



UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
CENTRO DE CIÊNCIAS COMPUTACIONAIS
CURSO DE ENGENHARIA DE AUTOMAÇÃO

Projeto de Graduação em Engenharia de Automação

Rede Neural de Convolução baseada em Lenet-5 para diagnóstico de falha de rolamento

Marcos Vinicius Moreira Ramis

Projeto de Graduação apresentado ao Curso de Engenharia de Automação da Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtenção do grau de Engenheiro de Automação

Orientador: Prof. Dr. Vinicius Menezes de Oliveira

Rio Grande, 2021

RESUMO

RAMIS, Marcos Vinicius Moreira. **Rede Neural de Convolução baseada em Lenet-5 para diagnóstico de falha de rolamento**. 2021. 52 f. Projeto de Graduação – Engenharia de Automação. Universidade Federal do Rio Grande - FURG, Rio Grande.

A estruturação e implantação do contexto da indústria 4.0 é um conceito que está ganhando relevância na indústria e na academia, principalmente devido ao desenvolvimento tecnológico das últimas décadas. O rolamento é uma parte central e vulnerável de máquinas rotativas, então qualquer falha ou rachadura que venha ocorrer, gera diferentes efeitos, que variam conforme a carga de trabalho e velocidade, prejudicando a eficiência do equipamento. Para identificar e classificar possíveis falhas, sinais mecânicos de vibração são considerados como a melhor fonte de informação. *Machine Learning* é um método amplamente utilizado em diagnósticos de falhas. Existem duas etapas principais nos métodos de diagnóstico de falhas baseados em ML, uma é a extração e a seleção de característica de falha, a outra é a classificação dos tipos de falha. Embora os métodos tradicionais de ML provaram ter um bom desempenho no diagnóstico de falhas, ainda existem algumas deficiências. *Deep Learning* pode efetivamente resolver os problemas em métodos tradicionais baseados em ML, que dependem muito do conhecimento de especialistas para extrair características do sinal bruto analisado, pois ele pode extrair automaticamente recursos representativos do sinal bruto, que pode eliminar a influência de experiência na extração das características do sinal. Apesar do grande esforço e do grande número de trabalhos acadêmicos dedicados a esse campo, ainda existem alguns desafios importantes que precisam ser enfrentados para aplicar com êxito os algoritmos de ML e DL em aplicações do mundo real. A maioria dos trabalhos estudados, se utilizaram de conjuntos de dados disponíveis publicamente, coletados de laboratórios, para treinar seus algoritmos de ML ou DL. No entanto, seria ideal transferir a estrutura de rede e os parâmetros aprendidos para detectar falhas de rolamento de configurações anteriormente não vistas, e um exemplo muito promissor seria aprender a prever falhas de rolamento de ocorrência natural no mundo real usando apenas os dados coletados de falhas artificiais no laboratório. Neste trabalho, foi utilizado o modelo CNN baseado em Lenet-5 proposto por Long Wen, que utilizou um data set formado de danos artificiais para treinar sua rede. O objetivo é avaliar o modelo proposto quando submetido a dados que simulam falhas reais em rolamentos. A rede atingiu em média 83% de acurácia, o que foi considerado aceitável pelo autor, pois em trabalhos anteriores a acurácia em trabalhos semelhantes foram de 75%.

Palavras-chave: Lenet-5, Falha de rolamento, Manutenção preditiva, Classificação de imagens, Modelo baseado em dados.

ABSTRACT

RAMIS, Marcos Vinicius Moreira. **Titulo do Trabalho em Ingles**. 2021. 52 f. Projeto de Graduação – Engenharia de Automação. Universidade Federal do Rio Grande - FURG, Rio Grande.

The structuring and implementation of the industry 4.0 context is a concept that is gaining relevance in industry and academia, mainly due to the technological development of the last decades. The bearing is a central and vulnerable part of inductive machines, so any failure or crack that occurs, generates different effects, which vary according to the workload, speed, impairing the efficiency of the equipment. To identify and classify possible failures, mechanical vibration signals are considered to be the best source of information. ML is the method widely used in fault diagnosis. There are two main steps in fault diagnosis methods based on machine learning, one is the extraction and selection of fault characteristics, the other is the classification of fault types. Although traditional ML methods have proven to perform well in fault diagnosis, there are still some shortcomings. DL can effectively solve the problems in traditional ML-based methods, which rely heavily on expert knowledge to extract characteristics from the analyzed raw signal, as it can automatically extract resources representative of the raw signal, which can eliminate the influence of experience in extracting the characteristics the signal. Despite the great effort and the large number of academic works dedicated to this field, there are still some important challenges that need to be faced to successfully apply ML and DL algorithms in real-world applications. Most of the studies studied used publicly available data sets collected from laboratories to train their custom ML or DL algorithms. However, it would be ideal to transfer the learned network structure and parameters to detect bearing failures from previously unseen configurations, and a very promising example would be to learn to predict naturally occurring bearing failures in the real world using only the data collected from failures in the laboratory. In this work, the CNN model based on Lenet-5 proposed by Long Wen was used, which used a data set formed of artificial damages to train its network. The objective is to evaluate the proposed model when submitted to data that simulate real bearing failures. The network reached an average of 83% accuracy, which was considered acceptable by the author, because in previous works the accuracy in similar works was 75%.

Keywords: Lenet-5, Bearing Fault, Predictive maintenance, Image classification, Data-driven.

LISTA DE FIGURAS

Figura 1	A evolução industrial	9
Figura 2	Políticas de manutenção	10
Figura 3	Passos para aplicação do CBM	11
Figura 4	Curva PF	12
Figura 5	Sistema CWRU para testes de rolamento	13
Figura 6	Estrutura tradicional de um rolamento e a vista em corte transversal de um rolamento	14
Figura 7	Sinais esperados	15
Figura 8	Fluxo de redes de ML e DL	17
Figura 9	CNN característica de identificação de falha de rolamento	18
Figura 10	Coordenadas cartesianas 2-D, da imagem de tamanho(m,n)	20
Figura 11	CNN LeNet-5	22
Figura 12	Processo de Padding	23
Figura 13	Processo de Stride	24
Figura 14	Convolução	26
Figura 15	Pooling	27
Figura 16	Fully-Connected	27
Figura 17	Danos Artificiais	32
Figura 18	Danos de rolamento gerados pela bancada de teste de vida acelerado.	33
Figura 19	Bancada de testes da danos reais	33
Figura 20	Equação de Transformação	36
Figura 21	CNN Proposta	37
Figura 22	Matriz de confusão	41
Figura 23	Matriz de confusão Padderborn	45

LISTA DE TABELAS

Tabela 1	Parâmetros da estrutura tradicional da rede LeNet-5	22
Tabela 2	Parâmetros de operação CWRU	30
Tabela 3	Danos Artificiais	32
Tabela 4	Parâmetros de operação	34
Tabela 5	Tabela com dados naturais	34
Tabela 6	Tabela com a quantidade de imagens utilizadas para treinar e testar o modelo proposto	38
Tabela 7	Tabela comparativa dos sinais de vibração e das imagens geradas . .	39
Tabela 8	Parâmetros de cada camada da CNN	40
Tabela 9	Resultados da CNN proposta	40
Tabela 10	Tabela com os dados de treino, danos artificiais, e dados de teste, danos reais	41
Tabela 11	Tabela com os dados de treino, danos artificiais, e dados de teste, danos reais, referentes ao Padderborn	43
Tabela 12	Parâmetros de cada camada da CNN aplicada ao Padderborn	44
Tabela 13	Resultados da CNN aplicada ao data set Padderborn	44
Tabela 14	Comparação dos algoritmos aplicados ao dataset Padderborn	45

LISTA DE ABREVIATURAS E SIGLAS

FBM	Manutenção baseada em falha
PM	Manutenção preventiva
CBM	Condição baseada em manutenção
TPM	Manutenção produtiva total
IEEE-IAS	<i>Institute of Electrical and Electronic Engineers - Industry Applications Society</i>
JEMA	<i>Japan Eletrical Manufactures Association</i>
ABNT	Associação Brasileira de Normas Técnicas
ML	<i>Machine Learning</i>
DL	<i>Deep Learning</i>
CNN	<i>Convolutional Neural Network</i>
ANN	<i>Artificial Neural Network</i>
SAE	<i>Support Auto Encoder</i>
DBN	<i>Deep Believe Network</i>
SVM	<i>Support Vector Machine</i>
Pool	<i>Pooling</i>
Conv	<i>Convolution</i>
RGB	<i>Red, Green, Blue</i>
SGD	<i>Stochastic Gradient Descent</i>

SUMÁRIO

1	Introdução	9
1.1	Caracterização do Problema	9
1.2	Análise de falha em rolamentos	12
1.3	Objetivos	16
1.3.1	Objetivos Específicos	16
1.4	Revisão de Literatura	17
2	Referencial Teórico	19
2.1	Representação de imagem	19
2.2	Representação de cores	19
2.3	Espaço de Cores	21
2.3.1	Conversão da imagem RGB para imagem em escala cinza	21
2.4	A estrutura da CNN Lenet-5	22
2.5	Preenchimento	23
2.6	Passos de convolução	24
2.7	Camadas de convolução	24
2.8	Camada de Pooling	26
2.9	Camada totalmente conectada	27
2.10	Treinamento da CNN via <i>backpropagation</i>	27
2.11	Data Sets de falha de rolamento disponíveis para treinamento de ML e DL	28
2.11.1	CWRU Data set	29
2.11.2	Data set da Universidade Padderborn	30
3	Metodologia	35
3.1	Conversão do sinal 1D para uma imagem 2D	36
3.2	Estrutura CNN proposta	37
4	Resultados	38
4.1	CNN aplicada ao DataSet CWRU	38
4.2	Resultado da conversão do sinal	38
4.3	Resultado da CNN proposta	40
4.4	CNN aplicada ao DataSet Padderborn	41
4.5	Resultado da conversão do sinal Padderborn	42
4.6	Resultado da CNN proposta	44
5	Conclusão	46

1 INTRODUÇÃO

1.1 Caracterização do Problema

A estruturação e implantação do contexto da indústria 4.0 é um conceito que está ganhando relevância na indústria e na academia, principalmente devido ao desenvolvimento tecnológico, caracterizado pelo estabelecimento da comunicação entre dispositivos, possibilitando a troca, armazenamento e interpretação de dados em um sistema inteligente [Cordeiro, 2017]. A indústria 4.0, representa a quarta revolução industrial. Conforme a figura 1, a primeira revolução ocorreu em 1784, e foi marcada pelo desenvolvimento da mecanização e da força a vapor, em 1870 surgiu a produção em escala e linhas de montagem, contando com o apoio de conceitos inovadores da época, como, a eletricidade e a combustão, a terceira revolução industrial, ocorreu em 1969, período o qual ficou marcado pelo surgimento dos computadores, a internet, a robótica e a automação, e nos dias de hoje vivemos a indústria 4.0, com sistemas cibernéticos, internet das coisas, redes e inteligência artificial.

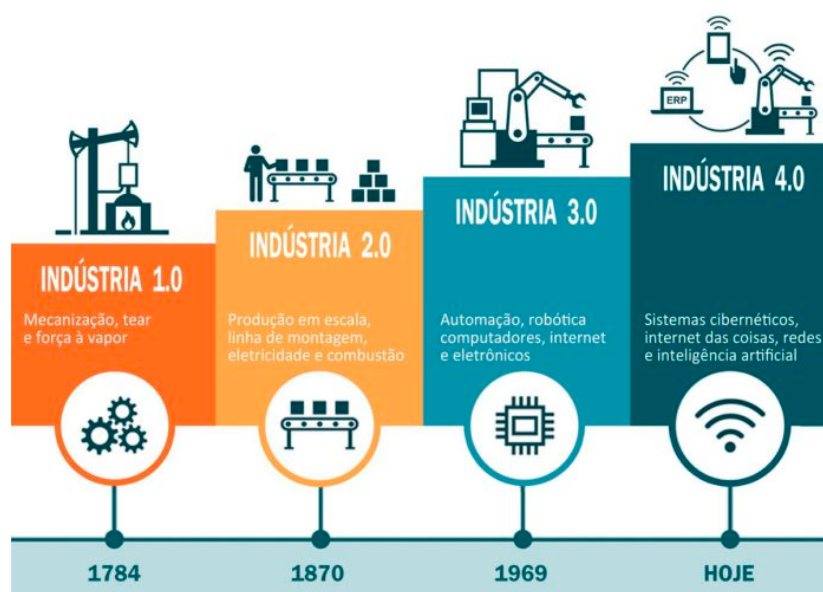


Figura 1: A evolução industrial
Fonte: [L.Egreja, 2019]

A manutenção desempenha um papel estratégico dentro de uma empresa, estudos mostram que, de 15% a 70% dos custos dos produtos manufaturados, são oriundos das atividades de manutenção [Krupitzer et al., 2020]. Para minimizar as paradas de manutenção, é primordial detectar antecipadamente as falhas nos equipamentos, e assim aumentar o tempo de disponibilidade, confiabilidade e a segurança das máquinas. Para isso a manutenção preditiva, se utilizando de conceitos da indústria 4.0, fig 1, como a inteligência artificial, é uma ótima ferramenta para atender essa demanda industrial [Krupitzer et al., 2020]. Além da manutenção preditiva existem outras políticas de manutenção, como manutenção baseada em falha (FBM), manutenção preventiva (PM), manutenção baseada em condição (CBM) e a manutenção produtiva total (TPM), conforme a figura 2.



Figura 2: Políticas de manutenção
Fonte: [D.Velázquez et.al, 2018]

O FBM, também chamada de manutenção corretiva, é uma técnica de manutenção reativa. É um conceito que no qual só irá atuar quando a máquina apresentar defeito, e concertando o mais rápido possível [Al-Najjar, 1997]. Na manutenção corretiva, a falha no equipamento pode ocorrer a qualquer momento, inclusive em momentos inconvenientes. Esse tipo de evento, aumenta o número de indisponibilidade de máquina, bem como os custos de produção. Para dar conta desse tipo de técnica, é necessário ter um alto número de peças no estoque, equipamentos reservas o que aumenta os custos com manutenção [Al-Najjar et al., 2018].

A Manutenção preventiva (PM), é baseada em manutenções agendas conforme o tempo de operação do equipamento. Esse tempo é definido por especialistas ou pelo fabricante do equipamento ou ferramenta [Pintelon and Parodiherz, 2008]. No entanto, existe a probabilidade do equipamento sofrer paradas desnecessárias, parando a produção sem necessidade.

Para realizar um programa de ações voltado a manutenção preditiva, as empresas podem se utilizar dos conceitos da *Condition Based Maintenance* (CBM). O processo de implementação do CBM segue conforme a figura 3, o processo possui quatro passos, a) Avaliação operacional, b) Aquisição de dados, c) Ferramentas de análise e d) Modelos de análise.

