

UNIVERSIDADE PAULISTA

ALEX LUÍS COLOMBARI

**APRENDIZADO DE MÁQUINA APLICADO EM MANUTENÇÃO PREDITIVA DE
EQUIPAMENTOS INDUSTRIAIS**

Limeira

2019

UNIVERSIDADE PAULISTA

ALEX LUÍS COLOMBARI

**APRENDIZADO DE MÁQUINA APLICADO EM MANUTENÇÃO PREDITIVA DE
EQUIPAMENTOS INDUSTRIAIS**

Monografia apresentada à banca examinadora do curso de
Ciência da Computação da Universidade Paulista, como
requisito parcial à obtenção do título de Bacharel em
Ciência da Computação, realizado sob a orientação do
professor Me. Amaury Bosso André.

Limeira

2019

ALEX LUÍS COLOMBARI

**APRENDIZADO DE MÁQUINA APLICADO EM MANUTENÇÃO PREDITIVA DE
EQUIPAMENTOS INDUSTRIAIS**

Monografia apresentada à banca examinadora do curso de
Ciência da Computação da Universidade Paulista, como
requisito parcial à obtenção do título de Bacharel em
Ciência da Computação, realizado sob a orientação do
professor Me. Amaury Bosso André.

Aprovada em XX de XXXXX de 201X.

BANCA EXAMINADORA

Prof. Dr. Nome completo

Prof. Me. Nome completo

Prof. Esp. Nome completo

Dedico este trabalho aos meus amigos que estiveram presentes durante o período da graduação, todos os professores pelos conselhos e todo o conhecimento compartilhado, e em especial meus pais, pois tornaram possível, com todo o apoio, a realização do Bacharel em Ciência da Computação.

AGRADECIMENTOS

Meu agradecimento vai para todos os professores da graduação, em especial ao Amaury André, Marcos Gialdi, Mateus Locci e Sérgio Nunes por todo o apoio e conversa que resultou na conclusão desta pesquisa.

Ao meu orientador Amaury André, que além de ter me proporcionado uma excelente oportunidade profissional na área, dedicou seu tempo para ajudar durante todo o processo.

Ao professor Mateus Locci, por sempre se mostrar interessado no desenvolvido deste trabalho e por dar outra perspectiva sobre meu trabalho.

Gostaria de agradecer também os meus colegas de classe pelas vezes em que fui questionado, me forçando a buscar novos conhecimentos sobre a pesquisa, e também pela troca de conhecimento.

Agradeço também os meus amigos de fora da universidade, que acompanharam de perto o desenvolvimento, onde, através de críticas e dúvidas, foi gerado um amadurecimento no decorrer da conclusão do trabalho.

Por fim, agradeço aos meus pais, que são os responsáveis por isso se tornar realidade. Por todo o apoio em nos momentos mais difíceis e pela oportunidade de obter o título de Bacharel em Ciência da Computação.

*“Nós só podemos ver um pouco do futuro, mas o suficiente para perceber que há
muito a fazer”. (Alan Turing)*

RESUMO

COLOMBARI, A. C. **Aprendizado de máquina aplicado em manutenção preditiva de equipamentos industriais**. 2019. 42p. Monografia (Trabalho de Conclusão de Curso) – Curso de Ciência da Computação, Universidade Paulista, Limeira, 2019.

Uma vez que algoritmos de aprendizado de máquina são adaptáveis, podemos então implementá-los nas mais distintas áreas profissionais, sendo necessário a utilização de grandes bases de dados. Tendo esta perspectiva, essa monografia tem como principal objetivo alavancar a pesquisa e o avanço científico no campo de manutenção preditiva de equipamentos industriais e inteligência artificial, através da implementação de um modelo de aprendizado de máquina utilizando redes neurais convolucionais para desempenhar a classificação de imagens de amostras de óleo, distribuídos em treze classes distintas de falhas morfológicas que podem ser, partículas contaminantes, como areia ou óxido e até mesmo partículas de desgaste de componentes do equipamento, oriundas de desbalanceamento e afins. Objetivando a compreensão do assunto, a monografia apresenta de forma introdutória as diferentes técnicas de manutenção de equipamentos industriais, abordando manutenção corretiva, preditiva e preventiva, além de dissertar e aprofundar os conhecimentos acerca de aprendizado de máquina e inteligência artificial.

Palavras-chave: Aprendizado de Máquina; Aprendizado Profundo; Inteligência Artificial; Manutenção Preditiva; Redes Neurais.

ABSTRACT

COLOMBARI, A. L. *Machine Learning applied to predictive maintenance of industrial equipment*. 2019. 42p. Monograph (Bachelor's thesis) – Computer Science, Universidade Paulista, Limeira, 2019.

Since machine learning algorithms are adaptable, we can implement them in many different professional areas, requiring the use of large databases. From this perspective, this monograph aims to leverage research and scientific advancement in predictive maintenance of industrial equipments and artificial intelligence, through the implementation of a machine learning model using convolutional neural networks to perform image classification of oil samples, distributed in thirteen distinct classes of morphological faults that can be contaminating particles, such as sand or oxide and wear particles of equipment components, coming from unbalance and the like. Aiming at understanding the subject, the monograph introduces the different techniques of maintenance of industrial equipment, addressing corrective, predictive and preventive maintenance, as well as disserting and deepening the knowledge about machine learning and artificial intelligence.

Key-words: *Artificial Intelligence; Deep Learning; Machine Learning; Neural Network; Predictive Maintenance.*

LISTA DE FIGURAS

Figura 1: Viscosímetro	18
Figura 2: Esquema de um ferrografo	19
Figura 3: Modelo de um neurônio biológico	22
Figura 4: Representação de um neurônio computacional	23
Figura 5: Exemplos de funções de ativação	24
Figura 6: Esquema de neurônios compondo uma Rede Neural Multicamadas	26
Figura 7: Representação da codificação one-hot	26
Figura 8: Exemplo de convolução de filtro 3x3	29
Figura 9: Funcionamento do parâmetro stride	30
Figura 10: Operação do Max Pooling em imagem 4x4	31
Figura 11: Arquitetura de rede neural convolucional para classificação de imagens	31
Figura 12: Exemplo de Autoencoder	32
Figura 13: Exemplos de amostra de óleo	33
Figura 14: Normalização em uma imagem 4x4	34
Figura 15: Reconstrução gerada pelo Autoencoder	37
Figura 16: Função de custo da validação do Autoencoder	38

LISTA DE TABELAS

Tabela 1: Codificação One-hot das classes	35
Tabela 2: Resultado de treino do Autoencoder	37
Tabela 3: Desempenho variando as camadas de convolução	38
Tabela 4: Desempenho variando as camadas FC	39

LISTA DE ABREVIATURAS

<i>CNN</i>	Redes Neurais Convolucionais (<i>Convolutional Neural Networks</i>)
<i>FC</i>	Camada Totalmente Conectada (<i>Fully Connected Layer</i>)
<i>GD</i>	Gradiente Descendente (<i>Gradient Descent</i>)
IA	Inteligência Artificial
<i>ML</i>	Aprendizado de máquina (<i>Machine Learning</i>)
<i>MLNN</i>	Rede Neural Multicamadas (<i>Multi-layer Neural Network</i>)
NAS 1638	<i>National Aerospace Standard 1638</i>
ReLU	Linear Retificada (<i>Rectified Linear Unit</i>)
RNA	Redes Neurais Artificiais
Tanh	Tangente Hiperbólica (<i>Hyperbolic Tangent</i>)

