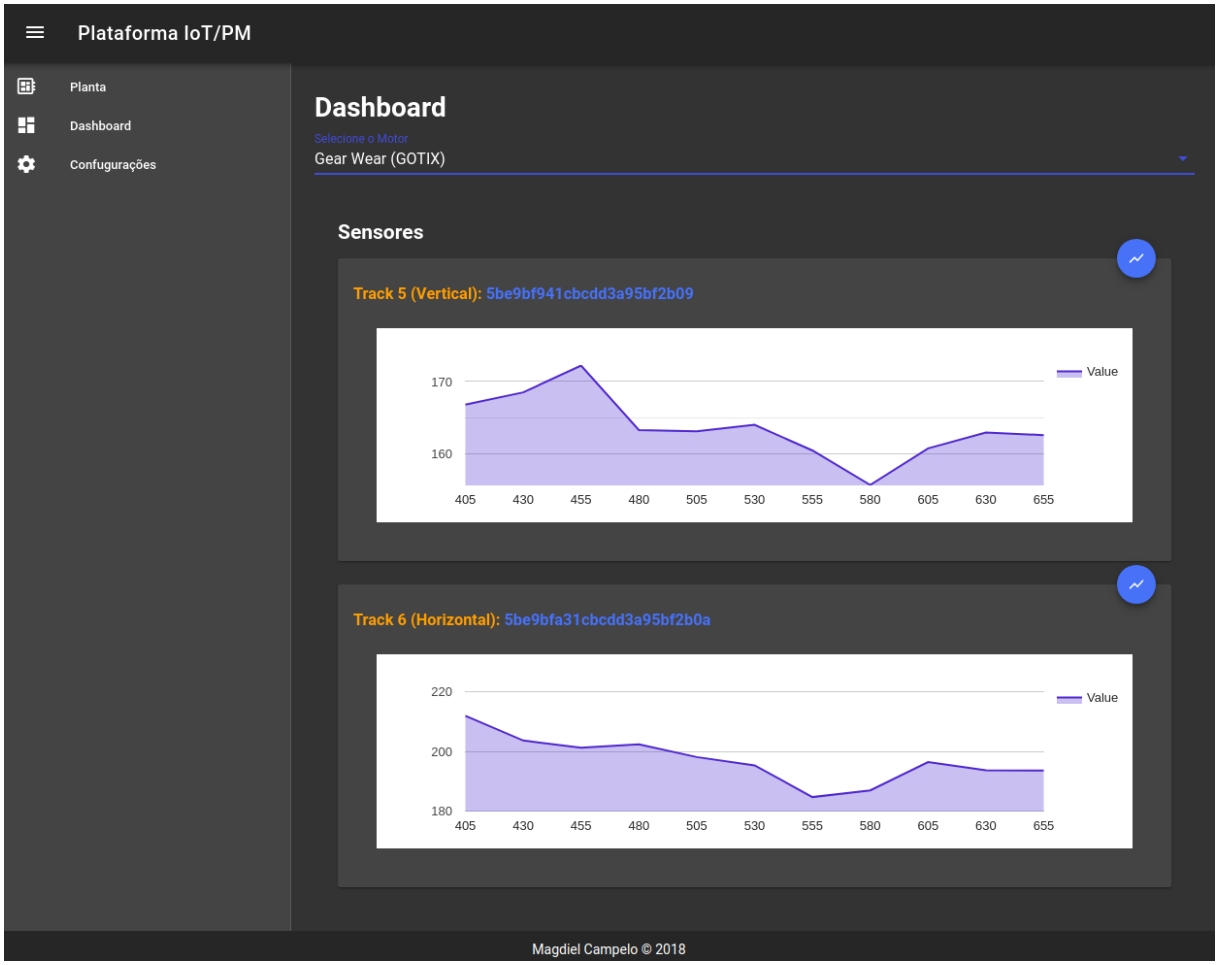
Fonte: Produzido pelo Autor.

Figura 28 – Caixa de dialogo para adicionar sensor à um motor.