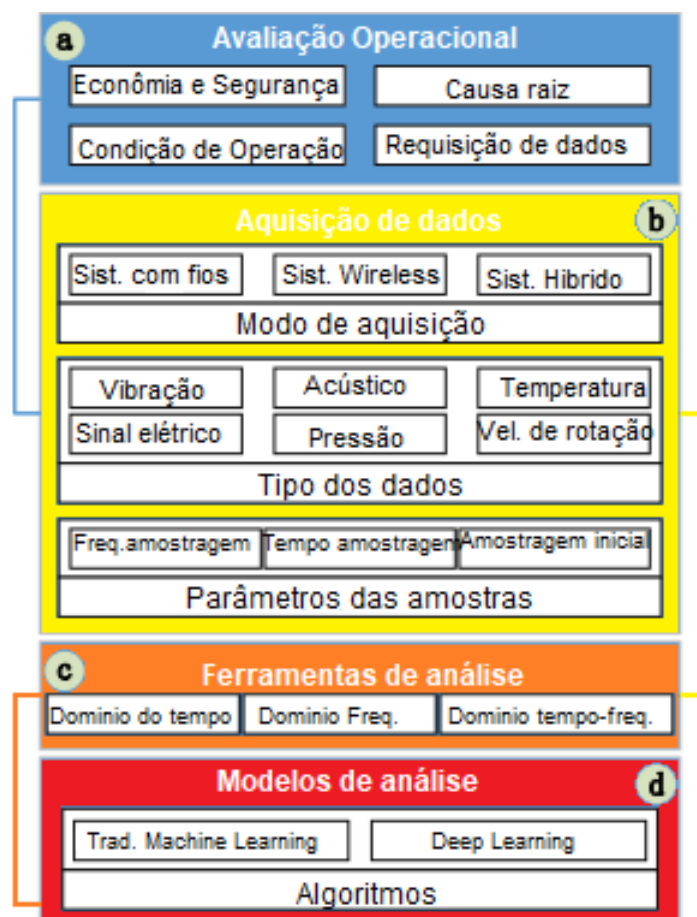


Figura 3: Passos para aplicação do CBM
Adaptado de: [W.Zhang, 2019]

A Manutenção Produtiva Total (TPM), é uma filosofia que envolve toda a organização, desde o executivo até ao chão de fábrica. Os objetivos do profissional de manutenção são buscar diariamente reduzir o tempo de parada de máquina, diminuir o desperdício de material, melhorar a qualidade e a efetividade geral do equipamento (OEE). A efetividade geral do equipamento é utilizada no TPM para medir a efetividade técnica do equipamento, ou seja o quanto bem ele desenvolve a sua atividade [Chan and Prakash]. Originalmente o TPM não tem uma técnica especial para efetivamente utilizar dados ou tecnologias para a utilização de condição de parâmetros [Al-Najjar and Ingwald].

A partir da análise de sinais do equipamento a ser monitorado, se pode prever futuras falhas. Assim, o tempo da descoberta, da falha em potencial deve ocorrer tempo o suficiente para a manutenção, eliminar ou evitar que a falha ocorra, antes que a produção seja afetada[Zhang et al., 2020]. Para isso se usa a curva PF, que define o ponto de falha em potencial e o ponto de falha funcional, conforme a figura 4.

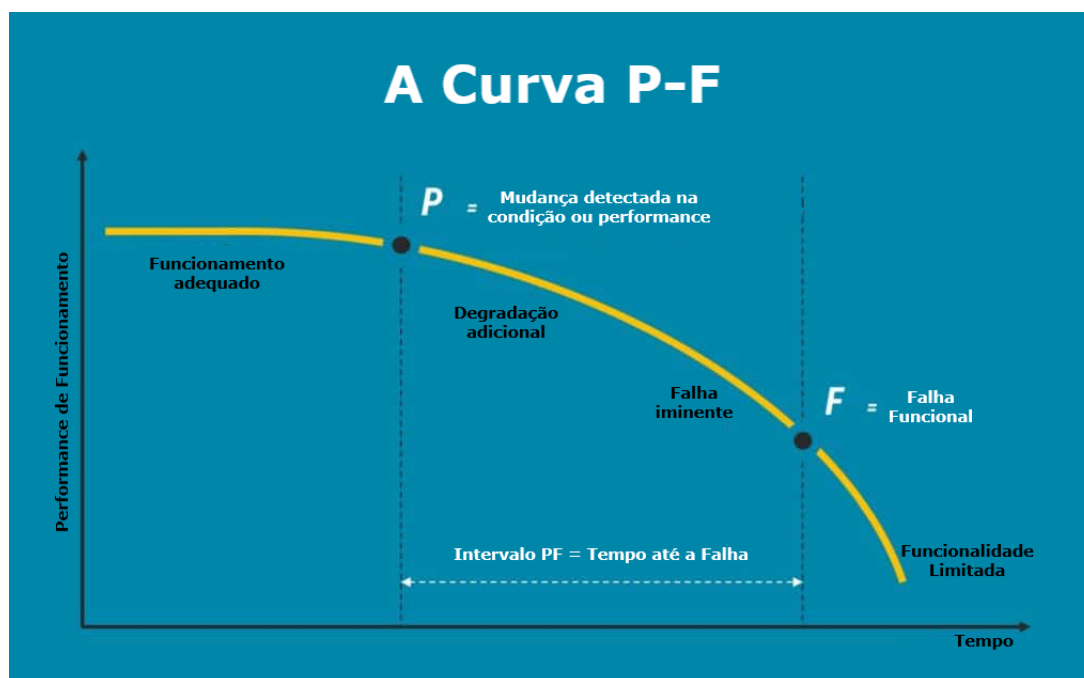


Figura 4: Curva PF

Adaptado de: [ENGETELES-Engenharia de manutenção, 2017]

Segundo o *Institute of Electrical and Electronic Engineers - Industry Applications Society* [IEEE-IAS] e a *Japan Eletrical Manufactures Association* [JEMA], a quebra de rolamentos representa de 30% a 40% das falhas em motores indutivos. Para realizar a análise de falha de rolamentos, diversos sinais já foram estudados, vibração, ruído acústico, corrente, imagem termal, e fusão de múltiplos sensores, no entanto a análise do sinal de vibração continua dominante [Zhang et al., 2020].

1.2 Análise de falha em rolamentos

O rolamento é uma parte central e vulnerável de máquinas indutivas. Então qualquer falha ou rachadura que venha ocorrer, gera diferentes efeitos, que variam conforme a carga de trabalho, velocidade, ou que podem afetar a estabilidade e ou a eficiência do equipamento [Neupane and Seok, 2020]. A estrutura do rolamento consiste em 4 componentes: pista-interna (*inner-race*), pista-externa (*outter-race*), esfera(*ball*) e a gaiola(*cage*). Na Fig. 5 mostra o experimento para o sistema de rolamentos, do centro de dados da CWRU.

Com o intuito de obter informações confiáveis sobre o estado atual do equipamento, o monitoramento da condição de um equipamento e um sistema de análise de falhas de

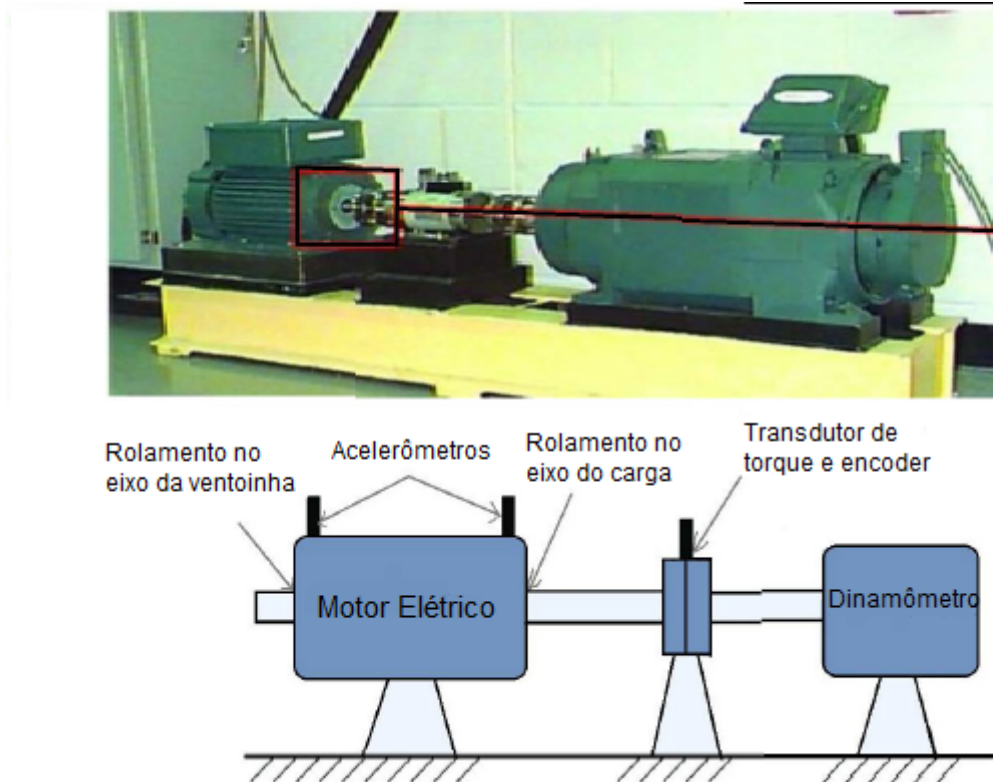


Figura 5: Sistema CWRU para testes de rolamento
Adaptado de: [Neupane, 2020]

rolamento, ajuda a reduzir a parada de produção. Para identificar e classificar possíveis falhas, sinais mecânicos de vibração são considerados como o melhor e mais produtiva fonte de informação [Boudiaf et al., 2016, Neupane and Seok, 2020, Safizadeh and Latifi, 2014].

A técnica de diagnóstico de rolamento mais clássica é análise de envelope, onde obtemos uma série de respostas de impulso de frequências repetidas. Os impulsos surgem tanto na passagem da falha para dentro e para fora da zona de carga quando há variação no caminho de transmissão da falha para o ponto de medição [McFadden and Smith, 1984, Smith and Randall, 2015].

A vibração produzida pela velocidade de rotação produz frequências características de cada componente do rolamento, conforme a figura 6.

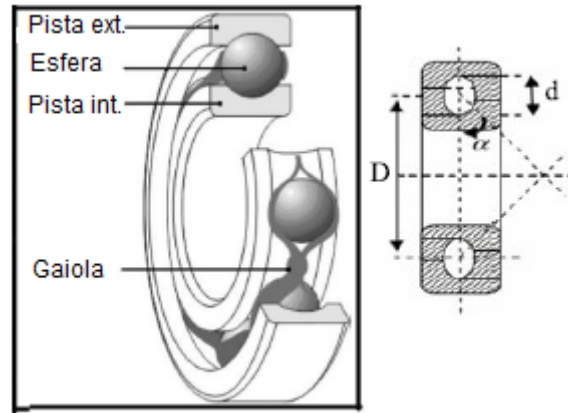


Figura 6: Estrutura tradicional de um rolamento e a vista em corte transversal de um rolamento

Adaptado de: [Neupane, 2020]

As frequências se referem a rotação das esferas e da passagem das esferas nas pistas internas e externas [Boudiaf et al., 2016]. As frequências de falha de rolamento associadas à pista interna(1), pista externa(2), gaiola(3) e esfera(4), seguem conforme as seguintes equações:

$$BPFI = \frac{n f_r}{2} \left(1 + \frac{d}{D} \cos(\theta) \right) \quad (1)$$

$$BPFO = \frac{n f_r}{2} \left(1 - \frac{d}{D} \cos(\theta) \right) \quad (2)$$

$$FTF = \frac{f_r}{2} \left(1 - \frac{d}{D} \cos(\theta) \right) \quad (3)$$

$$BSF = \frac{D f_r}{2d} \left(1 - \left[\frac{d}{D} \cos(\theta) \right]^2 \right) \quad (4)$$

onde, BPFI é a frequência de passagem da esfera na pista interna, BPFO é a frequência de passagem da esfera na pista externa, FTF é a frequência fundamental (velocidade da gaiola), BSF é a frequência de giro da esfera. Além disso, f_r é a velocidade do eixo, n é o número de esferas, d é o diâmetro da esfera, D é o diâmetro entre as esferas e θ é o ângulo da esfera em relação ao plano radial. As frequências são calculadas a partir de relações cinemáticas, assumindo que não existe escorregamento, no entanto sempre há, portanto uma variação entre 1-2% é comum. Figura 7 descreve alguns sinais de envelope baseados na teoria de McFadden and Smith [1984] e Smith and Randall [2015].

Geralmente, os métodos de diagnóstico de falhas podem ser divididos em orientado

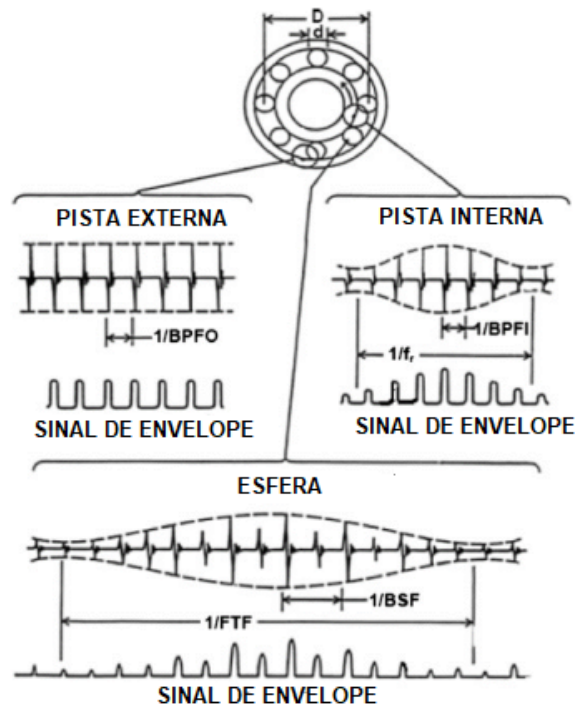


Figura 7: Sinais esperados
Adaptado de: [Smith, 2015]

por modelo, por experiência e por dados [Gao et al., 2015].

O método de diagnóstico de falhas baseado em dados pode aprender e extrair falhas sem qualquer conhecimento prévio, o que é ideal para sistemas complicados que são difíceis de construir um modelo [Tong Guan et al., 2017].

Recentemente, com o rápido desenvolvimento de sistemas inteligentes, dados de monitoramento de condição do equipamento são coletados e armazenados para análise posterior. É necessário extrair informações úteis desses dados massivos e aplicar essas informações ao diagnóstico de falhas baseado em dados [Liu and Huang, 2019, Gandomi and Haider, 2015].

O aprendizado de máquina é um método amplamente utilizado em diagnóstico de falha. Existem duas etapas principais nos métodos de diagnóstico de falhas baseados em ML, um é a extração e a seleção de característica de falha, a outra é a classificação dos tipos de falha [Liu and Huang, 2019, Ince et al., 2016]. Embora os métodos tradicionais de aprendizado de máquina provaram ter um bom desempenho no diagnóstico de falhas, ainda existem algumas deficiências. Esses recursos extraídos dependem muito do conhecimento dos especialistas e na experiência de diagnóstico.

Deep Learning (DL) pode efetivamente resolver os problemas de métodos tradicionais baseados ML, porque ele pode extrair automaticamente recursos representativos do sinal bruto [Lv et al., 2016, Liu and Huang, 2019].