SUMÁRIO

1 INTRODUÇÃO	14
1.1 OBJETIVO	14
1.2 JUSTIFICATIVA	15
1.3 METODOLOGIA	15
2 TÉCNICAS DE MANUTENÇÃO	17
2.1 TÉCNICA PREDITIVA	17
2.1.1 Análise de óleo	17
2.1.1.1 Análise físico-químico	17
2.1.1.2 Contagem de partículas	18
2.1.1.3 Análise espectrográfica	18
2.1.1.4 Ferrografia	18
2.2 TÉCNICA CORRETIVA	19
2.3 TÉCNICA PREVENTIVA	19
2.4 VANTAGENS DO USO DA MANUTENÇÃO PREDITIVA	20
3 INTELIGÊNCIA ARTIFICIAL	20
3.1 TESTE DE TURING	21
3.2 APRENDIZADO DE MÁQUINA	21
3.2.1 REDES NEURAIS ARTIFICIAIS	21
3.2.1.1 Neurônio Cerebral	22
3.2.1.2 Neurônio Artificial	22
3.2.2 Funções de ativação	24
3.2.2.1 Função Sigmoid	24
3.2.2.2 Função ReLU	25
3.2.2.3 Função Softmax	25
3.2.3 Rede Neural Multicamadas	25
3.2.4 Codificação One-hot	26
3.2.8 Overfitting e Underfitting	27
3.2.9 Treinamento de uma Rede Neural Multicamadas	27
3.3 APRENDIZADO PROFUNDO	28
3.3.1 Dropout	28
3.4 REDES NEURAIS CONVOLUCIONAIS	28

3.4.1 Camada Convolutacional	29
3.4.1.1 Passo (<i>stride</i>)	29
3.4.1.2 Preenchimento (<i>padding</i>)	30
3.4.1.3 Camadas de Pooling	30
3.4.1.4 Camadas <i>Dense</i> (<i>Fully Connected Layer</i>)	31
3.5 <i>AUTOENCODER</i>	31
4 IMPLEMENTAÇÃO	32
4.1 <i>MATERIAIS</i>	32
4.1.1 Keras e TensorFlow	33
4.1.2 Base de Dados (<i>Dataset</i>)	33
4.2 <i>PRÉ-PROCESSAMENTO</i>	33
4.2.1 Normalização	34
4.2.1.1 Conversão e separação dos dados	34
4.2.2 Codificação One-hot	34
4.3 <i>TREINAMENTO DA CNN</i>	35
5 TESTES E RESULTADOS	36
5.1 <i>Problemas e balanceamento das classes</i>	36
5.2 <i>Treinamento e teste com Autoencoder</i>	36
5.3 <i>Treinamento e Teste da Rede Neural Convolutacional</i>	38
6 CONCLUSÃO	39
6.1 <i>Trabalhos futuros</i>	40
REFERÊNCIAS BIBLIOGRÁFICAS	41

1 INTRODUÇÃO

A inteligência artificial (IA) teve seu crescimento elevado no mundo algumas décadas atrás, logo após o término da Segunda Guerra Mundial como avanço da tecnologia e a necessidade da evolução da mesma. O termo IA foi cunhado em 1956 e tem seu principal objetivo fazer com que os computadores trabalhem da mesma forma que o cérebro humano. Ganhou seu real significado após o desenvolvimento do famoso Teste de Turing, criado por Alan Turing em 1950 (NORVIG, Peter; RUSSELL, Stuart. 1995).

Quando falamos de inteligência artificial, estamos falando de um assunto abrangente e complexo. Dentro desta área científica, temos um campo de estudos conhecido como Redes Neurais Artificiais, responsável por desenvolver algoritmos que buscam imitar o funcionamento do neurônio humano e seu raciocínio, onde o modelo matemático utilizado até hoje foi criado por McCulloch e Pitts em 1943.

Com a invasão da tecnologia no mundo, originou-se inúmeras ferramentas para auxílio das tarefas diárias nos mais variados setores. Partindo desse princípio, nasceu o que chamamos de técnicas de manutenção, essas que são voltadas para equipamentos industriais. Existem algumas variações dentre as técnicas, conhecidas como corretiva, preditiva e preventiva.

Manutenção preditiva é aquela que busca constatar de forma antecipada a ocorrência de falhas em maquinários, é feita com a utilização de algumas técnicas como análise de amostras de óleo, sensores de vibração, termografia, análise de motores elétricos, ultrassom, entre outros (AZEVEDO, 2009).

Uma vez que as amostras de óleo são usadas para buscar a origem das falhas morfológicas e sabendo a fácil adaptação da IA, podemos então criar um modelo utilizando redes neurais a fim de realizar a classificação dessas amostras retornando qual falha morfológica está presente.

1.1 OBJETIVO

Essa pesquisa busca promover o avanço científico por meio do desenvolvimento de uma solução utilizando a junção de inteligência artificial e manutenção preditiva, com o intuito de criar um modelo de rede neural convolucional utilizando imagens microscópicas com o objetivo de analisar amostras de óleo em laboratório onde, através do treinamento do modelo, seja possível realizar a análise e a identificação do tipo de falha presente na amostra, e através disto,

testar e validar variações na arquitetura das redes neurais convolucionais buscando mostrar seu desempenho diante este problema.

1.2 JUSTIFICATIVA

A ideia de elaborar e desenvolver esse projeto se deu sob o convívio em um ambiente de pesquisa e desenvolvimento envolvendo manutenção preditiva, onde nesse momento, foi possível perceber a necessidade de criação de um agente computacional capaz de auxiliar os profissionais que trabalham em laboratórios e outros responsáveis por gerenciar a manutenção preditiva de equipamentos industriais onde, justifica-se o desenvolvimento desse projeto, visando ajudar profissionais dessa área, além de evidenciar a capacidade de adaptação da inteligência artificial nos mais variados meios.

1.3 METODOLOGIA

O primeiro passo para que se possa aplicar algoritmos de aprendizado de máquina é obter algum conjunto de dados. No caso desta pesquisa, foi adquirido um dataset com um total de nove mil imagens a nível microscópico de amostras de óleos, geradas a posteriori ensaios e testes físicos e químicos já dentro de laboratório. Tais imagens foram rotuladas previamente com as falhas encontradas nelas, o que nos possibilita trabalhar com aprendizado supervisionado.

Com a obtenção do conjunto de dados, devemos então converter todas as imagens para matrizes numéricas, uma vez que as redes neurais trabalham apenas com valores numéricos. Fazendo a representação das imagens em matrizes, temos então os valores equivalentes de cada cor presente, onde para facilitar a generalização da rede, devemos fazer a normalização dos pixels para valores menores.

Após a manipulação dos dados, é possível passar os valores para a rede neural, que neste caso, para teste, foi implementado a arquitetura de Rede Neural Convolucional, onde posteriormente, foram feitos testes com a arquitetura de Autoencoder. O objetivo é passar as imagens e seus respectivos rótulos que, por meio do aprendizado de máquina, o computador seja capaz de identificar padrões entre as imagens e os rótulos, até atingir um valor de precisão consideravelmente alto, gerando assim uma boa taxa assertiva.

Com um modelo capaz de realizar previsões com novas imagens, uma boa abordagem é implementar a geração de um relatório da amostra analisada, esse relatório por sua vez, tem

a autonomia de informar profissionais que atuam na área, as possíveis medidas que possam ser tomadas diante determinado problema, seja uma troca de lubrificante, substituição de uma peça, entre outros.

2 TÉCNICAS DE MANUTENÇÃO

Dentro do mundo industrial, há a existência de algumas técnicas para manutenção, que se diferenciam e possuem particularidades, sendo preditiva, preventiva e corretiva.