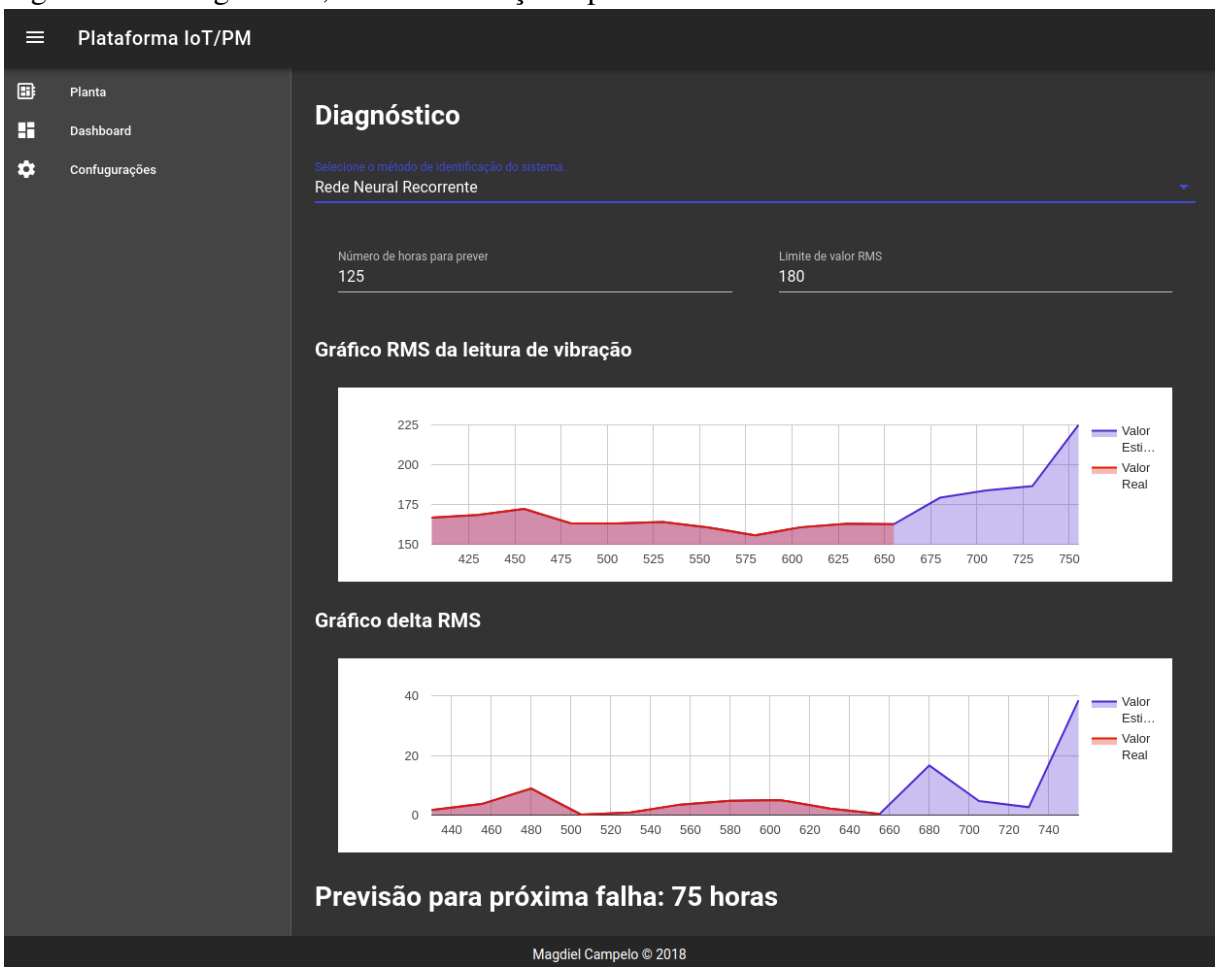
Fonte: Produzido pelo Autor.

Figura 29 – Dashboard, tela de visualização de dados.