Apesar do grande esforço e do expressivo número de trabalhos acadêmicos dedicados

a esse campo, ainda existem alguns desafios importantes que precisam ser enfrentados para aplicar com êxito os algoritmos de ML e DL em aplicações do mundo real. A maioria dos trabalhos estudados, usou conjuntos de dados disponíveis publicamente coletados de laboratórios para treinar seus algoritmos de ML ou DL personalizados. No entanto, seria ideal transferir a estrutura de rede aprendida e os parâmetros para detectar falhas de rolamento de configurações anteriormente não vistas, e um exemplo muito promissor seria aprender a prever falhas de rolamento de ocorrência natural no mundo real usando apenas os dados coletados de falhas artificiais no laboratório [Zhang et al., 2020]. Existem duas opções para testar e desenvolver métodos de monitoramento de condição com danos do mundo real: medir os danos em máquinas em operação ou em bancadas de teste científicas. O uso bem-sucedido de dados oriundos da indústria é bastante difícil, pois é complicado receber dados de treinamento sistemáticos e comparáveis para diferentes danos. Isso se deve à longa vida útil da maioria dos rolamentos e, se forem detectados danos, os rolamentos são substituídos antes da falha, de modo que raramente ocorrem defeitos. Portanto, frequentemente, apenas um pequeno número de estados de dano está disponível. Além disso, existem muitos tipos de rolamentos, tamanhos e tipos de máquinas diferentes, e as condições de operação podem mudar irregularmente, pois dependem da aplicação. Para reduzir as influências externas, são utilizadas bancadas de testes científicos, que permitem a geração de danos realistas aos rolamentos por meio de testes de vida útil acelerado [Lessmeier et al., 2016].

Neste trabalho, foi comparado a acurácia do modelo CNN baseado em Lenet-5 proposto por Wen et al. [2018], que utilizou o data set CWRU que dispõe de dados de danos artificiais em rolamento para validação do seu modelo, e comparar a acurácia adquirida com o data set Paderborn, que dispõe de dados de danos de rolamento reais, ou seja de máquinas em operação. Portanto, será analisada a performance do modelo CNN proposto, quando aplicado ao data set que fornece dados de danos reais de equipamentos.

1.3 Objetivos

O objetivo desse projeto é analisar o modelo CNN proposto por Long Wen quando submetido a dados que simulam falhas reais em rolamentos.

1.3.1 Objetivos Específicos

Os objetivos específicos desse projeto são:

- Replicar o modelo proposto por Long Wen com o data set de danos artificiais CWRU
- Aplicar o modelo proposto por Long Wen no data set de danos reais Paderborn

1.4 Revisão de Literatura

Modelos baseados em dados tem sido amplamente aplicados a manutenção preditiva de indústrias, por meio de *machine learning*(ML) [Jiang and Yin, 2018] e *deep learning*(DL)[You et al., 2015]. Algoritmos tradicionais de machine learning, como, Regressão Logística(LR), *support vector machine*(SVM), *decision tree* (DT), e *random forest* (RF) geralmente necessitam de um grande volume de dados das condições do equipamento para conseguir treinar o modelo. Nos modelos de ML a partir do sinal no domínio do tempo extraído dos acelerômetros, se deve extrair as características de vibração sobre esse sinal, assim se utiliza de equações no domínio do tempo ou frequência e tempo-frequência. No entanto, para treinar modelos de DL não se faz necessário analisar as características de vibração, e por meio de diversas camadas entre os dados brutos e a saída, é realizado um aprendizado de ponta-a-ponta, conforme a figura 8.

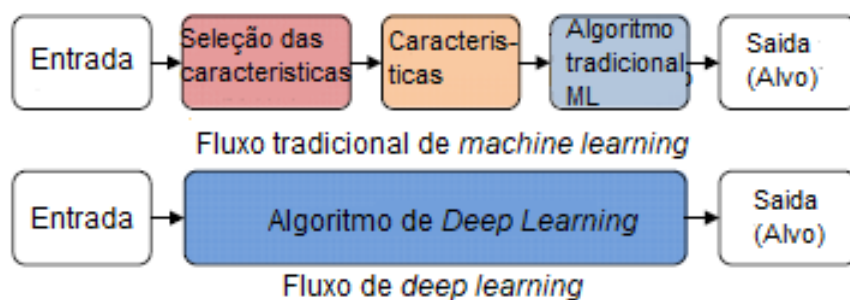


Figura 8: Fluxo de redes de ML e DL
Adaptado de: [W.Zhang, 2019]

Inspirado em como funciona o córtex visual animal, surgiu as redes neurais convolucionais. Uma rede neural convolucional é uma classe de rede neural artificial do tipo *feed-forward*, que vem sendo aplicada com sucesso no processamento e análise de imagens. Uma CNN usa uma variação de perceptrons multicamada desenvolvidos de modo a demandar o mínimo pré-processamento possível [Zhang et al., 2020].

Um dos primeiros artigos utilizando CNN para identificação de falha de rolamento, foi publicado em 2016 [Janssens et al., 2016]. Desde então começaram a surgir muitos artigos com a mesma intenção, identificação de falha de rolamento. Na figura 9, está a arquitetura básica de uma CNN para classificar falhas de rolamento. Os dados brutos em 1-D temporal, de diferentes acelerômetros são transformados em imagens 2-D. Assim, as imagens de entrada passam pela camada de convolução para extrair características da imagem, depois passam pela camada de *pooling* para diminuir as dimensões da imagem. A combinação Convolução-Pool é repetida, no entanto a repetibilidade desse padrão se altera dependendo da rede. Finalmente, a saída dessas camadas irá passar pela camada *fully-connected*, e o resultado é transferido para o classificador baseado nas funções, Soft-max ou Sigmoid para determinar qual a falha do rolamento[Zhang et al., 2020].

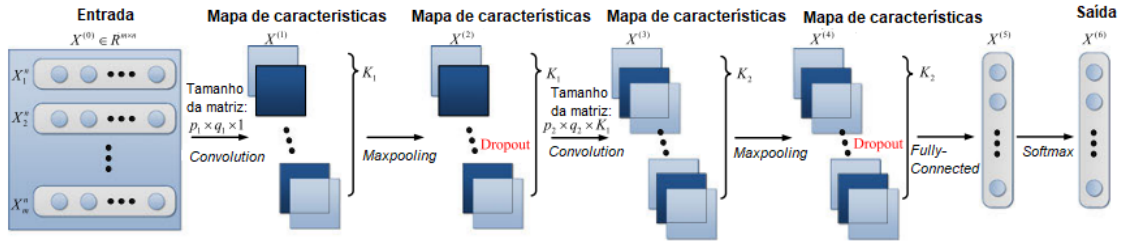


Figura 9: CNN característica de identificação de falha de rolamento
Adaptado de: [S.Zhang, 2020]

Em Xia et al. [2018] foi abordado uma fusão do sinal dos sensores, no qual informações provenientes do data set CWRU, oriundos do eixo de acionamento são transformados de 1-D temporal para 2-D em forma de matriz. A precisão foi calculada a partir da fusão de dois sensores, o que aumentou a acurácia do modelo de 99.41% para 98.35% comparado com o modelo que utilizou apenas um sensor. Diversos modelos de CNN foram aplicados ao data set CWRU por apresentar características bem definidas das falhas de rolamento, é um data set muito aplicado para testar novos modelos de CNN. Por exemplo a CNN baseada em Lenet-5, apresentada por [Wen et al., 2018], a qual contém 4 camadas, convolução-pooling, e duas camadas *fully-connected*.

A rede neural utiliza *padding* para controlar as dimensões, o zero-padding é utilizado para prevenir perdas de dimensão. Essa rede obteve ótimos resultados em aprender as características dos defeitos de rolamento, com acurácia de 99.79%. Comparada com outros métodos como *sparse filter*, DBN, SVM, que obtiveram acurácias de, 99,66%, 87,45% e 87,45% respectivamente. A CNN também foi comparada a um ANN que obteve 67,7% de acurácia, o que é inferior a CNN proposta por Long Wen, mostrando a melhora na acurácia que o modelo propõem [Wen et al., 2018, Zhang et al., 2020].

2 REFERENCIAL TEÓRICO

Este capítulo é destinado a revisar os conceitos necessários para realização deste projeto. O capítulo apresenta as definições sobre representação de imagem, representação de cores, espaço de cores, estrutura da CNN LeNet-5, Preenchimento, Passos de Convolução, Camadas de convolução, Camada de pooling, Camada totalmente conectada, Treinamento da CNN via backpropagation e Data Sets disponíveis para treinamento.

2.1 Representação de imagem

Uma imagem pode ser considerada como uma representação discreta, que possui informações sobre o espaço (plano) e intensidade (cor) de algo. Por meio de um processo chamado discretização, se pode representar sinais contínuos unidimensionais (1-D) ou bidimensionais (2-D) em uma plano cartesiano 2-D. A imagem, $I(m,n)$, pode representar a resposta de um valor de interesse por meio de um série de pontos fixos em posições no plano cartesiano ($m = 1,2,\dots,M$; $n = 1,2,\dots,N$). Vale ressaltar que a discretização é feita de forma automática em alguns equipamentos e sensores (como câmeras e sensores de vibração), e basicamente afetam a média local de um sinal contínuo apenas em pequenas regiões específicas. Os índices, m e n , indicam respectivamente a linha e a coluna da imagem, portanto imagens podem ser representadas pelo seu endereço 2-D (m,n). Seguindo as convenções do *MATLAB*[®], a biblioteca do Python, *matplotlib.pyplot* é uma coleção de funções que faz o *matplotlib* funcionar como o *MATLAB*[®]. Assim, $I(m,n)$ denota a resposta do pixel localizado na m th linha e n th coluna, começando a partir do canto esquerdo da imagem de origem, conforme a figura 10. Em outros sistemas de imagem, a convenção coluna-imagem pode ser usada [Solomon and Breckon, 2013].

2.2 Representação de cores

Uma imagem contém um ou mais canais que definem a intensidade ou a cor em um pixel em particular $I(m,n)$. Em casos simples, cada pixel contém um único valor numérico que representa a intensidade de um sinal em uma imagem. A conversão desse número para uma imagem real, é realizada por meio do mapa de cores. Com o objetivo de obter

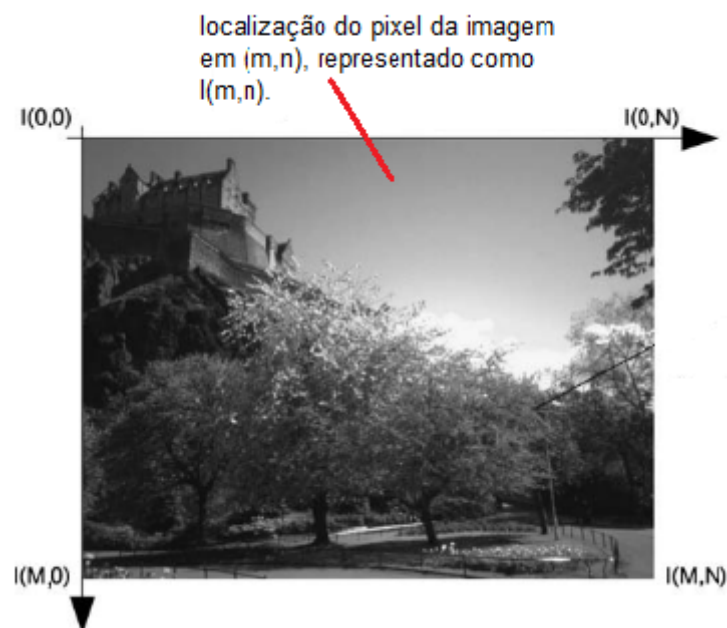


Figura 10: Coordenadas cartesianas 2-D, da imagem de tamanho(m,n)
Adaptado de: [C.Solomon, 2013]

uma representação visual dos dados, o mapa de cores define a tonalidade de cor específica para cada nível numérico da imagem. Uma das representações de cores mais utilizada é a escala cinza, que define a imagem a partir da intensidade do sinal, utilizando todas as tonalidades da cor cinza, variando de preto (Zero), a branco (valor máximo). A escala cinza, é adequado para representar imagens com valores únicos, a partir da intensidade de sinais captados [Solomon and Breckon, 2013]. Em alguns casos de representação de imagens por intensidade, é melhor utilizar o falso mapa de cor. Um dos principais motivos para se utilizar o falso mapa de cor, é o fato do sistema visual humano ter a capacidade de diferenciar aproximadamente 40 tons de cinza em uma escala que vai das cores preta a branca, a qual sensibilidade das cores são mais sutis. Além disso, o procedimento serve para acentuar ou delinear algumas características, tornando mais fácil a identificação pelo olho humano. A definição do mapa de cores, pode ser definida conforme as necessidades do usuário, no entanto a intensidade, tanto de cores ou em escala cinza, na maioria dos casos são lineares. Apesar disso, existem casos em que a não linearidade é mais apropriada [Forsyth and Ponce, 2002]. Além da escala cinza, representando um pixel por meio de um único valor numérico, se tem a representação por cores, onde se tem um vetor triplo(R,G,B) que representam todo o espectro de cores. As cores são representadas por meio de uma combinação linear básica de cores ou valores e a imagem consiste em três matrizes 2-D.

2.3 Espaço de Cores

Como brevemente comentado anteriormente, a representação de cores em uma imagem é atingida utilizando combinação de um ou mais canais de cores, que combinadas formam as cores utilizadas na imagem. As representações utilizadas para salvar as cores, especificando o número e o tipo de canais de cores, são denominadas como espaço de cores. A escala cinza (intensidade), é uma matriz 2-D que assume um valor numérico para cada pixel que representa a intensidade em um ponto. Em contraste, imagens RGB possuem matrizes 3-D, que assumem valores para cada pixel, sendo que cada valor representa uma componente, vermelho, verde e azul, respectivamente.

2.3.1 Conversão da imagem RGB para imagem em escala cinza

A conversão de uma imagem RGB para escala cinza é realizada utilizando uma simples transformação. A conversão para o mapa de cor em escala cinza é o passo inicial em muitos algoritmos de análise de imagem. Apesar da escala cinza apresentar menos informações que a imagem RGB, características importantes como, bordas, regiões, bolhas, junções entre outras são mantidas. Em algoritmos de detecção de características e processamento, geralmente utilizam a imagem em versão escala cinza, pois diminui o tempo de treinamento de uma rede neural [Kanan and Cottrell, 2012]. A equação de conversão da RGB para escala cinza é a seguinte:

$$I_{cinza}(m, n) = \alpha I_{vermelho}(m, n, r) + \beta I_{verde}(m, n, g) + \gamma I_{Azul}(m, n, b) \quad (5)$$

Onde, n, m representa os pixels de uma imagem em escala cinza, e escala de cor, vermelho, verde e azul. As letras r, g e b representam a localização do pixels nos canais de cores, vermelho, verde e azul respectivamente. Como se pode ver na equação 5, a imagem em escala cinza é a soma dos pesos das cores, vermelho, verde e azul.

2.4 A estrutura da CNN Lenet-5

A rede tradicional LeNet-5, fig 11 proposta por Lecun et al. [1998] foi originalmente proposta para reconhecer algarismos escritos com a mão. A rede consiste em duas camadas de convolução, duas camadas de *pooling* e três camadas *full-connection*. A estrutura tradicional de uma rede LeNet-5 segue conforme a tabela 1.