2.1 TÉCNICA PREDITIVA

De acordo com Azevedo (2009) a manutenção preditiva busca constatar antecipadamente a ocorrência de falhas em maquinários, através de monitoramento ou análise das condições de funcionamento, dadas durante a operação do equipamento. Ainda segundo Azevedo (2009), através da aplicação dos métodos preditivos, é criado um relatório final, onde caso se constate avarias é encaminhado à um responsável para que sejam tomadas todas as medidas necessárias, a fim de resolver os problemas especificados.

Existem metodologias empregadas pela técnica preditiva, que são sensores de vibração, termografia, análise elétrica de motores, ultrassom, entre outros. Para suprir o entendimento e compreensão desta pesquisa, será detalhado apenas a análise de óleo, como vemos a seguir:

2.1.1 Análise de óleo

Após a coleta das amostras, são levadas para laboratório, onde são submetidas à uma série de ensaios físico-químicos (viscosidade, ponto de fulgor, absorção etc.). Além disso, um passo muito importante é a contagem de partículas sendo ela, espectrográfica e ferrografia.

2.1.1.1 Análise físico-químico

Para Ronilson de Carvalho (2014), essa análise busca apontar a aptidão de isolamento e o estado de desgaste do óleo lubrificante. Sendo o resultado da análise aferido com valores já determinados por normas. Caso haja discrepância entre os valores obtidos no teste em comparação às normas (fora do padrão), são indicadores de que pode haver a necessidade de realizar a troca do lubrificante. É realizado também o teste de viscosidade da amostra, com a utilização do viscosímetro (Figura 1).

Figura 1: Viscosímetro



Fonte: Láctea Científica

2.1.1.2 Contagem de partículas

É feita a partir de um contador de partículas, que realiza a classificação em faixas granulométricas. Essa técnica segundo Ronilson (2014), dentro da análise de óleo, para realizar a classificação do nível de impureza usam-se os métodos *NAS 1638*, *ISO 4406-99* e *SAE AS 4059*.

2.1.1.3 Análise espectrográfica

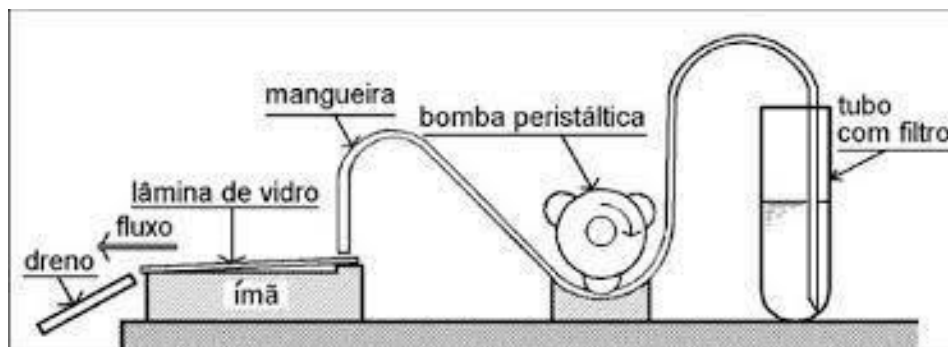
Ronilson de Carvalho (2014, p.6) fala que “A análise espectrográfica informa ao usuário sobre as concentrações de determinados elementos químicos”. A partir do momento em que é obtido conhecimento sobre as concentrações máximas toleráveis no óleo lubrificante, pode ser detectados proporções de contaminação que apontem avarias no equipamento, o que torna totalmente viável a tomada de ações preventivas para corrigir as contaminações antes mesmo que ocorra falha ou degradação irreversível. Ainda para Ronilson (2014), damos como exemplo: em uma amostra de óleo diferencial com alto nível de metais, pode indicar que um rolamento está prestes a falhar e assim por diante.

2.1.1.4 Ferrografia

Esse processo envolve a contagem e observação visual das partículas, sendo capaz de produzir resultados bastante precisos, pois tem a particularidade de avaliar quantitativamente os componentes.

O processo envolve o uso de um ferrografo (Figura 2) sendo parecida com a análise espectrográfica, com o intuito de determinar o grau de desgaste. Nesse processo, são contadas as partículas menores, oriundas de desgaste normal, e as maiores que são de desgastes severos.

Figura 2: Esquema de um ferrografo



Fonte: Retirado do artigo escrito por Ronilson de Carvalho (2014)

2.2 TÉCNICA CORRETIVA

A empregabilidade dessa técnica também busca corrigir falhas para recuperar o funcionamento de equipamentos industriais. Sendo metodologia de correção, acaba sendo muito caro, principalmente pelo fato de haver estoque de peças, outro problema é que muitas vezes há paralisação do maquinário para realizar a manutenção, e acarreta de forma negativa na produtividade.

Caio Monteiro (2010, p.4) ressalta “Manutenção corretiva é aquela de atendimento imediato à produção. Esse tipo de manutenção baseia-se na seguinte filosofia: “equipamento parou, manutenção conserta imediatamente”.

2.3 TÉCNICA PREVENTIVA

Obedece um plano de manutenção previamente elaborado, que são baseados em intervalos de tempo, desse modo, é uma técnica totalmente baseada no tempo.

Segundo Alien Souza (200-?), é determinado um tempo para se estabelecer as manutenções preventivas, de acordo com uma série de dados estatísticos ou históricos, a partir disso, procura-se estipular o tempo provável para a ocorrência da avaria, sabendo que em algum momento haverá o problema, porém sem saber o momento exato.

2.4 VANTAGENS DO USO DA MANUTENÇÃO PREDITIVA

Como tivemos uma visão geral entre três diferentes métodos de manutenção, podemos então dizer o porquê a manutenção preditiva se destaca no meio das outras.

A manutenção preditiva leva vantagem em comparação às outras no momento em que ela é capaz de adotar metodologias que buscam antecipar de forma precisa, o desgaste de óleo lubrificante ou peças de equipamentos industriais. Diferente da corretiva que atua a posteriori do acontecimento da falha, e distinta da preventiva que é realizada de tempos em tempos, a preditiva monitora constantemente a condição dos ativos, por meio de sensores, análises de amostras de lubrificantes ou outros métodos. Por tanto, através da sua qualificação para realizar manutenções precisas, tem a capacidade de efetuar trocas de lubrificantes e peças apenas em momentos onde há necessidade, ou seja, é uma técnica que diminui gastos desnecessários, trazendo conforto e segurança para as empresas.

3 INTELIGÊNCIA ARTIFICIAL

Visando entender o assunto como um todo, é de suma importância a compreensão do significado de inteligência artificial, sendo descrita por Peter Norvig e Stuart Russell (1995):

Denominamos nossa espécie Homo Sapiens - homem sábio - porque nossa inteligência é tão importante para nós. Durante milhares de anos, procuramos entender como pensamos, isto é, como um mero punhado de matéria pode perceber, compreender, prever e manipular um mundo muito maior e mais complicado que ela própria. O campo da inteligência artificial, ou IA, vai ainda mais além: ele tenta não apenas compreender, mas também construir entidades inteligentes. (p.24).

Sendo um dos campos mais recentes no ramo científico, teve seu surgimento logo após o término da Segunda Guerra Mundial. Segundo Peter e Stuart (1995), atualmente a IA está presente em diversos campos, partindo do contexto geral (aprendizagem e percepção) até tarefas mais particulares como, diagnósticos de doenças, carros autônomos, jogos de xadrez, entre outros.

A partir disso, podemos concluir que inteligência artificial é aquela que compreende, de forma computacional, a mente humana. Sendo capaz de agir racionalmente em tomadas de decisões, criar máquinas capacitadas para exercer atividades que hoje são executada por pessoas, resolver problemas, etc.