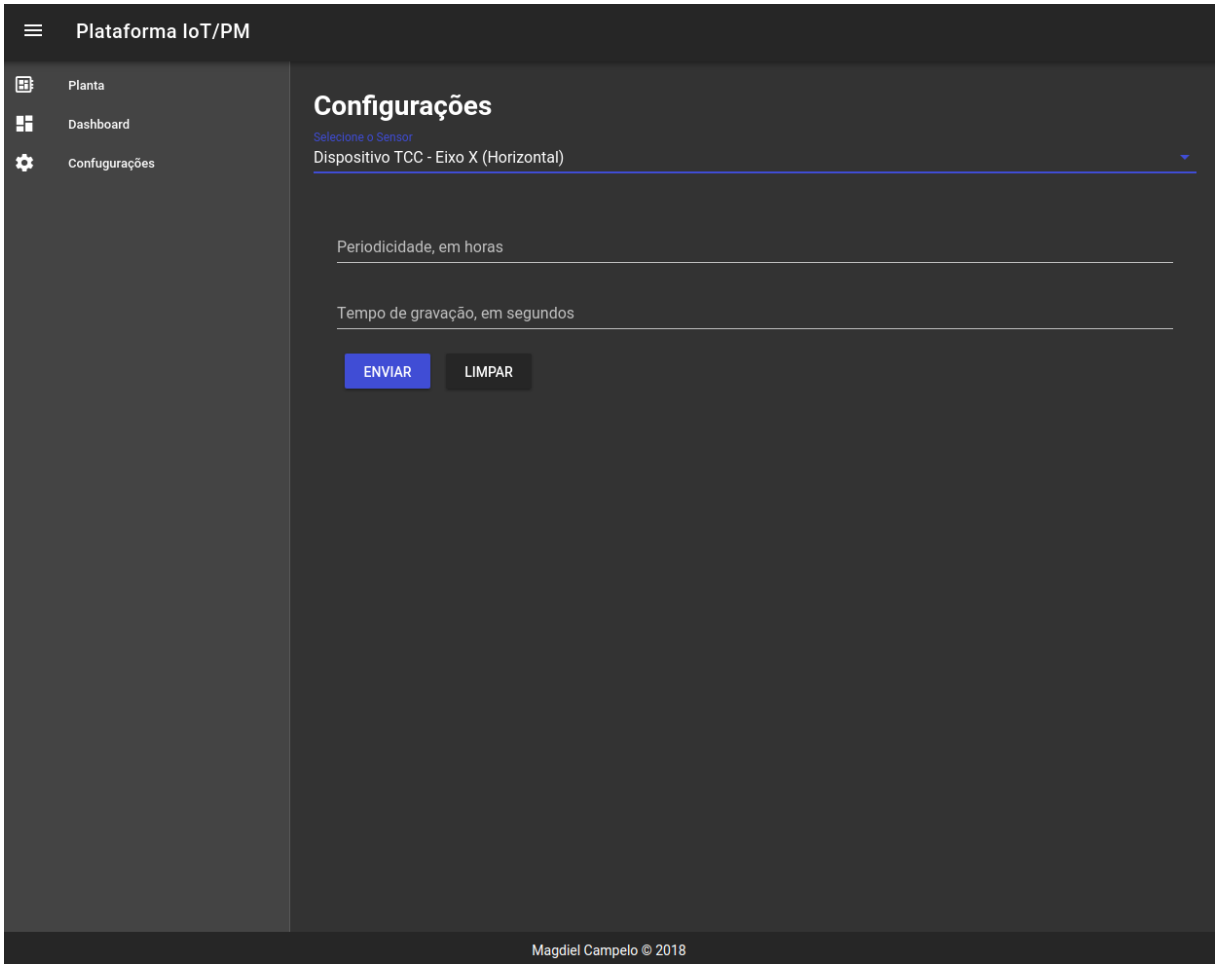
Fonte: Produzido pelo Autor.

Figura 30 – Diagnóstico, tela de estimação e previsão de falha.



Fonte: Produzido pelo Autor.

Figura 31 – Configurações, tela de configuração do dispositivo.