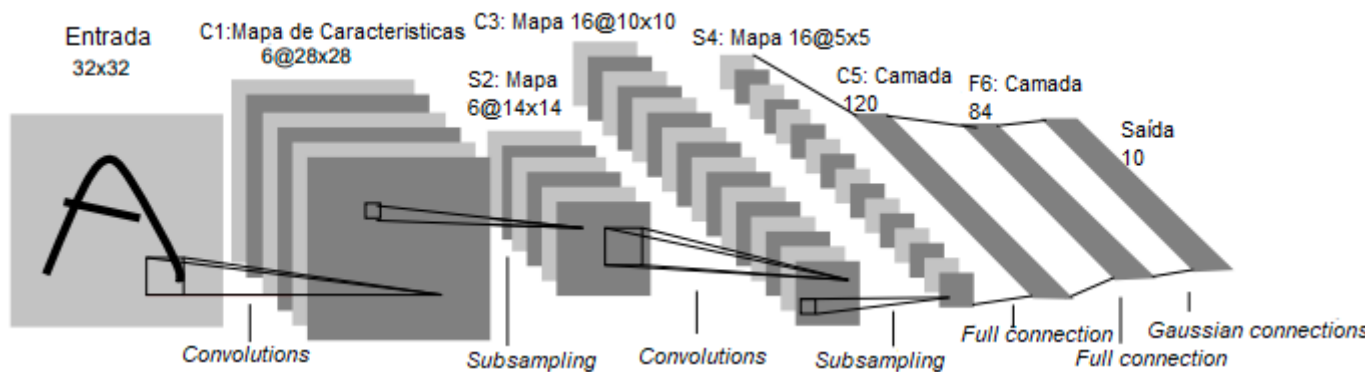


Figura 11: CNN LeNet-5
Adaptado de: [Y. Lecun, 1998]

Camadas da Rede	Configuração	Número de parâmetros treinados	Característica de saída
Entrada	image em preto e branco, 32x32 pixels	0	32x32x1
Conv 1	6 matrizes 5x5, stride=1	156	28x28x6
Pool 1	Pool 2x2, stride=2	12	14x14x6
Conv 2	16 matrizes, 5x5, stride=1	1516	10x10x16
Pool2	Pool 2x2, stride=2	32	5x5x16
FC1	120 neurônios	48120	1x1x120
FC2	84 neurônios	10164	1x84
FC3	10 neurônios	840	1x10

Tabela 1: Parâmetros da estrutura tradicional da rede LeNet-5

Como ilustrado na tabela 1, a entrada da rede é uma imagem preta e branca com o tamanho 32x32 pixels. A primeira camada, Conv1, utiliza um *Kernel* do tamanho 5x5 para gerar seis mapas de características de tamanho 28x28 pixels. A camada Pool1, utiliza uma matriz 2x2 com *max-pooling* na saída da camada Conv1, gerando seis mapas de características do tamanho, 14x14 pixels. A camada Conv2 usa dezesseis *kernels* de 5x5 para gerar dezesseis mapas de características de 10x10 pixels. A Pool2 utiliza uma matriz 2x2 com a operação max-pooling a partir do valor de saída da Conv2, o que gera dezesseis mapas de características de 5x5 pixels. A FC1 é uma camada totalmente conectada que possui 120 neurônios, a qual está totalmente conectada a camada pool2, o que produz cento e vinte mapas de características de 1x1 pixel. A FC2, é uma camada totalmente conectada com 84 neurônios, aqui se calcula o produto entre o vetor e os pesos do vetor e adiciona o valor de bias, e o resultado é a saída da função *sigmoid*. A camada FC3, também chamada como camada de saída, gera 10 neurônios e divide as imagens de entrada em 10 categorias diferentes, variando de 0-9.

2.5 Preenchimento

Padding é um termo relevante quando se trata de redes neurais de convolução, o termo se refere a pixels adicionados a imagem quando o processo de *Kernel* começa. Se um *padding* em uma CNN é definido como zero, *zero-padding*, então cada pixel adicionado a imagem de entrada será zero, conforme a figura 12.

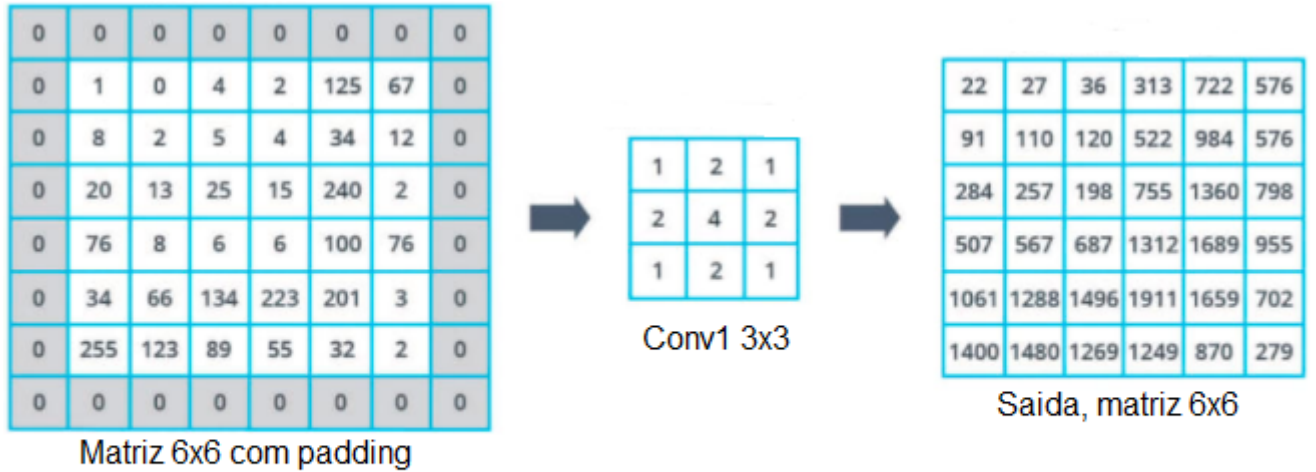


Figura 12: Processo de Padding
Adaptado de: [S. Balachandran, 2018]

O preenchimento funciona estendendo a área da qual uma rede neural convolucional processa uma imagem. O *kernel* é o filtro de redes neurais que se move pela imagem, varrendo cada pixel e convertendo os dados em um formato menor ou, às vezes, maior. Para ajudar o *kernel* a processar a imagem, o preenchimento é adicionado ao quadro da imagem para permitir mais espaço para o *kernel* cobrir a imagem. Adicionar preenchimento, segundo Lecun et al. [1998], a uma imagem processada por uma CNN permite uma análise mais precisa das imagens. O valor de *padding* a ser adicionado a esquerda e a direita da imagem, é calculada pelo seguinte:

$$N = \text{ceil} \left(\frac{M}{S} \right) \quad (6)$$

$$PT = (N - 1) \cdot S + F - M \quad (7)$$

$$PL = \text{floor} \left(\frac{PT}{2} \right) \quad (8)$$

$$PR = PT - PL \quad (9)$$

onde M é o tamanho da matriz de entrada, N o tamanho da matriz de saída, F o filtro, S o valor de *stride*. O número de *padding* total, para o lado esquerdo e pelo lado direito, são definidos como, PT , PL e PR respectivamente [Wen, 2020].

2.6 Passos de convolução

Stride é um parâmetro muito utilizado em redes neurais que como o *Kernel* irá se movimentar sobre a imagem ou vídeo. Por exemplo, se uma rede neural com *stride*=1, o filtro irá se movimentar um pixel, ou uma unidade, por vez. Como o tamanho do *kernel* afeta a matriz de saída, o *stride* é um valor inteiro, em vez de fração ou valor decimal. Se o tamanho do filtro é 3 x 3 pixels, então ele contém 9 pixels, que serão convertidos em 1 pixel na matriz de saída. Portanto, quanto maior valor definido para o *stride*, menor será a dimensão da matriz de saída. *Stride* é um parâmetro que trabalha de forma conjunta ao *Padding*, característica que adiciona pixels a matriz de saída, para possibilitar a redução mínima da matriz resultado. *Padding* e *Stride* são parâmetros fundamentais em qualquer rede neural. Nesse trabalho o valor de *stride* foi definido sendo igual a 1.

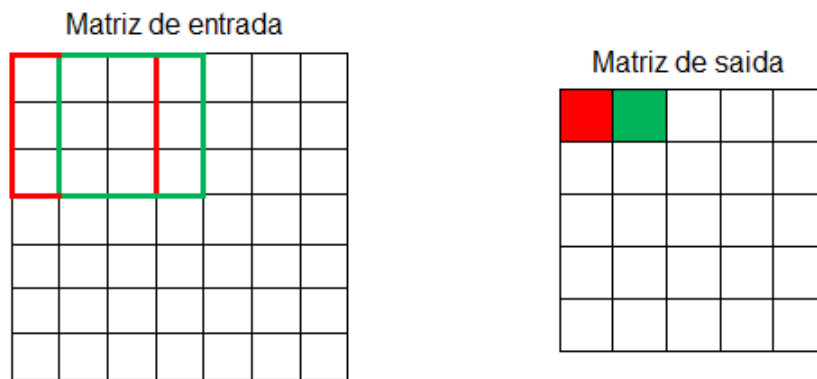


Figura 13: Processo de Stride
Fonte: [Gonzalez and Woods, 2011].

2.7 Camadas de convolução

Uma camada convolucional contém um conjunto de filtros cujos parâmetros precisam ser aprendidos. A altura e o peso dos filtros são menores do que os do volume de entrada. É realizada a convolução de cada filtro com o volume de entrada para calcular um mapa de ativação feito de neurônios. Em outras palavras, o filtro é deslizado ao longo da largura e altura da entrada e os produtos escalares entre a entrada e o filtro são calculados em cada posição espacial. Formalmente, o valor da característica na localização (i, j) referente

ao k-ésimo mapa de características, resultado das operações de cada filtro, da l-ésima camada, conjunto de filtros aplicados a uma mesma entrada, $z_{i,j,k}^l$ é calculado como:

$$Z_{i,j,k}^l = \mathbf{W}_k^{lT} \mathbf{X}_{i,j}^l + b_k^l. \quad (10)$$

onde $x_{i,j}^l$ é o fragmento da entrada centrada na posição (i, j) referente à l-ésima camada; w_k^l e b_k^l são o vetor de pesos e o termo de bias do k-ésimo filtro da l-ésima camada, respectivamente.

A conectividade local da camada convolucional permite que a rede aprenda filtros que respondem ao máximo a uma região local da entrada, explorando assim a correlação local espacial da entrada (para uma imagem de entrada, um pixel é mais correlacionado com os pixels próximos do que com os pixels distantes). Além disso, como o mapa de ativação é obtido realizando a convolução entre o filtro e a entrada, os parâmetros do filtro são compartilhados para todas as posições locais. O compartilhamento de peso reduz o número de parâmetros para eficiência de aprendizagem e boa generalização.

A primeira etapa em uma rede neural é fazer a convolução da entrada. Suponha que o tamanho de entrada de uma imagem seja $32 \times 32 \times 3$. A melhor maneira de expressar essas camadas é imaginar uma lanterna que brilha no canto superior esquerdo da imagem, conforme a figura 14. A lanterna cobre uma área de 5×5 , agora, essa área desliza pelas imagens de entrada. No aprendizado de máquina, essas lanternas são chamadas de filtros (também conhecidos como *kernels*) e a região sobre a qual elas brilham é chamada de campos receptivos. O filtro deve ser igual à profundidade da entrada, de modo que essas dimensões sejam $5 \times 5 \times 3$. A primeira posição do filtro pode ser no canto superior esquerdo. Conforme o filtro desliza sobre a convolução em torno das imagens de entrada, ele multiplica os valores no filtro com a entrada original da imagem (multiplicações por elemento). Essas multiplicações são todas somadas, então agora temos um único número. Repete o processo. A próxima etapa é mover esses filtros para 1 unidade e, em seguida, para a direita novamente 1 unidade, isso caso o stride seja igual a um. Cada local exclusivo no volume de entrada produz um número após deslizar o filtro sobre todos os locais. Agora, esses mapas de recursos ou função de ativação são mapeados como uma matriz de tamanho de $28 \times 28 \times 1$ (porque o tamanho de entrada é $32 \times 32 \times 3$). O motivo pelo qual obteremos uma precisão de 28×28 é que há um total de 784 locais diferentes, ou seja, filtros 5×5 em imagens de entrada de 32×32 . Esses 784 são mapeados com matrizes 28×28 .

Em geral, é aplicada uma função de ativação não-linear aos resultados da convolução [Karpathy A., 2016]. Embora tal função não seja obrigatória a uma camada de convolução, ela permite que a CNN detecte e aprenda características não-lineares, fazendo com que quase todas as camadas convolucionais sejam seguidas de uma função de ativação não-linear. Seja $f(\bullet)$ a função de ativação não-linear, então, o valor que a função

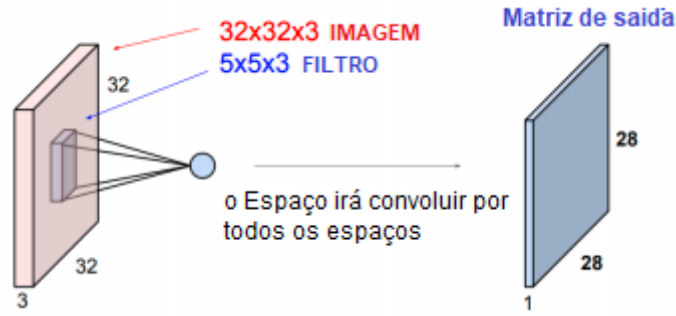


Figura 14: Convolução
Adaptado de: [Karpathy et al., 2016].

de ativação retorna quando a entrada é a característica convolucional $z_{i,j,k}^l$ pode ser calculado como:

$$a_{i,j,k}^l = f(z_{i,j,k}^l) \quad (11)$$

Funções de ativação comumente utilizadas são as funções sigmoide, tangente hiperbólica Lecun et al. [1998] e, mais recentemente, ReLU [Krizhevsky et al., 2017]. Nesse trabalho a função de ativação utilizada foi a ReLU.

2.8 Camada de Pooling

A camada de *pooling* visa reduzir o tamanho dos mapas de características considerando a semelhança de valores vizinhos [Karpathy A., 2016]. Esse objetivo é alcançado ao definir um novo mapa de característica através do *downsampling* do mapa de característica gerado pela camada de convolução. O *downsampling* é realizado ao associar um conjunto de valores vizinhos do mapa de característica anterior a um único valor no novo mapa de característica.

Formalmente, seja a função de *pooling* escrita como $\text{pool}(\bullet)$, então, a saída gerada pela função de *pooling* para o mapa de características $a_{i,j,k}^l$, é representada pela expressão:

$$y_{i,j,k}^l = \text{pool}(a_{m,n,k}^l), \forall (m, n) \in \mathcal{R}_{i,j} \quad (12)$$

onde $\mathcal{R}_{i,j}$ é uma vizinhança local centrada na posição (i, j). As operações típicas de pooling são:

- *max pooling*, que retorna apenas o maior valor dentro de seu campo receptivo [Boureau et al., 2010];
- *average pooling*, que retorna a média dos valores do seu campo receptivo [Wang et al., 2012].