3.1 TESTE DE TURING

Foi arquitetado por Alan Turing em 1950 para fornecer uma definição satisfatória de inteligência. Um computador só passará neste teste a partir do momento em que um interrogador humano, após propor algumas perguntas por escrito, não conseguir identificar se as respostas escritas vêm de uma pessoa ou de um computador (NORVIG, Peter; RUSSELL, Stuart. 1995).

Para um computador ser dado como inteligente, precisaria possuir as seguintes capacidades:

- **Aprendizado de Máquina** onde se adapta a novas circunstâncias e para detectar e extrapolar padrões;
- **Processamento de linguagem natural** que permite que ele se comunique com sucesso em um idioma natural;
- **Representação do conhecimento** para armazenar o que sabe ou ouve;
- **Raciocínio automatizado** para usar as informações armazenadas com a finalidade de responder a perguntas e tirar novas conclusões.

3.2 APRENDIZADO DE MÁQUINA

Sabendo o que é Inteligência Artificial, esse capítulo introduz o Aprendizado de Máquina, do inglês *Machine Learning* (ML), mostrando quais elementos foram utilizados nesse projeto. Serão abordados Redes Neurais Artificiais (RNA) explicando o que são os neurônios artificiais, funções de ativação, *Cross Entropy* e codificação *one-hot*.

3.2.1 REDES NEURAIIS ARTIFICIAIS

Edberto Ferneda (2006) nos diz que, as redes neurais constituem em um campo de inteligência artificial que destinam-se a efetuar cálculos matemáticos que representem estruturas neurais biológicas.

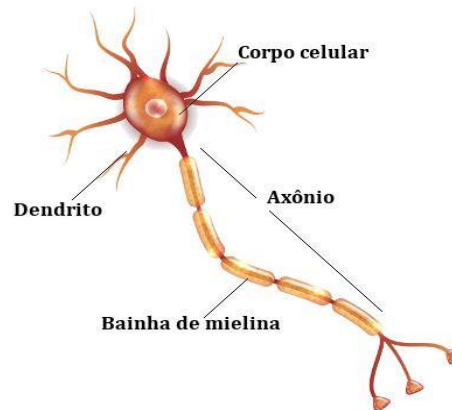
Ainda sobre a explicação teórica para os autores Peter e Stuart (1995), temos que a inteligência artificial é também uma junção da biologia, no caso das redes neurais artificiais, inspirado na neurociência, dizendo que a atividade mental constitui em atividade eletroquímica, onde as células são chamadas de neurônios.

3.2.1.1 Neurônio Cerebral

A conexão das células (neurônios) cria a rede neural do nosso cérebro. As RNA (Redes Neurais Artificiais) são uma imitação da rede neural biológica, onde os neurônios artificiais são conectados no mesmo estilo da rede cerebral.

Um neurônio biológico (Figura 3) é composto por seu corpo celular, axônio e dendrito. O dendrito é responsável por receber os sinais eletroquímicos de outros neurônios para seu corpo celular. Já o corpo celular conhecido como Soma, contém o núcleo e outras estruturas químicas. Os axônios carregam os sinais de um neurônio a outro. A conexão entre os dendritos de dois neurônios ou para uma célula muscular é chamada de sinapse (SHIRURU, 2016).

Figura 3: Modelo de um neurônio biológico



Fonte: Artigo “O que é neurônio?”, do site Brasil Escola

Um sinal recebido pelo dendrito, quando ultrapassa certo *threshold* (limite) o neurônio dispara seu próprio sinal para ser passado ao próximo neurônio através do axônio usando a sinapse. Um enorme número de neurônios trabalham de forma simultânea. Nosso cérebro possui capacidade de armazenar uma grande quantidade de dados.

3.2.1.2 Neurônio Artificial

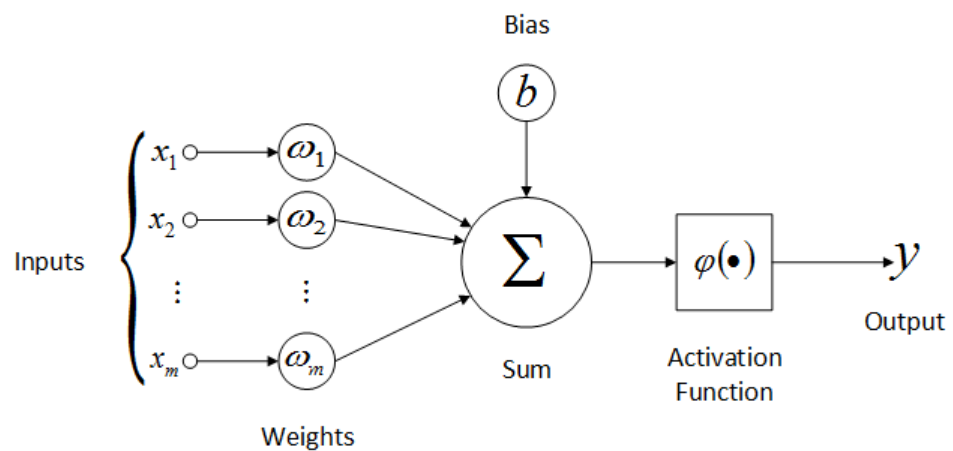
Um neurônio artificial tenta replicar o comportamento e a estrutura de um neurônio natural. A composição base é dada pela analogia com o neurônio biológico, onde o *input* (entrada) é comparado aos dendritos, e o *output* (saída) é a sinapse que ocorre via axônio.

A explicação de um neurônio artificial é dada por:

Grosseiramente falando, ele “dispara” quando uma combinação linear de suas entradas excede algum limiar (rígido ou suave), ou seja, ele implementa um classificador linear do tipo descrito na seção anterior. Uma rede neural é apenas uma coleção de unidades conectadas; as propriedades da rede são determinadas pela sua topologia e pelas propriedades dos “neurônios”. (NORVIG, Peter; RUSSELL, Stuart. 1995, p. 843).

A representação de um neurônio computacional é dado pelo modelo matemático desenvolvido por McCulloch e Pitts em 1943 (Figura 4).

Figura 4: Representação de um neurônio computacional



Fonte: Research Gate

Onde, os elementos representam:

- Um conjunto de n entradas (x_1, x_2, \dots, x_n), e seus respectivos pesos (w_1, w_2, \dots, w_n);
- Os pesos sinápticos (W_n) onde os sinais de entrada é correlacionado à um peso diferente que varia conforme o treinamento da RNA, sendo determinado a influência da entrada para o resultado final;
- Somatório (Σ) dos valores da entrada;
- Bias (b) é uma entrada “1” associada a um peso “ b ” individualmente em cada neurônio, tem como função aumentar ou diminuir a entrada, de forma que possa transferir a função de ativação no eixo;
- Função de ativação (φ), atua como *threshold* (limiar de ativação) do intervalo aceitável do sinal de saída com um valor fixo (y). Aplica não-linearidade no valor do neurônio e determina sua forma de ativação.

O neurônio computacional em sua forma matemática, é descrita pela equação 3.1:

$$Y_k = \phi \left(\sum_{i=1}^M X_i W_{ik} + b_k \right) \quad (3.1)$$

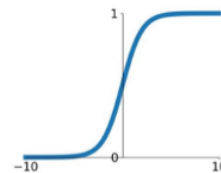
3.2.2 Funções de ativação

Existem várias funções de ativação que podem ser utilizadas dentro da rede neural, as mais comuns são: ReLU (*Rectified Linear Unit*), Sigmoid, Softmax e Tanh (*Hyperbolic Tangent*). Segundo Miyazaki (2017), são funções não-lineares que são conectadas ao final de cada neurônio artificial (Figura 4), baseadas em seu modelo biológico e que definem sua saída através dos dados da entrada em seu limiar de ativação.