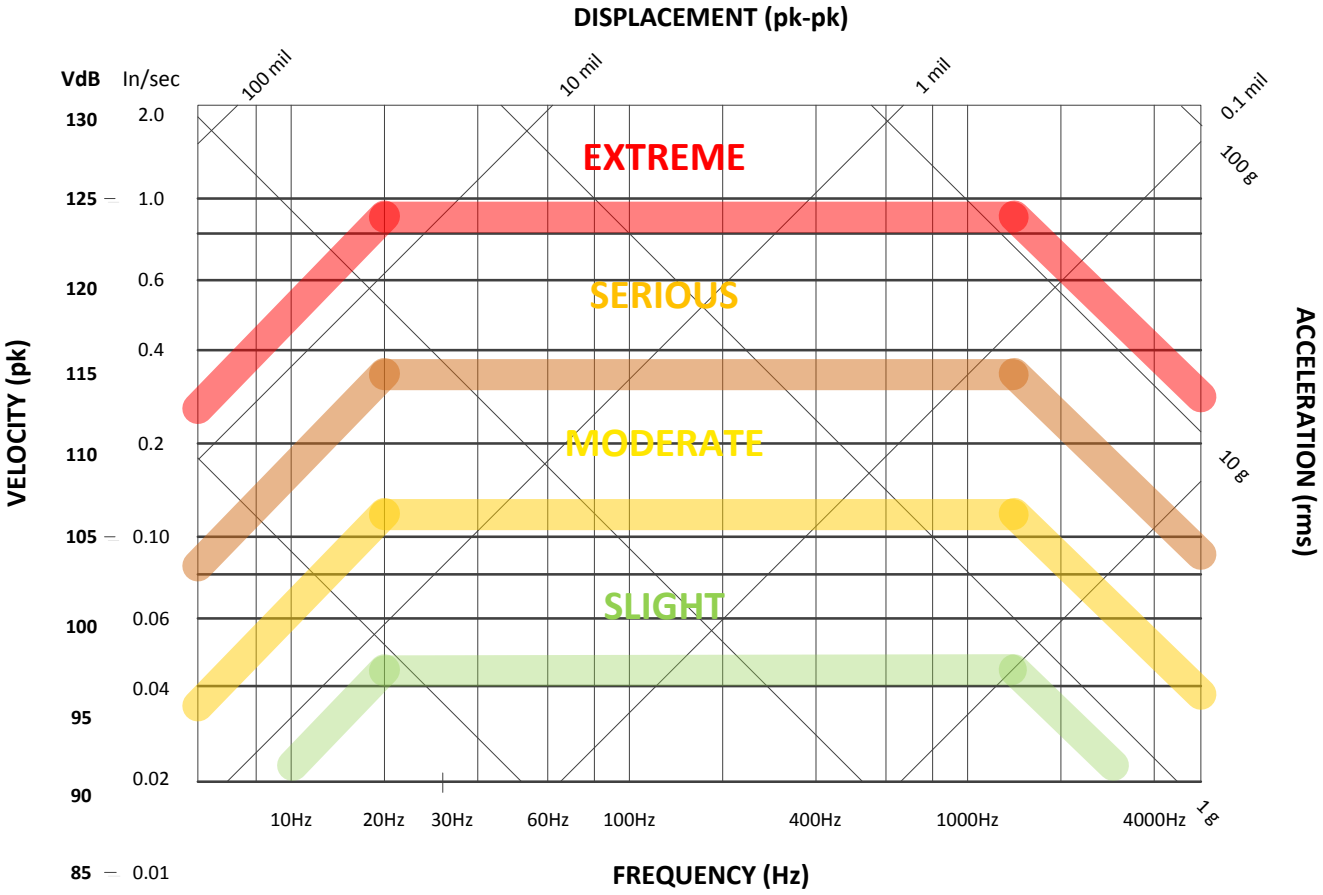
Fonte: Produzido pelo Autor.

ANEXO A – GUIA DE SEVERIDADE DA VIBRAÇÃO DE UMA MÁQUINA

Neste anexo pode ser visto um guia de severidade de vibração para máquinas rotacionais, disponibilizado pela empresa Azima DLI, que fornece serviços de manutenção preditiva.

MACHINERY VIBRATION SEVERITY GUIDE

For rotational rate and harmonics or broadband vibration measurements



Amplitude Adjustments: Reciprocating machinery, increase limits by 8 dB
Bearing tones, reduce limits by 12 dB

STANDARD ENGLISH UNITS

D	Displacement	mils (pk-pk) (0.001in)
V	Velocity	in/sec (pk)
A	Acceleration	g (rms) (386in/sec^2)
f	Frequency	Hz (cycles /sec)

Velocity Decibels

Ref: 0 VdB = 10^{-8} m/s (rms)

Ref: 0 VdB = 5.568 x 10^{-7} in/sec (pk)

Acceleration Decibels

Ref: 0 AdB = 10^{-5} m/sec^2 (rms)

RELATIONSHIPS & CONVERSIONS

D (pk-pk)	= $10^{(VdB\ -75)\ /\ 20}$ / f
VdB	= 20 $\log_{10}$ (V x 10^7 / 5.568)
V (pk)	= $10^{(VdB\ -\ 125)\ /\ 20}$
AdB	= 20 log (A x 10^4 / 3.861)
A (rms)	= $10^{(AdB\ -\ 68)\ /\ 20}$

D	= 318 (V / f)	= 27600 (A / f^2)
V	= 86.9 (A / f)	= f D / 318
A	= f^2 D / 27600	= f V / 86.9

dB RATIO

dB	RATIO
1	1.12
2	1.26
3	1.41
4	1.58
5	1.78
6	1.99
8	2.50
10	3.16
15	5.63
20	10.00
40	100.00

ADD dB differences, MULTIPLY Ratios

HEADQUARTERS
300 TradeCenter, Suite 4610
Woburn, MA 01801 USA
www.AzimaDLI.com