A Figura 15 ilustra a operação de max pooling, utilizada neste trabalho.

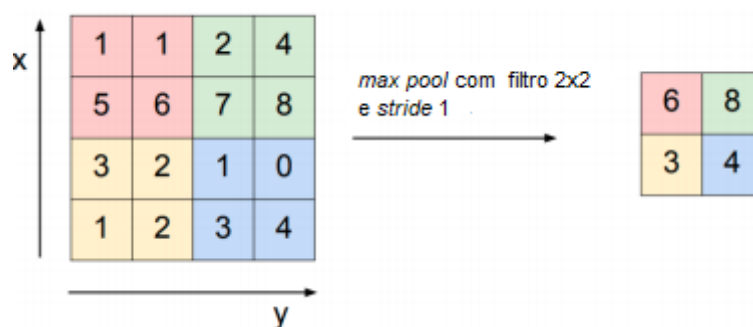


Figura 15: Pooling
Adaptado de: [Karpathy et al., 2016].

2.9 Camada totalmente conectada

As *Fully Connected Layers* são camadas de uma *Artificial Neural Network* - ANN Nielsen [2015] nas quais cada nó de uma camada está conectado a todos os outros nós da camada subsequente. A Figura 16 apresenta uma representação de *fully connected layers*.

As características extraídas anteriormente pelas diversas camadas de convolução e *pooling* servirão de entrada para as camadas totalmente conectadas que irão interpretar essas características de alto nível de abstração e realizar operações complexas, por exemplo, classificar objetos [Yosinski et al., 2014]. Após as operações nas camadas totalmente conectadas tem-se a camada de saída. O tamanho da camada de saída, ou seja, o número de neurônios, é igual ao número de classes no problema. Além disso, cada neurônio de saída apresentará a probabilidade de compatibilidade da entrada com a classe associada a ele via função de ativação de softmax.

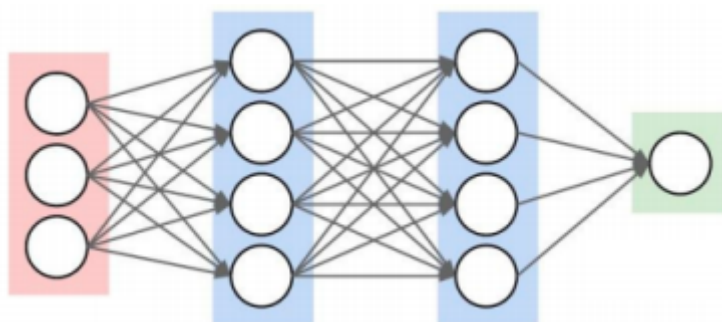


Figura 16: Fully-Connected
Adaptado de: [Karpathy et al., 2016].

2.10 Treinamento da CNN via *backpropagation*

O aprendizado de uma CNN é baseado no reajuste proporcional ao erro calculado na saída dos seus parâmetros livres, bias e pesos. Esta etapa de reajuste é responsável pelo treino da CNN, logo, ao final da execução do reajuste, a CNN é considerada treinada para

a entrada fornecida. Assim, é esperado que, após ser treinada para diferentes entradas, a CNN responda corretamente a uma entrada não treinada. Esse reajuste é, usualmente, realizado a partir do algoritmo de *backpropagation* [Lecun et al., 1998]. O algoritmo calcula o gradiente de uma função de custo para determinar o ajuste dos parâmetros da CNN a fim de minimizar os erros que afetam o desempenho da rede, ou seja, os valores de saída. Formalmente, seja θ o conjunto de todos os parâmetros livres da CNN (bias e pesos). O parâmetro ótimo, θ_{opt} , para uma tarefa específica pode ser obtida ao minimizar uma função de custo apropriada. Supondo um conjunto de treinamento de N pares entrada-saída $(x_k, y_{k=1}^N)$, onde, x_k é o k -ésimo valor de entrada e y_k é o k -ésimo valor de saída esperada e o_k é a saída gerada pela CNN para a entrada x_k , então uma função de custo a minimizar é a média das perdas sobre o conjunto de treinamento, ou seja:

$$L(\theta) = \frac{1}{N} \sum_{k=1}^N l(\theta, y_k, o_k) \quad (13)$$

Treinar uma CNN é um problema de otimização global, ou seja, problemas cuja função a ser otimizada possui mínimos locais na região de interesse. Ao minimizar a função de custo, é possível encontrar o conjunto de parâmetros ótimo. Ou seja:

$$\theta_{opt} = \min_{\theta \in A} \{L(\theta)\} \quad (14)$$

em que A é o espaço de parâmetros. A solução comumente utilizada para otimizar os parâmetros de uma rede CNN ao solucionar a expressão é o algoritmo Stochastic Gradient Descent (SGD) em Zinkevich et al. [2010].

2.11 Data Sets de falha de rolamento disponíveis para treinamento de ML e DL

Dados são fundamentais quando o assunto é arquiteturas de ML e DL. Segundo [Zoph et al., 2020], algoritmos de DL trabalham melhor quando treinadas com uma vasta amostra de dados. Em um data set com poucas amostras, geralmente a diversidade dos dados é limitada, o que pode influenciar na baixa eficiência da rede. Geralmente, quanto mais dados disponíveis, melhor a acurácia do modelo. As estruturas de dados com *labels* diferentes podem ser melhoradas usando variações abundantes, e o modelo reconhecerá as características invariantes em comparação com tais diferenças [Roh et al., 2019]. Os data set podem ser categorizados em dois grupos dependendo da complexidade: Data set simples e Data set Complexo. Um data set simples, tende a permitir a manipulação e cálculos de dados fáceis e eficazes para algum tipo de análise estatística significativa [Kitchin and McArdle, 2016]. Um conjunto de dados bem rotulado e balanceado que não contém os dados aberrantes ou valores ausentes também é considerado bom. Por outro lado, um conjunto complexo de dados pode ser definido como um grande conjunto de da-

dos, que possui grande volume, grande variedade, grande velocidade e grande veracidade [Zhang et al., 2018]. Os dados do mundo real são em sua maioria altamente distorcidos e aprender com esses conjuntos de dados é uma tarefa desafiadora para algoritmos de classificação padrão. Quando apresentados com conjuntos de dados desequilibrados complexos, esses algoritmos falham em representar as características distributivas dos dados com precisão e fornecem, de forma resultante, precisões desfavoráveis entre as classes dos dados. Embora seja bastante desafiador lidar com dados complexos ou desequilibrados, os dados desequilibrados são as características de várias aplicações do mundo real, como diagnóstico médico, detecção de fraude, detecção de falha de maquinário, catástrofe de preços [Douzas and Bacao, 2018]. O rolamento é um tópico amplamente aplicado para detecção de falhas de máquinas ou detecção de anomalias. A razão pode ser o conjunto de dados público prontamente disponível [Zheng et al., 2019]. Alguns dos conjuntos de dados de rolamento de código aberto acessíveis usados para a detecção e diagnóstico de falha de rolamento são os seguintes:

- CWRU Data set
- Paderborn University Data set
- FEMTO Data set
- IMS Data set

2.11.1 CWRU Data set

O conjunto de dados CWRU é um conjunto de dados popular, de código aberto e facilmente acessível. O conjunto de dados gerado é registrado e disponível no site da CWRU, que fornece acesso aos dados do rolamento para rolamentos normais e com defeito. Neste banco de dados, os dados foram coletados para rolamentos normais, defeitos no rolamento no eixo de acionamento (DE) e ponta da ventoinha (FE). O conjunto de dados de rolamento CWRU serve como referência padrão [Smith and Randall, 2015] para autenticar o desempenho de diferentes algoritmos de ML e DL. O arranjo da plataforma de teste de rolamento usado para obter o conjunto de dados CWRU é mostrado na figura 5, que consiste em um motor elétrico de indução Reliance de 2 HP, um transdutor de torque, um dinamômetro e componentes eletrônicos de controle, que não são mostrados na figura. Os rolamentos de teste suportam o eixo do motor. O torque é aplicado ao eixo por meio de um dinamômetro e sistema de controle eletrônico. As falhas foram semeadas nos seguintes componentes do rolamento, pista interna (IR) e pista externa (OR), e cada rolamento defeituoso foi reinstalado na plataforma de teste. A usinagem por eletrodescarga (EDM) foi usada para introduzir as falhas de ponto único nos rolamentos de teste com diâmetros de falha de 0,18, 0,36, 0,54 e 0,71 mm. Os dados de aceleração foram medidos em locais próximos e longe dos rolamentos do motor. Os dados são coletados

de vários sensores colocados em locais diferentes. Os acelerômetros, que foram fixados ao alojamento com bases magnéticas e colocados na posição de 12 horas em ambos DE e FE do rolamento do motor, foram usados para coletar dados de vibração. Além disso, para alguns experimentos, um acelerômetro foi anexado à placa de base de suporte do motor também. Uma vez que os dados foram coletados, eles foram processados no ambiente *MATLAB*, e todos os arquivos de dados foram armazenados no formato *MATLAB* (.mat). Cada arquivo contém um ou mais dos dados registrados de DE, FE e placa de base (BA). As frequências de amostragem de 12 kHz e 48 kHz foram utilizadas para a coleta de dados. Para os experimentos de mancal de acionamento, os dados foram coletados em 12kHz e 48kHz amostras por segundo. Os dados do rolamento próximo a ventoinha foram coletados em 12kHz amostras por segundo. Para os dados coletados da base, a taxa de coleta de dados foi de 48kHz amostras por segundo. O conjunto de dados consiste em arquivos de dados para diferentes cargas de torque aplicado pelo dinamômetro, a potência e velocidade, do motor varia de 0 a 3 hp e 1730 a 1797 rpm conforme a tabela 2 abaixo.

No.	Velocidade de rotação (rpm)	Potência do motor (hp)
0	1797	0
1	1772	1
2	1750	2
3	1730	3

Tabela 2: Parâmetros de operação CWRU

Portanto, o conjunto de dados consiste em 161 registros, que são agrupados em quatro classes: dados da base 48kHz, falha de extremidade do eixo de acionamento 48kHz, falha da extremidade do eixo de acionamento 12kHz e falha da extremidade do ventilador 12kHz. Cada grupo, novamente, consiste em conjuntos de dados para falha do rolamento de esferas, falha da pista interna e falhas da pista externa. De acordo com a localização da falha em relação à zona de carga, as falhas da pista externa são ainda classificadas em três categorias: 'centrado' (falha na posição 6,00 horas), 'ortogonal' (3,00 horas) e 'oposto' (12h00). Nesse trabalho a conversão do sinal para imagem foi realizado a partir dos dados de falha de extremidade do eixo de acionamento de 12kHz e utilizando as falhas de pista externa centrado, pois segundo Wen et al. [2018], devido ao fato de apresentar sinais com pouco ruído, esses dados proporcionam acurácias melhores.

2.11.2 Data set da Universidade Padderborn

O conjunto de dados de Lessmeier et al. [2016], também são para diagnóstico de falha de rolamento e está disponível no data center KAT na Padderborn University. Os componentes essenciais do equipamento de teste são um motor de acionamento, um eixo de medição de torque, um módulo de teste e um motor de carga. O conjunto de dados de rolamentos da Padderborn University consiste em dados de vibração de alta resolução,

que são coletados de experimentos realizados em seis rolamentos saudáveis e 26 conjuntos de rolamentos danificados. Dos 26 conjuntos de rolamentos danificados, 12 foram danificados artificialmente e 14 foram danificados usando testes de vida acelerada para simular danos reais. Ele fornece a base para o desenvolvimento, validação e treinamento dos algoritmos de diagnóstico para monitoramento de condições de rolamentos. A bancada de teste foi operada em diferentes condições operacionais para garantir a robustez dos métodos em diferentes condições operacionais. No total, foram realizados experimentos com 32 danos diferentes em rolamentos de esferas, do tipo 6203. Este conjunto de dados consiste nas correntes do motor e sinais de vibração medidos de forma síncrona, que permitem testes mais precisos e implementação dos algoritmos ML em aplicações práticas, onde os defeitos reais são gerados pelo envelhecimento e pela perda gradual de lubrificação [Zhang et al., 2020]. Pandhare et al. [2019], Chen et al. [2018], Zhu et al. [2019], Wen [2020] usam este conjunto de dados em suas pesquisas.

2.11.2.1 Danos artificiais

O uso de danos artificiais para o desenvolvimento de métodos de classificação de falha é comum, como reportado em artigos [Lessmeier et al., 2016]. Fazer cavidades na pista externa é um dos principais danos usados para estudar falhas de rolamento, como em Zarei and Poshtan [2009], Lessmeier et al. [2016].

Os métodos mais utilizados para gerar danos artificiais são:

- Usinagem por Eletro-Descarga (EDM).
- Fazer buracos nas pistas do rolamento.

O data set Padderborn, realizou testes com os dois principais danos encontrados em referência, além de realizar *electrical engrave*. Segundo Lessmeier et al. [2016], os primeiros dois métodos são muito precisos e fáceis de identificar. Portanto, esses danos artificiais são apropriados para realizar comparação entre resultados com outros estudos. No entanto, existe uma falta de correlação com danos reais, porque geralmente danos reais possuem, uma transição muito abrupta e acentuada entre o dano e as áreas da pista não danificadas [Lessmeier et al., 2016]. Para examinar essa correlação, o terceiro tipo de dano artificial causado de forma manual com um gravador elétrico, a qual irá gerar uma superfície irregular e não profunda, simulando dano de corrosão. Na figura 17, temos os três tipos de danos artificiais realizados nesse data set.

Detalhes sobre o componente de falha e o tipo de dano artificial que o data set Padderborn dispõem, na tabela 3



Figura 17: Danos Artificiais
Adaptado de: [C.Lessmeier et al., 2016].

Código do Rolamento	Componente	Método do dano
KA01	OR	EDM
KA03	OR	GRAVADOR MANUAL
KA05	OR	GRAVADOR MANUAL
KA06	OR	GRAVADOR MANUAL
KA07	OR	PERFURAÇÃO
KA08	OR	PERFURAÇÃO
KA09	OR	PERFURAÇÃO
KI01	IR	EDM
KI03	IR	GRAVADOR MANUAL
KI05	IR	GRAVADOR MANUAL
KI07	IR	GRAVADOR MANUAL
KI08	IR	GRAVADOR MANUAL
OR: Pista externa; IR: Pista interna;		

Tabela 3: Danos Artificiais

2.11.2.2 Danos reais

Existem duas opções para medir danos de rolamentos reais: Medir os danos de rolamento em situações reais de operação ou em bancadas de teste científicas.

O sucesso do uso de dados da indústria é difícil, é complicado para capturar e comparar dados para treino de diferentes tipos de danos. Isso ocorre devido a longo período de vida dos rolamentos, e quando o dano é reconhecido, os rolamentos são trocados antes da falha ocorrer, o que dificulta para capturar dados de falha de rolamento reais. Para reduzir as influências externas, as bancadas de teste são usadas, o qual possibilita a geração de danos realísticos, figura 18, de rolamento [Nectoux et al., 2012].