Figura 5: Exemplos de funções de ativação

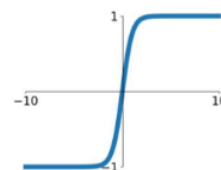
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



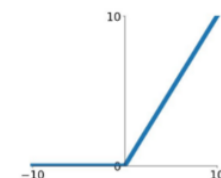
tanh

$$\tanh(x)$$



ReLU

$$\max(0, x)$$



Fonte: Introduction to Exponential Linear Unit (2018)

Para esta pesquisa, será abordado apenas as funções utilizadas, que são a ReLU, Sigmoid e a Softmax (apesar de não ser utilizada no resultado final).

3.2.2.1 Função Sigmoid

A função *Sigmoid* é abundantemente utilizada em problemas que envolvem classificação (sendo a principal função implementada nesta pesquisa) e tem como saída valores entre 0 e 1, e retorna a probabilidade de cada dado de entrada pertencer às classes analisadas, seu cálculo é:

$$\phi(x) = \frac{1}{1+e^{-x}} \quad (3.2)$$

3.2.2.2 Função ReLU

Função Linear Retificada, do inglês *Rectified Linear Unit* (ReLU), foi a mais recente função criada, testes realizados por Krizhevsky reportou que a ReLU teve um resultado 6 vezes mais rápido em comparação a outras ativações na mesma rede neural. Dentre elas, é a única que não utiliza expoente em sua fórmula. Sua função é dada por:

$$\phi(x) = \max(0, x) \quad (3.3)$$

3.2.2.3 Função Softmax

A função *Softmax* é muito utilizada em classificadores na camada de saída, seu objetivo é representar a possibilidade de cada classe em função da entrada, individualmente. Em sua equação, é possível notar que os resultados podem ser valores entre 0 e 1, e a soma de probabilidade das classes é igual a 1, ou seja, a classe determinada pela rede será a de maior valor, a maior probabilidade.

$$\text{softmax}(Z_i) = \frac{e^{(Z_i)}}{\sum_j e^{(Z_j)}} \quad (3.4)$$

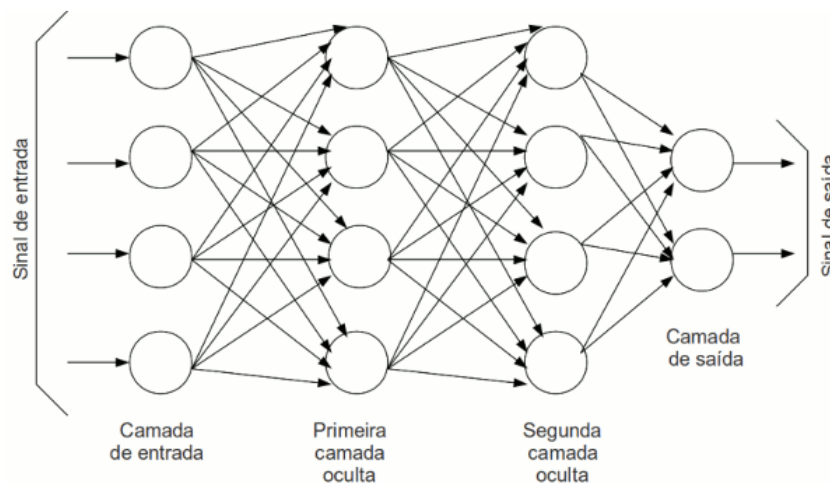
3.2.3 Rede Neural Multicamadas

Modelos baseados em uma única camada, ou seja, de apenas um neurônio artificial, são chamados de Perceptron, eles podem ser utilizados para resolver problemas de classificação, porém de forma ineficaz, pois é capaz de associar apenas dados que são linearmente separáveis. Para resolver esse impasse, tendo em vista que a maioria dos problemas não são linearmente separáveis, foi criada a arquitetura mais desenvolvida chamada Rede Neural Multicamadas (*Multi-layer Neural Network - MLNN*) (HAFEMANN, 2014).

A MLNN nada mais é que a generalização de um *perceptron*, sendo determinado de forma que cada neurônio artificial seja dividido em camadas. As camadas recebem nomes, são

chamadas de entrada (*input*), intermediárias que também são conhecidas como ocultas (*hidden layers*) e as camadas de saída (*output*) (Figura 6).

Figura 6: Esquema de neurônios compondo uma Rede Neural Multicamadas



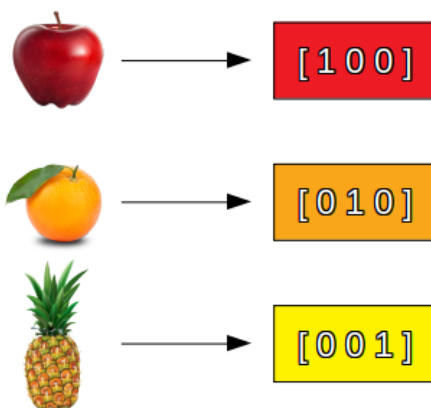
Fonte: “Redes Neurais Artificiais” retirado do site Monolito Nimbus

3.2.4 Codificação One-hot

Para Miyazaki (2017), a saída da rede neural é dada em valores numéricos, sendo necessário converter os rótulos de cada classe de objetos de maneira em que a RNA consiga diferenciar cada uma na saída da rede.

É possível realizar esse processo através da codificação *one-hot*, que converte os rótulos em vetores binários. Temos como exemplo a Figura 7, onde existem três classes: Maçã, Laranja e Abacaxi. Cada uma tem seu valor binário, respectivamente.

Figura 7: Representação da codificação one-hot



Fonte: Elaborado pelo próprio autor

3.2.8 Overfitting e Underfitting

Para determinar o desempenho da rede neural é importante separar os dados em dois grupos: conjunto de treino e conjunto de teste. Tem como objetivo diminuir o erro de treinamento e o erro de teste. No momento em que esses dois erros não caminham juntos, ocorrem os casos conhecidos como: sub-ajuste (do inglês *underfitting*) e o sobre-ajuste (em inglês *overfitting*).

O caso de *overfitting* ocorre no momento em que o erro de treinamento decresce mas o de teste continua alto. Isto é, a rede não tem capacidade de generalizar para exemplos que não foram vistos. Pode ocorrer se o conjunto de treino não for suficientemente grande ou caso exista um número excessivo de características (nós).

Já o *underfitting* ocorre quando o erro de treinamento não reduz. Isso quer dizer que o modelo não conseguiu correlacionar os dados durante o treino. Geralmente acontece quando os modelos e/ou conjunto de dados são simples, fazendo com que a rede não consiga aprender o padrão (MIYAZAKI, 2017).

3.2.9 Treinamento de uma Rede Neural Multicamadas

O treinamento de uma rede neural artificial consiste no ajuste de pesos sinápticos e *bias* de modo que o vetor de saída seja o mais próximo do resultado esperado. Existem duas etapas principais: Propagação (*forward-propagation*) e retro propagação (*back-propagation*) (NORVIG; RUSSELL, 1995).

Primeiramente o estágio de propagação dá-se quando é aplicado a Equação 3.1 do neurônio artificial para cada um dos nós da rede. Para as tarefas de classificação, a sequência de propagação segue da camada da entrada, é processado pelas camadas ocultas e por fim, termina na camada de saída. A função de ativação *Softmax* (equação 3.4) é comumente utilizada nesta última camada, sendo produzido uma saída adequada para comparação com o rótulo esperado da codificação *one-hot* (HAFEMANN, 2014).