A bancada de teste, figura 19, para acelerar o tempo de vida de um rolamento, consiste no mancal de rolamento, motor elétrico, que alimenta um eixo com quatro rolamentos de teste do tipo 6203. Os rolamentos de teste giram sob uma carga radial que é aplicada por um mecanismo de parafuso de mola. A força radial aplicada é maior do que em aplicações normais de rolamentos para acelerar o aparecimento de danos por fadiga, mas ainda baixa o suficiente para não exceder a capacidade de carga estática do rolamento.

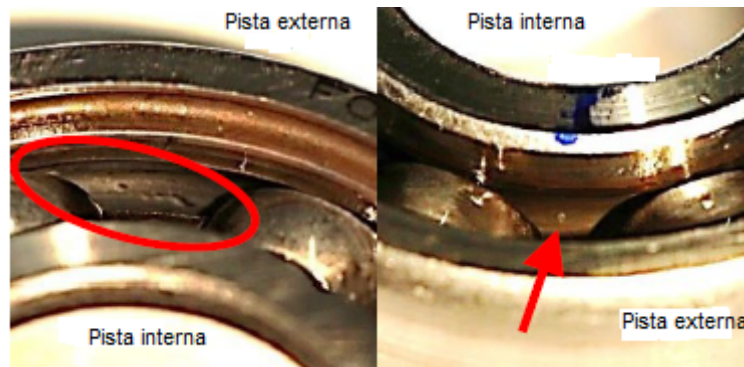


Figura 18: Danos de rolamento gerados pela bancada de teste de vida acelerado.
Adaptado de: [C.Lessmeier et al., 2016].

Além disso, foi utilizado óleo de baixa viscosidade, o que leva a condições de lubrificação inadequadas e favorece o aparecimento de danos. Em torno de 70% dos danos ocorreram devido a fadiga, geraram corrosão na superfície do rolamento. O restante dos rolamentos, foram danificados por deformação na superfície do rolamento, em forma de elevações ou recuos nas pistas do rolamento. Os danos de corrosão ocorrerem tanto na pista interna quanto na pista externa dos rolamentos. Deformação foram encontrados apenas nas pistas externas. Enquanto danos nas esferas do rolamento não foram observadas durante os testes [Lessmeier et al., 2016].

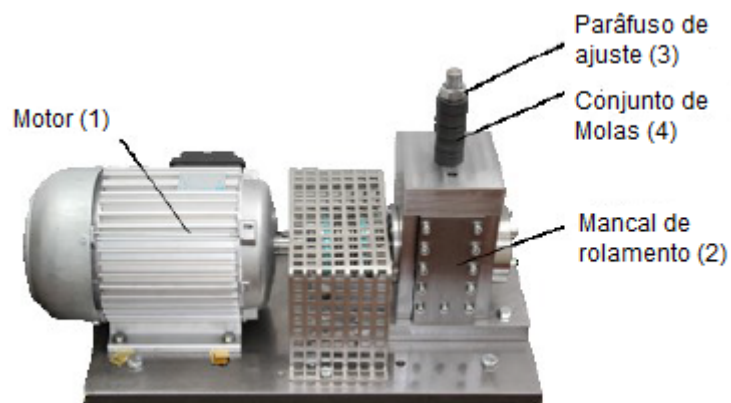


Figura 19: Bancada de testes da danos reais
Adaptado de: [C.Lessmeier et al., 2016].

Para gerar os resultados foram utilizados quatro condições de operação, conforme a tabela 4. As velocidades de rotação testadas foram de 900 e 1500 rpm, o torque de carga foram de 0.1 e 0.7 Nm e a força radial foi de 400 e 1000 N.

No.	Velocidade de rotação (rpm)	Torque da carga(Nm)	Força Radial(N)
0	1500	0.7	1000
1	900	0.7	1000
2	1500	0.1	1000
3	1500	0.7	400

Tabela 4: Parâmetros de operação

Detalhes sobre o código do rolamento, componente de falha e o método de dano que o data set Padderborn dispõem, na tabela 5.

Código do Rolamento	Componente	Método do dano
KA04	OR	Corrosão
KA15	OR	Deformação na superfície
KA16	OR	Corrosão
KA22	OR	Corrosão
KA30	OR	Deformação na superfície
KB23	IR (+OR)	Corrosão
KB24	IR (+OR)	Corrosão
KB27	OR + IR	Deformação na superfície
KI04	IR	Corrosão
KI14	IR	Corrosão
KI16	IR	Corrosão
KI17	IR	Corrosão
KI18	IR	Corrosão
KI21	IR	Corrosão
OR: Pista externa; IR: Pista interna		

Tabela 5: Tabela com dados naturais

3 METODOLOGIA

Nesse capítulo será abordado questões sobre os softwares e o hardware necessário para desenvolver esse trabalho. Além disso será detalhado a conversão do sinal de vibração em imagem e a estrutura para implementação da CNN proposta.

As etapas de treinamento e teste da arquitetura proposta foram realizadas utilizando Keras com *backend* baseado em Tensorflow, software de código aberto desenvolvido pela *Google brain Team*, destinado à programação numérica utilizando programação baseada em fluxo de dados em grafos [Abadi et al., 2016]. Apesar de ser utilizado para outros propósitos, devido ao amplo suporte à utilização de GPUs, o Tensorflow é frequentemente utilizado em problemas de ML e DL para acelerar o treinamento de redes neurais. O hardware necessário para o desenvolvimento do trabalho se restringe a, no mínimo, um computador com capacidade de processamento suficiente para o treinamento da arquitetura de CNN que foi implementada. Desta forma, o trabalho foi realizado, principalmente, no computador pessoal do autor. Visando acelerar o processo de treino e teste das redes implementadas, além do computador pessoal do autor, foram utilizados recursos do *google colabory*. *Google Colabory* é baseado no *Jupyter Notebook*. *Jupyter* é uma ferramenta *open-source*, o qual interpreta e integra linguagens, bibliotecas e ferramentas de visualização [Perez and Granger, 2007]. *Google Colabory* é um projeto que tem o objetivo de disseminar pesquisas e estudos que utilizam *machine learning* [Carneiro et al., 2018]. *Colabory* dispõem de um ambiente que provém de Python 2 e 3, além de possui bibliotecas para *machine learning* e *artificial intelligence* como, *TensorFlow*, *Matplotlib* e *Keras*, ferramentas as quais foram utilizadas nesse projeto. Portanto, o *Google* disponibiliza um ambiente para aceleração de treinamento de modelos utilizando GPU. Contudo, não se pode escolher qual a GPU se quer trabalhar, mas segundo Carneiro et al. [2018], o ambiente apresenta as seguintes configurações:

CONFIGURAÇÃO 1: (i) sistema operacional Windows 10; (ii) processador Intel Xeon com dois *cores* @2.3 GHz; (iii) memória RAM de 12 GB RAM; (iv) unidade de armazenamento de 25GB (disco rígido); (v) placa de vídeo NVIDIA Tesla T4, com 16 GB de memória dedicada.

CONFIGURAÇÃO 2: (i) sistema operacional Windows 10; (ii) processador Intel

Xeon com dois *cores* @2.3 GHz; (iii) memória RAM de 12 GB RAM; (iv) unidade de armazenamento de 25GB (disco rígido); (v) placa de vídeo NVIDIA Tesla K80, com 12 GB de memória dedicada.

3.1 Conversão do sinal 1D para uma imagem 2D

Em métodos tradicionais de ML baseado em dados, um pré-processamento dos dados é vital, pois a maioria dos métodos não utiliza o sinal bruto diretamente como entrada. Uma das principais características de um método de pré-processamento de dados é extrair atributos de identificação a partir do sinal bruto analisado. Entretanto, a extração de características é um trabalho exaustivo e vale ressaltar que os resultados finais variam linearmente com as características extraídas do sinal bruto analisado. Portanto, essa conversão propõem em converter o um sinal no domínio do tempo em imagens[Wen et al., 2018].

Como mostra a figura 20, nesse método de conversão, o sinal no domínio do tempo preenche os pixels da imagem em sequência. Para obter uma imagem $M \times M$, se deve obter randomicamente um sinal secundário do tamanho M^2 , a partir do sinal bruto analisado.

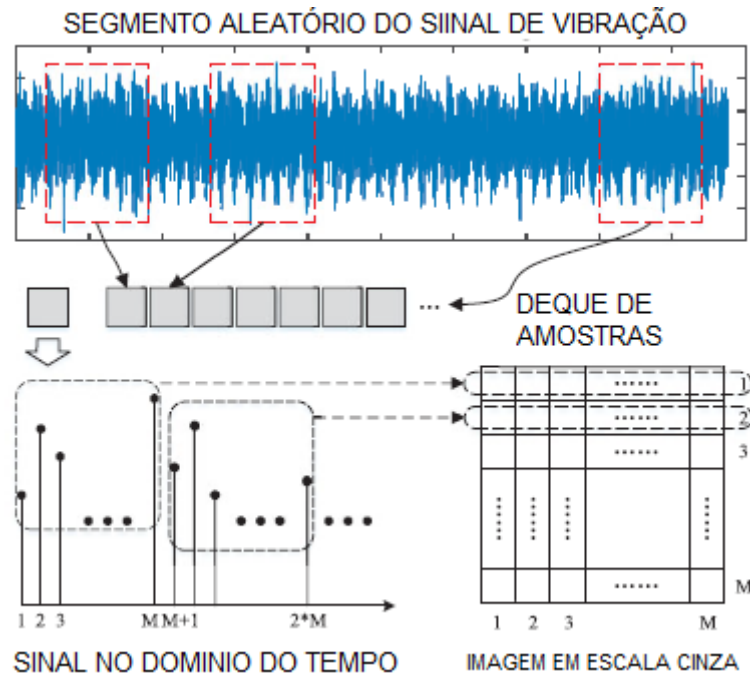


Figura 20: Equação de Transformação
Adaptado de: [L.Wen et al., 2018].

Então $L(i)$, $i=1, \dots, M^2$, são os valores do sinal secundário. $P(j,k)$, $j=1, \dots, M$, $k=1, \dots, M$, demonstra a intensidade do pixel da imagem, como mostrado na equação 15:

$$P(j, k) = \text{round} \left(\frac{L((j-1) \times M + k) - \text{Min}(L)}{\text{Max}(L) - \text{Min}(L)} * 255 \right) \quad (15)$$

3.2 Estrutura CNN proposta

Uma vez que os sinais de vibração foram convertidos para imagem, se pode treinar a CNN para classificar imagens. LeNet-5 é uma clássica CNN, conhecida por ter bons resultados em reconhecer padrões em imagens [Lecun et al., 1998, Zhang et al., 2020]. Nesse estudo, um modelo baseado em LeNet-5 foi desenvolvido para resolver o problema de classificação de imagens para diagnóstico de falha. O tamanho da imagem de entrada em LeNet-5 é 32x32, mas afim de melhorar os resultados, o autor propõem que se altere o tamanho das imagens conforme o data set disponível. Para esse estudo as imagens são do tamanho de 64x64 pixels. O modelo CNN proposto, possui quatro camadas *convolutional* e *pooling* com duas camadas *fully-connected*, FC1 e FC2. *Zero Padding* é utilizado nesse estudo. A figura 21 mostra a estrutura da CNN proposta para imagens com 64x64 pixels. É importante ressaltar que para esse estudo, os valores de *stride* e *padding* foram definidos como 1.

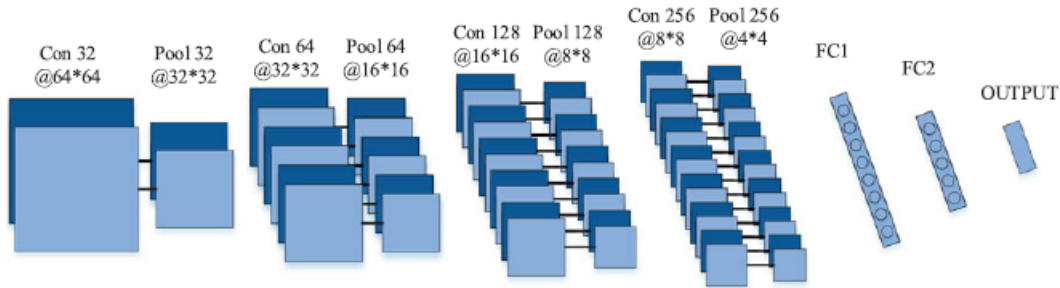


Figura 21: CNN Proposta
Adaptado de: [S.Zhang et al., 2016].

4 RESULTADOS

Nesse capítulo, a CNN proposta é aplicada em dois famosos datasets de diagnóstico de falhas para rolamentos de motores, o CWRU e Padderborn. Vale ressaltar que o dataset Padderborn, contempla dados de falhas baseado em equipamentos em operação.

4.1 CNN aplicada ao DataSet CWRU

Os sinais de vibração de rolamento no eixo de acionamento da carga do motor, são coletados sobre quatro condições (0,1,2 e 3 hp) para verificar a performance do modelo proposto. São 20000 amostras por condição para o dataset de treinamento e 4000 amostras por condição para o dataset de teste, totalizando 80000 amostras para treino e 16000 amostras para teste, conforme a tabela 6. Todas as amostras são randomicamente selecionadas, portanto vale ressaltar que as amostras do dataset de treinamento e do dataset de teste são diferentes.

Condições carga do motor	Treino	Teste
0 hp	20000 amostras	4000 amostras
1 hp	20000 amostras	4000 amostras
2 hp	20000 amostras	4000 amostras
3 hp	20000 amostras	4000 amostras
TOTAL DE AMOSTRAS POR CONDICA0(0,1,2,3 hp)	80000 amostras	16000 amostras

Tabela 6: Tabela com a quantidade de imagens utilizadas para treinar e testar o modelo proposto

4.2 Resultado da conversão do sinal

Nesse dataset são dez condições analisadas, nove são condições de falha e uma é a condição normal(10 classes). Os três tipos de falha de rolamento nesse dataset são, falha na esfera do rolamento, falha na pista interna do rolamento e falha na pista externa do rolamento. O tamanho dos danos são 0.18,0.36 e 0.54mm. O tamanho da imagem convertida é de 64 x 64 pixels. O resultado da conversão das condições é apresentado na tabela 7. A imagem em escala cinza contém 4096 pixels. A partir das imagens convertidas, se pode notar que as imagens são diferentes uma das outras.

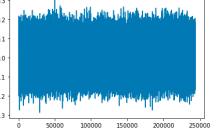
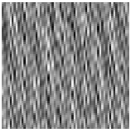
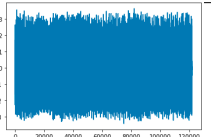
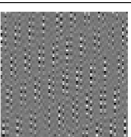
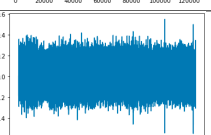
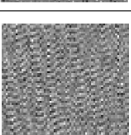
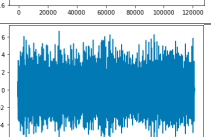
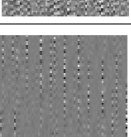
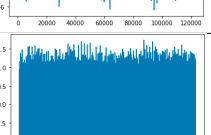
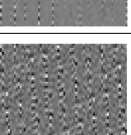
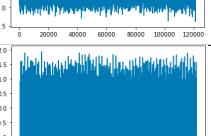
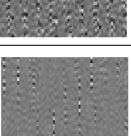
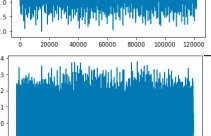
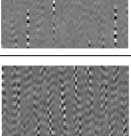
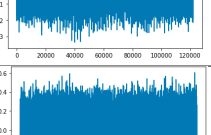
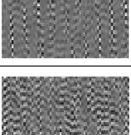
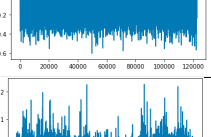
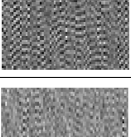
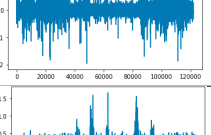
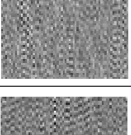
Componente	Sinal de vibracao no dominio do tempo	Imagem Gerada	Profundidade de falha	Treino	Teste
Normal			NA	2000 amostras	400 amostras
OR			0.18 mm	2000 amostras	400 amostras
OR			0.36 mm	2000 amostras	400 amostras
OR			0.54 mm	2000 amostras	400 amostras
IR			0.18 mm	2000 amostras	400 amostras
IR			0.36 mm	2000 amostras	400 amostras
IR			0.54 mm	2000 amostras	400 amostras
Esfera			0.18 mm	2000 amostras	400 amostras
Esfera			0.36 mm	2000 amostras	400 amostras
Esfera			0.54 mm	2000 amostras	400 amostras
TOTAL DE AMOSTRAS POR CONDICAO(0,1,2,3 hp)				20000 amostras	4000 amostras

Tabela 7: Tabela comparativa dos sinais de vibração e das imagens geradas

4.3 Resultado da CNN proposta

Os parâmetros de cada camada da estrutura da CNN, é apresentado na tabela 8. A notação Conv(5 x 5 x 32) mostra os valores da camada de *convolution*, o tamanho do filtro é 5 x 5 com 32 canais. Maxpool(2 x 2) mostra os valores da camada maxpool com filtro 2 x 2. FC mostra os valores de neurônios nas camadas totalmente conectadas.

Nome da camada	Configuração das camadas	Parâmetros Treináveis
L1	Conv (5 x 5 x 32)	832 parâmetros
L2	Maxpool (2 x 2)	0 parâmetros
L3	Conv (3 x 3 x 64)	18.496 parâmetros
L4	Maxpool(2 x 2)	0 parâmetros
L5	Conv (3 x 3 x 128)	73.856 parâmetros
L6	Maxpool (2 x 2)	0 parâmetros
L7	Conv (3 x 3 x 256)	295.168 parâmetros
L8	Maxpool(2 x 2)	0 parâmetros
L9	FC1 (2056)	10.488.320 parâmetros
L10	FC2(256)	655.616 parâmetros
L11	FC3(10)	2570 parâmetros
TOTAL DE PARÂMETROS TREINADOS		11.534.858 parâmetros

Tabela 8: Parâmetros de cada camada da CNN

Foram testados 7 configurações para a CNN, como se pode ver na tabela 9, a acurácia dos modelos estão bem próximas, os pontos que se alteram são as últimas duas camadas totalmente conectadas, a FC1 e a FC2. As acurácias médias dos modelos configurados com, CNN-2560, CNN-2560-128, e CNN-2560-512 são ligeiramente inferiores ao modelo CNN-2560-1024. CNN-2560-64, CNN-2560-256, e CNN-2560-768 possuem as melhores acurácias. O melhor modelo foi o CNN-2560-256, sua acurácia média é de 99,95%, o valor máximo foi de 100% e o valor mínimo foi de 99,87%.

No.	CNN-2560	CNN-2560-64	CNN-2560-128	CNN-2560-256	CNN-2560-512	CNN-2560-768	CNN-2560-1024
Max	100	100	100	100	99,97	100	100
Min	99,65	99,84	99,52	99,87	99,62	99,82	99,80
Média	99,87	99,92	99,88	99,95	99,90	99,93	99,91
Desvio Padrão	0,1266	0,0585	0,1489	0,0380	0,1130	0,0726	0,0772

Tabela 9: Resultados da CNN proposta

A figura 22 apresenta a matriz de confusão do modelo de melhor resultado, CNN-2560-256. As linhas representam o label verdadeiro da imagem sendo analisada, a coluna representa a predição do label para cada condição. Após a análise da matriz, se pode notar que todas as condições atingiram 100% de acurácia. O modelo apresentou uma acurácia de predição média de 99,95% (O valor máximo da CNN-2560-256 pode ser visto na tabela 9). O objetivo desse resultado é mostrar que o algoritmo se adapta quando treinado e testado com dados de danos artificiais, conforme proposto em Wen et al. [2018]. Na próxima seção o modelo é analisado quando aplicado a dados de danos reais.

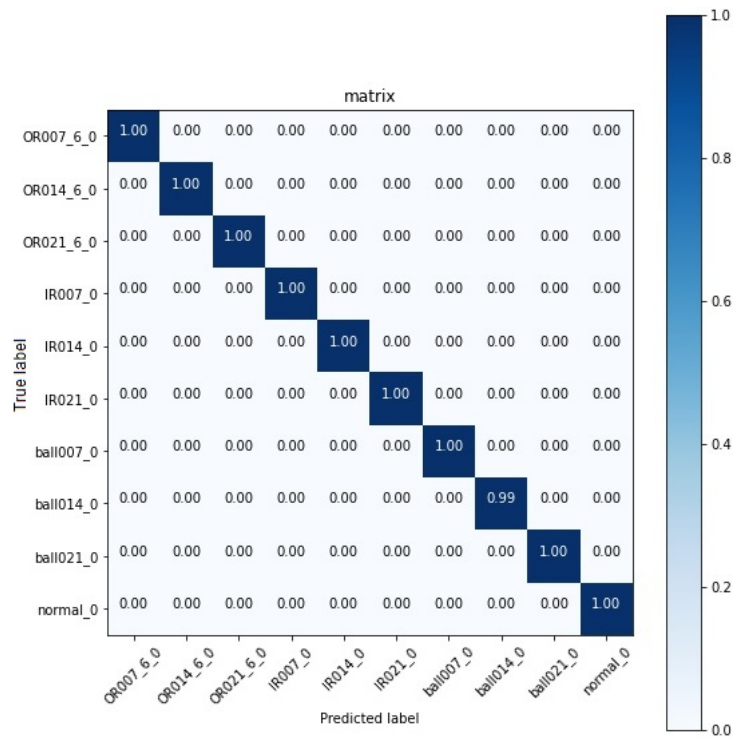


Figura 22: Matriz de confusão

4.4 CNN aplicada ao DataSet Padderborn

Nesse dataset o objetivo é utilizar a combinação de rolamentos saudáveis e com danos artificiais, para identificar rolamentos com danos reais e classificá-las em normal (classe 1), dano na pista interna (classe 2) ou dano na pista externa (classe 3). Conforme a tabela 10, foram gerados 2000 imagens para cada rolamento de cada classe para gerar o dataset de treino, com danos artificiais, e 400 imagens por rolamento de cada classe, para gerar o dataset de teste com danos reais. Portanto foram gerados 14000 amostras de treino e 4000 amostras para teste, para cada condição de operação tabela 4, gerando 56000 imagens de treino e 16000 imagens para teste.

Classe		Treino	Teste
1	Normal	K002	K001
2	DANOS PISTA EXTERNA OR		KA22
		KA01	KA04
		KA05	KA15
		KA07	KA30
3	DANOS PISTA INTERNA IR		KA16
			KI14
		KI01	KI21
		KI05	KI17
		KI07	KI18

Tabela 10: Tabela com os dados de treino, danos artificiais, e dados de teste, danos reais

4.5 Resultado da conversão do sinal Padderborn

O tamanho da imagem convertida é de 64 x 64 pixels. O resultado das conversões das principais condições de falha, são apresentados na tabela 11. A tabela, apresenta o código do rolamento, o componente que foi danificado, qual foi o modo de falha, a aplicação dos dados para a rede neural, o espectro de frequência do sinal analisado, e a representação de imagem em escala cinza. O data set de teste é composto com dados de danos causados por lubrificação inadequada e por fadiga, o que geram corrosão e algumas deformações na superfície das pistas do rolamento. Enquanto o data set de treino é composto pelos três principais danos artificiais já citados, EDM, Gravação manual e Perfuração. A imagem em escala cinza contém 4096 pixels. A partir das imagens convertidas, se pode notar que as imagens são distintas uma das outras.

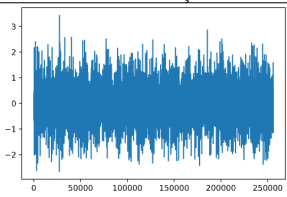
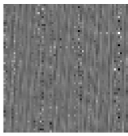
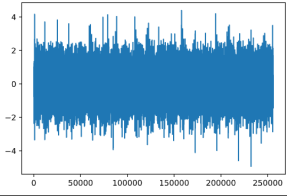
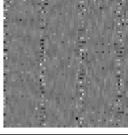
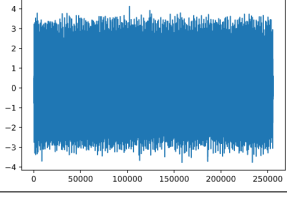
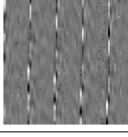
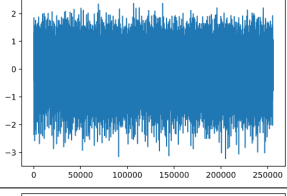

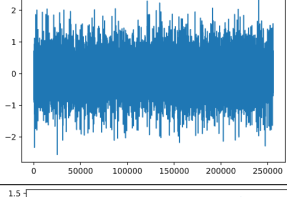
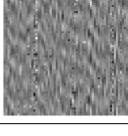
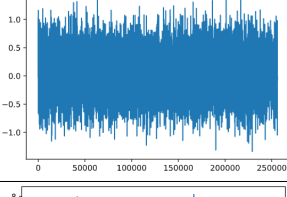
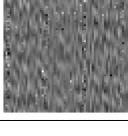
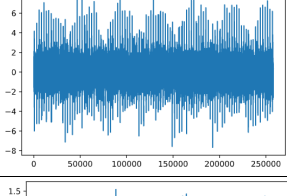
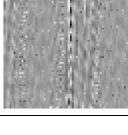
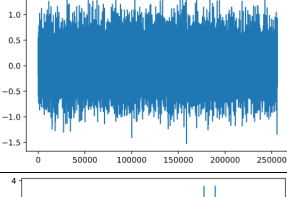
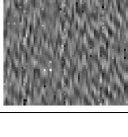
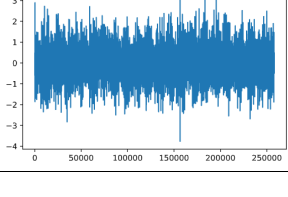

Código do Rolamento	Componente	Modo de falha	Aplicação	Sinal de vibração	Representação em imagem
K002	Normal	NA	Treino		
K001	Normal	NA	Teste		
KA01	OR	EDM	Treino		
KA04	OR	Corrosão	Teste		
KA05	OR	Gravação Elétrica	Treino		
KA15	OR	Deformação na superfície	Teste		
KI01	IR	EDM	Treino		
KI14	IR	Corrosão	Teste		
KI05	IR	Gravação Elétrica	Treino		

Tabela 11: Tabela com os dados de treino, danos artificiais, e dados de teste, danos reais, referentes ao Padderborn

4.6 Resultado da CNN proposta

Os parâmetros de cada camada da estrutura da CNN é a mesma apresentado na tabela 12, exceto pela diferença nas camadas FC, que possuem menos neurônios para compilar as características aprendidas nas camadas anteriores, o que foi observado como positivo, conforme se pode analisar na tabela 13, onde demonstra uma tendência a ter uma melhor acurácia, pois segundo Chen et al. [2018], alguns problemas de *overfitting* ocorrem devido ao fato dos neurônios estarem retendo características que confundem a CNN quando testadas com padrões de dados diferentes do treino.”

Nome da camada	Configuração das camadas	Parâmetros Treináveis
L1	Conv (5 x 5 x 32)	832 parâmetros
L2	Maxpool (2 x 2)	0 parâmetros
L3	Conv (3 x 3 x 64)	18.496 parâmetros
L4	Maxpool(2 x 2)	0 parâmetros
L5	Conv (3 x 3 x 128)	73.856 parâmetros
L6	Maxpool (2 x 2)	0 parâmetros
L7	Conv (3 x 3 x 256)	295.168 parâmetros
L8	Maxpool(2 x 2)	0 parâmetros
L9	FC1 (1024)	4.195.328 parâmetros
L10	FC2(3)	655.616 parâmetros
TOTAL DE PARÂMETROS TREINADOS		4.586.755 parâmetros

Tabela 12: Parâmetros de cada camada da CNN aplicada ao Padderborn

Foram estudados 6 configurações de modelos CNN, tabela 13, como se pode ver na tabela de resultados, a acurácia média começa a decair conforme se aumenta o número de neurônios nas camadas FC1 e FC2. Os modelos CNN-1024-512 e 1024-768, apresentaram as piores predições. Os modelos CNN-1024 e CNN-1024-256 são ligeiramente superiores a CNN-1024-64 e a CNN-1024-128. CNN-1024 e CNN-1024-256 possuem as melhores acurácias. O melhor modelo foi o CNN-1024, sua acurácia média é de 83,54%, o valor máximo foi de 89,14% e o valor mínimo foi de 77,74%.

No.	CNN-1024	CNN-1024-64	CNN-1024-128	CNN-1024-256	CNN-1024-512	CNN-1024-768
Max	89,14	88,17	87,77	86,62	86,19	88,22
Min	77,74	73,92	65,37	79,32	67,12	67,82
Média	83,54	82,40	82,24	83,07	79,06	80,57
Desvio Padrão	3,8615	4,8320	6,7094	2,5471	7,4632	6,3271

Tabela 13: Resultados da CNN aplicada ao data set Padderborn

A figura 23 apresenta a matriz de confusão do modelo de melhor resultado, CNN-1024. As linhas representam o label verdadeiro da imagem sendo analisada, a coluna representa a predição do label para cada condição. O modelo apresentou uma acurácia de predição média de 83,54% (O valor máximo da CNN-1024 pode ser visto na tabela 13). Após a análise da matriz, se pode notar que a condição com as melhores acurácia são da classe 1 e classe 2. A classe 2 e a classe 3 ainda estão apresentando dificuldades de classificação.