Com o fim da propagação, a rede entrega como saída os valores conjecturados por ela. Sendo assim, é estabelecido a função perda e o desempenho da rede no momento em que a saída é comparado com os valores desejados. Enquanto este desempenho não for suficiente, inicia a fase de retro propagação, buscando reduzir o erro da estimação. Para esse processo, é necessário calcular a função *Cross Entropy* e a função de perda.

Segundo Miyazaki (2017) a cada iteração de propagação e retro propagação, o vetor dos pesos é alterado na direção em que se produz a maior queda ao longo da superfície de erro, que continua até atingir um erro mínimo local.

3.3 APRENDIZADO PROFUNDO

O aprendizado profundo (do inglês *Deep Learning* – *DL*) é a definição que se dá à modelos de MLNN que possuem mais de duas camadas ocultas e técnicas que treinam o modelo de forma eficiente. Foi desenvolvido quando os algoritmos tradicionais falharam em generalizar tarefas de IA como reconhecimento de falas e objetos (GOODFELLOW; BENGIO; COURVILLE, 2016).

De acordo com Ferreira (2017), o aumento das camadas da RNA acarreta em um crescimento considerável de parâmetros que devem ser ajustados durante a aprendizagem, portanto há dois fatores cruciais para a viabilidade da ferramenta: alto poder computacional e extensas bases de dados.

3.3.1 Dropout

É uma técnica que é responsável por regularizar os modelos de forma computacionalmente barata. Durante cada iteração, remove de forma aleatória, determinada quantidade de neurônios de uma camada que são readicionados na próxima iteração. Com a aplicação desse método, é possível fazer com que a rede aprenda atributos mais robustos, já que cada neurônio não pode depender de outro (GOODFELLOW; BENGIO; COURVILLE, 2016).

3.4 REDES NEURAIAS CONVOLUCIONAIS

As Redes Neurais Convolucionais são uma variação de arquitetura de aprendizado profundo (do inglês *Deep Learning* - *DL*) que subdivide os dados para tentar extrair características de cada conjunto. Busca reduzir o número de parâmetros a serem ajustados pela rede.

Abundantemente utilizada em tarefas com estruturas em grade, como por exemplo, processamento de fala e entendimento da linguagem natural (1D, convoluções temporais), segmentação e classificação de imagens (2D, convolução espacial), e análise de vídeos (3D,

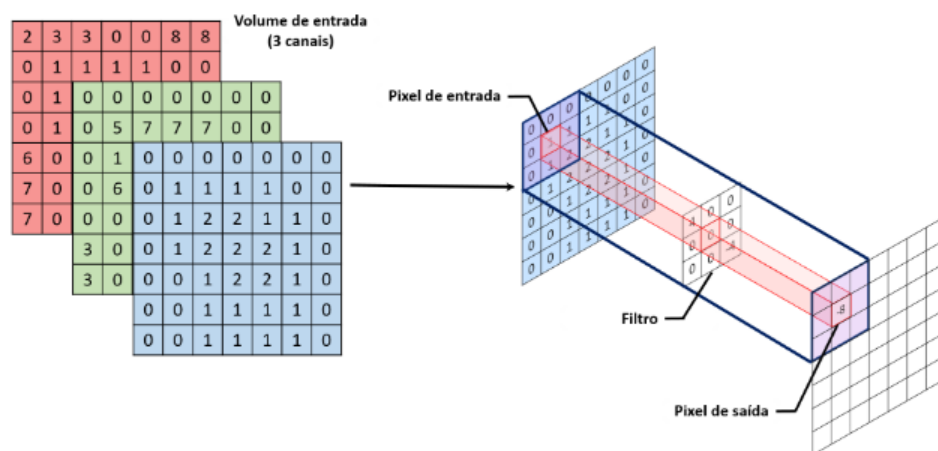
convolução volumétrica) (SIMONOVSKY; KOMODAKIS, 2017) (LECUN; BENGIO; HINTON, 2015) (MIYAZAKI, 2017).

3.4.1 Camada Convolutiva

As camadas convolucionais consistem em um conjunto de filtros que recebem as imagens como entrada. Por exemplo, no caso das imagens coloridas, existem 3 canais de cores (no padrão RGB). Sendo assim, ao usar imagens 28x28x3 significa que o filtro da primeira camada terá 28 pixels de altura e largura, e profundidade igual a 3.

Cada filtro gera uma estrutura conectada localmente que percorre toda a entrada. O somatório do produto dos valores de cada filtro e posição do volume de entrada é uma operação conhecida como convolução (Figura 8). Os valores de cada convolução são então passados para uma função de ativação (KRIZHEVSKY et al. 2012) (FLÁVIO et al. 2017).

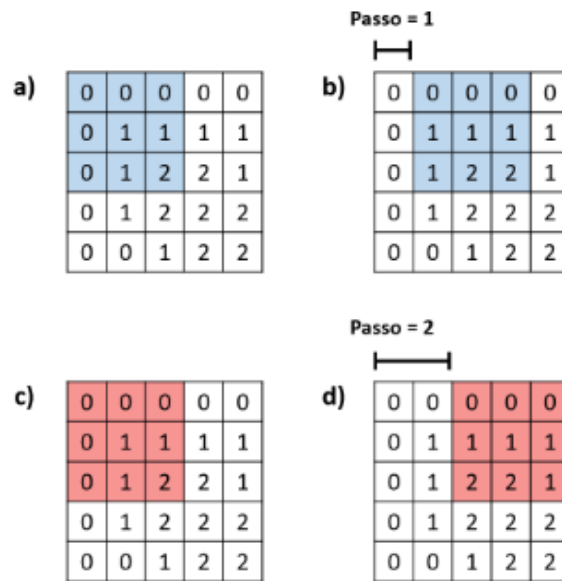
Figura 8: Exemplo de convolução de filtro 3x3



Fonte: Retirado do artigo de Flávio

3.4.1.1 Passo (*stride*)

Esse parâmetro especifica qual será o salto que haverá na matriz de convolução (Figura 9). Se o passo for igual a 1, significa que o filtro saltará apenas uma posição por vez, sendo igual a 2, saltará duas posições por vez, assim por diante. Quanto maior o salto, maior será a perda de informações. Por esse motivo, é incomum utilizar valores maiores que 2 (FLÁVIO et al. 2017).

Figura 9: Funcionamento do parâmetro *stride*


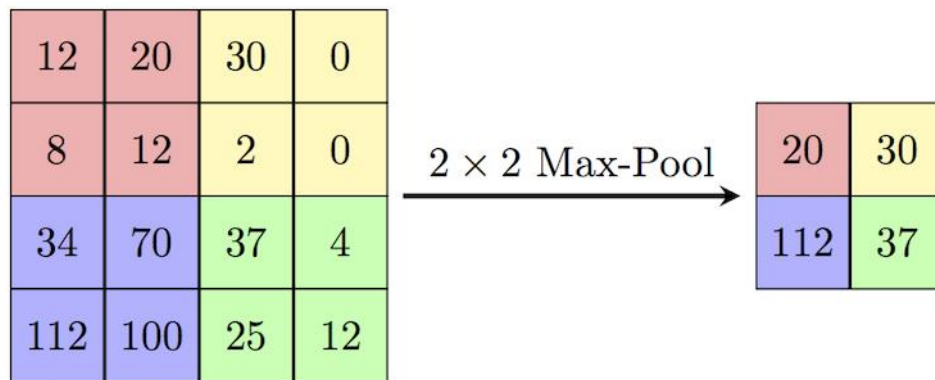
Fonte: Redes Neurais Convolucionais, por Flávio et al.