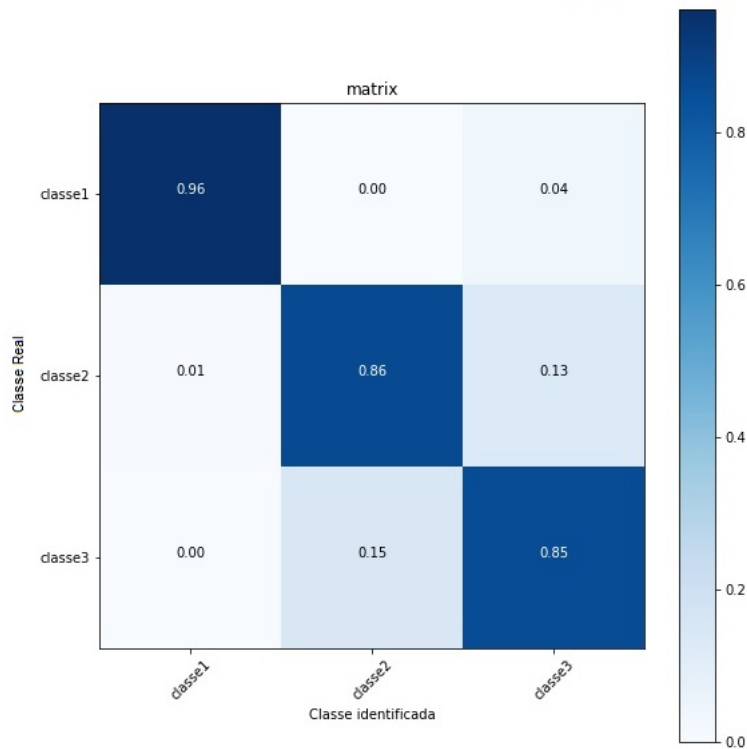


Figura 23: Matriz de confusão Padderborn

No trabalho de Lessmeier et al. [2016], foi realizado um estudo com os seguintes métodos tradicionais de aprendizado de máquina, como RF, SVM, kNN entre outros. A maior acurácia baseado nesses modelos, foi a obtida pelo algoritmo *Ensemble*, que atingiu 75% de acurácia, conforme a tabela 14. Portanto fica evidente que o modelo CNN proposto atingiu uma acurácia aceitável, apesar de apresentar dificuldades na classificação dos defeitos de IR e OR.

Algoritmo	Acurácia (%)
kNN	63,2
RF	64,1
SVM	65,5
<i>Ensemble</i>	75
CNN	83

Tabela 14: Comparação dos algoritmos aplicados ao dataset Padderborn

5 CONCLUSÃO

O projeto objetivou-se em aplicar um modelo CNN para diagnóstico de falhas, e estudar o seu comportamento quando aplicado a dados de danos reais para classificação de falhas de rolamento.

Para alcançar este objetivo algumas etapas foram realizadas, primeiro foi necessário estudar os principais algoritmos de inteligência artificial aplicado a diagnóstico de falha. Com isso, se chegou a conclusão que os algoritmos de *deep learning* se destacaram nos últimos cinco anos nessa área, e portanto foi escolhido o modelo proposto por Long Wen para realizar esse trabalho. O modelo foi escolhido por apresentar os melhores resultados para extração de características da imagem, diminuindo assim a necessidade de conhecimento sobre a extração de características de sinais de vibração antes de classificá-los.

Para validar que o modelo proposto nesse trabalho estava correto, foi necessário replicar o trabalho de Long Wen e avaliar os resultados gerados, esperando uma acurácia de 97.55%. O resultado obtido nessa primeira parte do trabalho foi de 99%. Após o modelo proposto, ter sido replicado, foi necessário aplicar o novo modelo aos dados de danos reais em falha de rolamentos. Os resultados obtidos de acurácia média para aplicação da estrutura proposta para falhas de rolamentos com danos reais, foi 83%. Uma vez que a estrutura era conhecida por não aprender novas imagens a partir do seu treinamento, foram necessárias pequenas alterações em sua estrutura, logo o resultado foi considerado aceitável, apesar da rede neural confundir algumas classificações.

Como trabalhos futuros, almejo melhorar a acurácia do modelo proposto aplicando métodos de processamento de imagens, como nivelamento de histogramas e utilizar a técnica de validação cruzada para avaliar a habilidade de adaptação da CNN quando aplicada a novos dados. Além disso, outro questionamento levantado ao longo do trabalho, foi de que ainda existem poucos data sets disponíveis com danos reais de rolamento, e para uma aplicação de diagnóstico de falhas online, ter um modelo já robusto se faz necessário. Portanto, a criação de uma bancada de testes para gerar danos reais de rolamento ainda se faz necessário, principalmente porque para uma empresa gerar dados em equipamentos em funcionamento irá gerar custos altos de manutenção, o que dificulta a aquisição de dados em equipamentos em operação de chão de fábrica.

REFERÊNCIAS

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

B.. Al-Najjar. *Condition Based Maintenance: Selection and Improvement of a Cost effective Vibration Based Policy for Rolling Element Bearing*. Doctoral Thesis, Lund University, Sweden, 1997.

B.. Al-Najjar and A.. Ingwald. *Identification, analysis, elimination and prevention of recurrence of problems: methods and concepts*.

Basim. Al-Najjar, Hatem. Algabroun, and Mikael Jonsson. *Maintenance 4.0 fulfil the demands of Industry 4.0 and Factory of the Future*. Journal of Engineering Research and Application, 8^a edition, 2018. ISSN 2248-9622.

Adel Boudiaf, Abdelkrim Moussaoui, Amin Dahane, and Issam Attoui. A comparative study of various methods of bearing faults diagnosis using the case western reserve university data. *Journal of Failure Analysis and Prevention*, 16, 02 2016. doi: 10.1007/s11668-016-0080-7.

Y-Lan Boureau, J. Ponce, and Yann Lecun. A theoretical analysis of feature pooling in visual recognition. pages 111–118, 11 2010.

T. Carneiro, R. V. Medeiros Da Nóbrega, T. Nepomuceno, G. Bian, V. H. C. De Albuquerque, and P. P. R. Filho. Performance analysis of google colab as a tool for accelerating deep learning applications. *IEEE Access*, 6:61677–61685, 2018. doi: 10.1109/ACCESS.2018.2874767.

F.T.S.. Chan and A.. Prakash. *Maintenance policy selections in manufacturing firm using the fuzzy MCDM approach*.

Yuanhang Chen, Gaoliang Peng, Chaohao Xie, Wei Zhang, Chuanhao Li, and Shaohui Liu. Acдин: Bridging the gap between artificial and real bearing damages for bearing fault diagnosis. *Neurocomputing*, 294:61 – 71, 2018. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2018.03.014>. URL <http://www.sciencedirect.com/science/article/pii/S092523121830300X>.

G. Cordeiro. Etapas para implantacao da industria 4.0: Uma visao sob aspectos estrategicos e operacionais. *ENEGEP ENCONTRO NACIONAL DE ENGENHARIA DE PRODUCAO*, XXXVII, 2017.

Georgios Douzas and Fernando Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with Applications*, 91:464 – 471, 2018. ISSN 0957-4174. doi: <https://doi.org/10.1016/j.eswa.2017.09.030>. URL <http://www.sciencedirect.com/science/article/pii/S0957417417306346>.

David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002. ISBN 0130851981.

Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35(2): 137 – 144, 2015. ISSN 0268-4012. doi: <https://doi.org/10.1016/j.ijinfomgt.2014.10.007>. URL <http://www.sciencedirect.com/science/article/pii/S0268401214001066>.

Z. Gao, C. Cecati, and S. X. Ding. A survey of fault diagnosis and fault-tolerant techniques—part i: Fault diagnosis with model-based and signal-based approaches. *IEEE Transactions on Industrial Electronics*, 62(6):3757–3767, 2015.

T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj. Real-time motor fault detection by 1-d convolutional neural networks. *IEEE Transactions on Industrial Electronics*, 63 (11):7067–7075, 2016.

Olivier Janssens, Viktor Slavkovikj, Bram Vervisch, Kurt Stockman, Mia Loccupier, Steven Verstockt, Rik Van de Walle, and Sofie Van Hoecke. Convolutional neural network based fault detection for rotating machinery. *Journal of Sound and Vibration*, 377:331 – 345, 2016. ISSN 0022-460X. doi: <https://doi.org/10.1016/j.jsv.2016.05.027>. URL <http://www.sciencedirect.com/science/article/pii/S0022460X16301638>.

Y. Jiang and S. Yin. Recursive total principle component regression based fault detection and its application to vehicular cyber-physical systems. *IEEE Transactions on Industrial Informatics*, 14(4):1415–1423, 2018. doi: 10.1109/TII.2017.2752709.

Christopher Kanan and Garrison Cottrell. Color-to-grayscale: Does the method matter in image recognition? *PloS one*, 7:e29740, 01 2012. doi: 10.1371/journal.pone.0029740.

et al. Karpathy A. Cs231n convolutional neural networks for visual recognition. 1, 2016.

Rob Kitchin and Gavin McArdle. What makes big data, big data? exploring the ontological characteristics of 26 datasets. *Big Data & Society*, 3(1):2053951716631130, 2016. doi: 10.1177/2053951716631130.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782. doi: 10.1145/3065386. URL <https://doi.org/10.1145/3065386>.

Christian Krupitzer, Tim Wagenhals, Marwin Züfle, Veronika Lesch, Dominik Schäfer, Amin Mozaffarin, Janick Edinger, Christian Becker, and Samuel Kounev. A survey on predictive maintenance for industry 4.0, 02 2020.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Christian Lessmeier, James Kimotho, Detmar Zimmer, and Walter Sextro. Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification. 07 2016.

Qing Liu and Chenxi Huang. A fault diagnosis method based on transfer convolutional neural networks. *IEEE Access*, PP:1–1, 11 2019. doi: 10.1109/ACCESS.2019.2956052.

F. Lv, C. Wen, Z. Bao, and M. Liu. Fault diagnosis based on deep learning. In *2016 American Control Conference (ACC)*, pages 6851–6856, 2016.

P.D. McFadden and J.D. Smith. Model for the vibration produced by a single point defect in a rolling element bearing. *Journal of Sound and Vibration*, 96(1):69 – 82, 1984. ISSN 0022-460X. doi: [https://doi.org/10.1016/0022-460X\(84\)90595-9](https://doi.org/10.1016/0022-460X(84)90595-9). URL <http://www.sciencedirect.com/science/article/pii/0022460X84905959>.

Patrick Nectoux, Rafael Gouriveau, Kamal Medjaher, Emmanuel Ramasso, Brigitte Chebel-Morello, Noureddine Zerhouni, and Christophe Varnier. Pronostia: An experimental platform for bearings accelerated degradation tests. pages 1–8, 06 2012.

D. Neupane and J. Seok. Bearing fault detection and diagnosis using case western reserve university dataset with deep learning approaches: A review. *IEEE Access*, 8:93155–93178, 2020.

M Nielsen. Neural networks and deep learning. 1, 2015.

- V. Pandhare, J. Singh, and J. Lee. Convolutional neural network based rolling-element bearing fault diagnosis for naturally occurring and progressing defects using time-frequency domain features. In *2019 Prognostics and System Health Management Conference (PHM-Paris)*, pages 320–326, 2019. doi: 10.1109/PHM-Paris.2019.00061.
- F. Perez and B. E. Granger. Ipython: A system for interactive scientific computing. *Computing in Science Engineering*, 9(3):21–29, 2007. doi: 10.1109/MCSE.2007.53.
- L. Pintelon and A. Parodiherz. *Maintenance: An evolution perspective*. Complex System maintenance handbook. Springer, Berlin, 2008.
- Y. Roh, G. Heo, and S. E. Whang. A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2019. doi: 10.1109/TKDE.2019.2946162.
- M.S. Safizadeh and Kourosh Latifi. Using multi-sensor data fusion for vibration fault diagnosis of rolling element bearings by accelerometer and load cell. *Information Fusion*, 18:1–8, 07 2014. doi: 10.1016/j.inffus.2013.10.002.
- Wade Smith and R.B. Randall. Rolling element bearing diagnostics using the case western reserve university data: A benchmark study. *Mechanical Systems and Signal Processing*, 64-65, 05 2015. doi: 10.1016/j.ymssp.2015.04.021.
- C. Solomon and T. Breckon. Fundamentos de processamento digital de imagens: uma abordagem prática com exemplos em matlab. *Grupo Gen-LTC*, 2013.
- Tong Guan, Z. Zhang, Wen Dong, Chunming Qiao, and X. Gu. Data-driven fault diagnosis with missing syndromes imputation for functional test through conditional specification. In *2017 22nd IEEE European Test Symposium (ETS)*, pages 1–6, 2017.
- T. Wang, D. J. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3304–3308, 2012.
- L. Wen, X. Li, L. Gao, and Y. Zhang. A new convolutional neural network-based data-driven fault diagnosis method. *IEEE Transactions on Industrial Electronics*, 65(7):5990–5998, 2018. doi: 10.1109/TIE.2017.2774777.
- Li X. Gao L. Wen, L. A transfer convolutional neural network for fault diagnosis based on resnet-50. *Neural Comput Applic*, 32:6111–6124, 2020. doi: <https://doi.org/10.1007/s00521-019-04097-w>.
- M. Xia, T. Li, L. Xu, L. Liu, and C. W. de Silva. Fault diagnosis for rotating machinery using multiple sensors and convolutional neural networks. *IEEE/ASME Transactions on Mechatronics*, 23(1):101–110, 2018. doi: 10.1109/TMECH.2017.2728371.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27, pages 3320–3328. Curran Associates, Inc., 2014. URL <https://proceedings.neurips.cc/paper/2014/file/375c71349b295fbe2dcdca9206f20a06-Paper.pdf>.

D. You, X. Gao, and S. Katayama. Wpd-pca-based laser welding process monitoring and defects diagnosis by using fnn and svm. *IEEE Transactions on Industrial Electronics*, 62(1):628–636, 2015. doi: 10.1109/TIE.2014.2319216.

Jafar Zarei and Javad Poshtan. An advanced park’s vectors approach for bearing fault detection. *Tribology International*, 42(2):213 – 219, 2009. ISSN 0301-679X. doi: <https://doi.org/10.1016/j.triboint.2008.06.002>. URL <http://www.sciencedirect.com/science/article/pii/S0301679X08001291>.

Qingchen Zhang, Laurence T. Yang, Zhikui Chen, and Peng Li. A survey on deep learning for big data. *Information Fusion*, 42:146 – 157, 2018. ISSN 1566-2535. doi: <https://doi.org/10.1016/j.inffus.2017.10.006>. URL <http://www.sciencedirect.com/science/article/pii/S1566253517305328>.

S. Zhang, S. Zhang, B. Wang, and T. G. Habetler. Deep learning algorithms for bearing fault diagnostics—a comprehensive review. *IEEE Access*, 8:29857–29881, 2020. doi: 10.1109/ACCESS.2020.2972859.

H. Zheng, R. Wang, Y. Yang, J. Yin, Y. Li, Y. Li, and M. Xu. Cross-domain fault diagnosis using knowledge transfer strategy: A review. *IEEE Access*, 7:129260–129290, 2019. doi: 10.1109/ACCESS.2019.2939876.

Zhiyu Zhu, Gaoliang Peng, Yuanhang Chen, and Huijun Gao. A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis. *Neurocomputing*, 323:62 – 75, 2019. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2018.09.050>. URL <http://www.sciencedirect.com/science/article/pii/S0925231218311238>.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex Smola. Parallelized stochastic gradient descent. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23, pages 2595–2603. Curran Associates, Inc., 2010. URL <https://proceedings.neurips.cc/paper/2010/file/abea47ba24142ed16b7d8fbf2c740e0d-Paper.pdf>.

Barret Zoph, Ekin D. Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V. Le. Learning data augmentation strategies for object detection. In Andrea Vedaldi, Horst

Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 566–583, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58583-9.