3.4.1.2 Preenchimento (*padding*)

Durante o processo de convolução, há a necessidade de lidar com as bordas das imagens, as mais comuns são: *valid padding* e *same padding*. Miyazaki (2017), no processo do *valid padding* as bordas do filtro não ultrapassam as bordas das imagens. Agindo de forma contrária, o *same padding* preenche as bordas com 0 buscando controlar a altura e largura na camada de saída.

3.4.1.3 Camadas de Pooling

Para Karpathy (2017), as camadas de *pooling* tem como objetivo reduzir de forma progressiva a dimensão espacial dos dados de entrada, o que acarreta na diminuição do custo computacional e auxilia para evitar o *overfitting*. Na operação de *pooling*, os valores relativos da região do mapa de atributos oriundos das operações convolucionais, são então substituídos por alguma métrica. Segundo Goodfellow et al. (2016) a metodologia utilizada comumente consiste em substituir os valores da região pelo valor máximo, essa operação é conhecida como “max pooling” (Figura 10). Tem como utilidade eliminar valores insignificantes, além de reduzir a dimensão dos dados de entrada e acelerar a computação das próximas camadas.

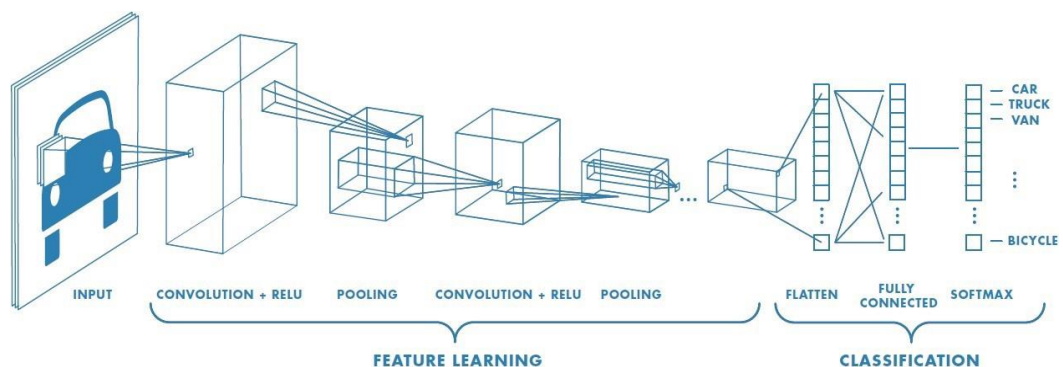
Figura 10: Operação do *Max Pooling* em imagem 4x4


Fonte: Retirado do site Computer Science Wiki

3.4.1.4 Camadas *Dense* (*Fully Connected Layer*)

Essa camada recebe este nome pois todos os neurônios da camada anterior estão conectados com os neurônios desta camada. As características que foram extraídas das camadas de convolução e de *pooling* chegam nessas camadas onde então são classificadas e através de alguma função de ativação (geralmente são utilizados *softmax* e *sigmoid*), podemos prever as classes dos dados de entrada (MIYAZAKI, 2017).

Figura 11: Arquitetura de rede neural convolucional para classificação de imagens



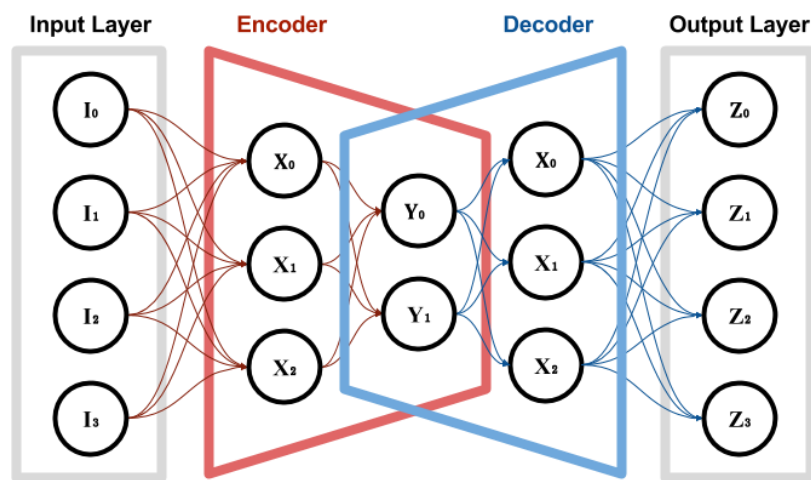
Fonte: Towards Data Science

3.5 AUTOENCODER

Os *autoencoders* são uma arquitetura de rede neural que é treinada justamente para tentar copiar seu dado de entrada para sua saída, ou seja, no caso da utilização de imagens, a saída da rede deve ser a melhor reconstrução possível das imagens da entrada.

Internamente há uma camada oculta (*hidden layer*) que contém a representação dos dados de entrada. São projetados para serem incapazes de aprender a copiar perfeitamente, são restritos de maneira que permitem copiar somente o que melhor se assemelha aos dados de treinamento. O algoritmo é forçado a priorizar quais aspectos da entrada devem ser copiados, que geralmente aprende os parâmetros úteis dos dados (GOODFELLOW; BENGIO: COURVILLE, 2016).

Figura 12: Exemplo de *Autoencoder*



Fonte: “An Introduction to Neural Networks and Autoencoder” - Alan Zucconi

4 IMPLEMENTAÇÃO

Neste capítulo serão descritos como foram feitos os procedimentos para realização desta pesquisa, tais como os materiais, os métodos e solução do problema apresentado.

4.1 MATERIAIS

O pré-processamento, a rede neural convolucional e todo o desenvolvimento foram feitos em Python, com a utilização de bibliotecas *Numpy* e *Pandas* para processamento e *Keras* e *TensorFlow* para aprendizado de máquina. Os testes foram devidamente executados em um computador equipado com CPU Intel® Core™ i7-8700K @ 3.7 GHz, com 16GB de memória RAM e GPU NVIDIA® GeForce GTX 1080 Ti com 12GB VRAM.

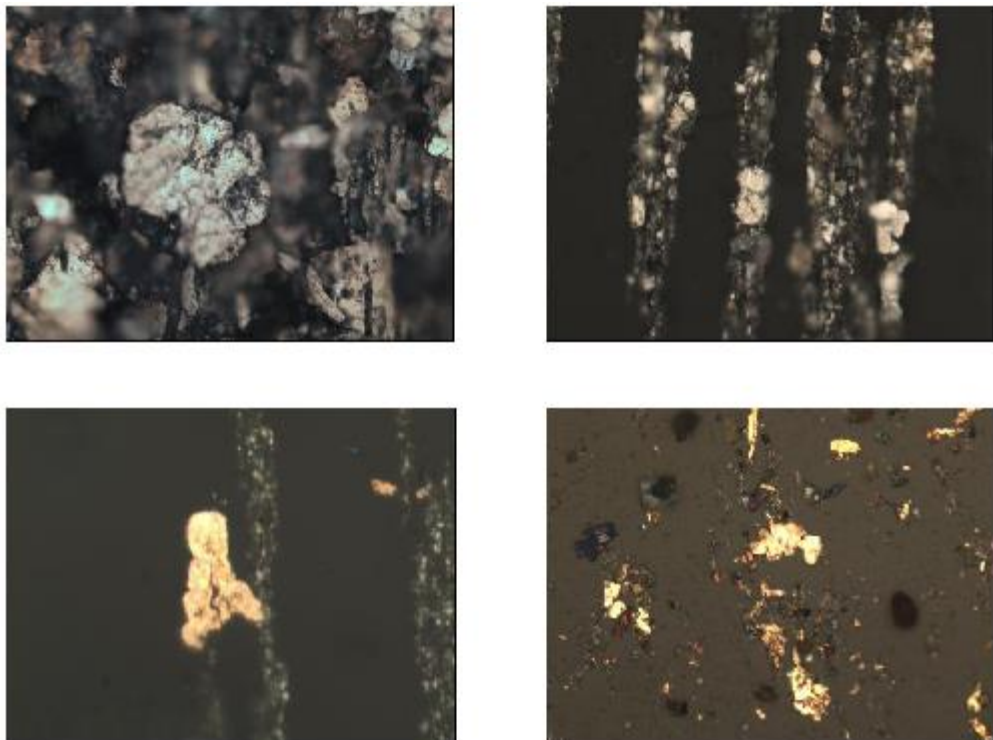
4.1.1 Keras e TensorFlow

O Tensorflow é uma *framework* para o desenvolvimento de *Machine Learning* desenvolvido e disponibilizado pela *Google*. O *Keras* também é uma *framework* que pode ser implementada juntamente com o *TensorFlow*, *Theano* entre outros. Apesar do *TensorFlow* possuir maior capacidade de adaptação e manipulação, o *Keras* é mais amigável para iniciantes em *ML*, sua simplicidade o torna mais fácil de usar.

4.1.2 Base de Dados (*Dataset*)

O *dataset* utilizado foi cedido pela empresa SEMEQ - Serviços de Monitoramento de Equipamentos. A base de dados contém 9000 diferentes imagens microscópicas de amostras de óleo (Figura 12).

Figura 13: Exemplos de amostra de óleo



Fonte: Elaborado pelo autor


4.2 PRÉ-PROCESSAMENTO

4.2.1 Normalização

Buscando minimizar possíveis problemas envolvendo instabilidades numéricas dentro da rede neural, é essencial fazer a normalização dos dados. No caso das imagens que são representadas em matrizes com os valores de seus pixels (a partir de 0 até 255), é comum realizar a divisão de todos por 255, fazendo com que a rede trabalhe com valores entre 0 e 1, isso gera facilidade para a generalização da mesma.

Figura 14: Normalização em uma imagem 4x4

255	150	0.0	0.0
180	240	180	180
240	150	255	35
240	35	255	255



1.0	0.58	0.0	0.0
0.70	0.94	0.70	0.70
0.94	0.58	1.0	0.13
0.94	0.13	1.0	1.0

Fonte: Elaborado pelo autor

4.2.1.1 Conversão e separação dos dados

Para a utilização das imagens foi necessário a conversão desses objetos em *arrays*, através da biblioteca *Numpy*. Onde, para cada imagem, foi criado um *array* contendo os valores respectivos de cada pixel.

A partir disso, é possível separar os dados em dois conjuntos, um para treino e outro para teste. Para isso foi utilizado a biblioteca *scikit-learn*, com o método “*train_test_split*”, que separou os dados em 75% para treino e o restante para teste.

4.2.2 Codificação One-hot

Sabendo que as redes neurais não podem operar diretamente com rótulos, a codificação one-hot tem o papel de transformar os rótulos em valores binários. Nesse caso de classificação, temos 13 classes diferentes, descritas abaixo juntamente com seu vetor binário, respectivamente.

Tabela 1: Codificação One-hot das classes

Par. corrosivas	1	0	0	0	0	0	0	0	0	0	0	0	0
Fibras	0	1	0	0	0	0	0	0	0	0	0	0	0
Esferas	0	0	1	0	0	0	0	0	0	0	0	0	0
Óxido preto	0	0	0	1	0	0	0	0	0	0	0	0	0
Óxido verm.	0	0	0	0	1	0	0	0	0	0	0	0	0
Arrancamento	0	0	0	0	0	1	0	0	0	0	0	0	0
Mancal	0	0	0	0	0	0	1	0	0	0	0	0	0
Deslizamento	0	0	0	0	0	0	0	1	0	0	0	0	0
Atrito normal	0	0	0	0	0	0	0	0	1	0	0	0	0
Esferas cont.	0	0	0	0	0	0	0	0	0	1	0	0	0
Areia / Sujeira	0	0	0	0	0	0	0	0	0	0	1	0	0
Engrenagem	0	0	0	0	0	0	0	0	0	0	0	1	0
Lubrificante	0	0	0	0	0	0	0	0	0	0	0	0	1

Fonte: Elaborado pelo autor

4.3 TREINAMENTO DA CNN

O aperfeiçoamento de uma rede neural é um processo experimental, pois não há padrões, a topologia varia de acordo com o problema a ser solucionado, por tanto foram testados as mais variadas abordagens, buscando o melhor resultado.

Podemos definir então a quantidade de filtros convolucionais, os *strides*, tamanho do *Max Pooling*, quantidade de épocas e *Dropout*, assim como todos os parâmetros abordados anteriormente. Foram exploradas várias topologias diferentes ao decorrer dos testes.

Foram definidos, de forma fixa, alguns parâmetros para treinamento:

1. Stride da camada convolucional = 1;
2. Taxa de aprendizagem = 1×10^{-3} (0,003);
3. Tamanho do filtro Max Pooling = 2×2 ;
4. Número de épocas = 5000;
5. Dropout = 0.3;
6. Quantidade de mapa de características:
 - a. para 2 camadas de convolução:
 - (i) Camada 1 = 4
 - (ii) Camada 2 = 8
 - b) para 3 camadas de convolução:
 - (i) Camada 1 = 4

(ii) Camada 2 = 8

(iii) Camada 3 = 16

5 TESTES E RESULTADOS

A implementação foi feita com a linguagem de programação Python, com o auxílio das bibliotecas *Numpy*, *scikit-learn*. Foi utilizado a *framework* Keras, uma ferramenta voltada para o treinamento de redes neurais artificiais que trabalha com o *TensorFlow* no *backend*.

Para alcançar os resultados esperados, foi implementado para teste, *Autoencoder* e Rede Neural Convolutiva juntamente com Camadas Totalmente Conectadas.

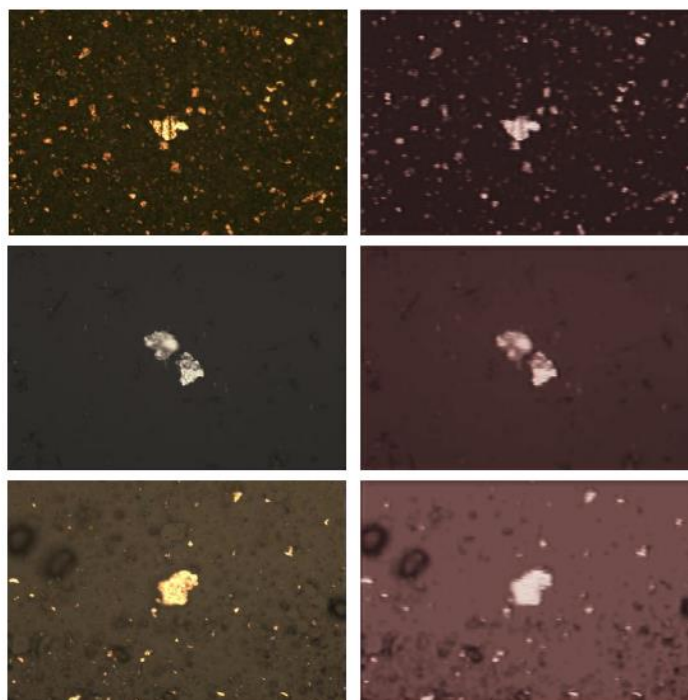
5.1 Problemas e balanceamento das classes

Houve dificuldades ao alcançar o resultado durante o treinamento no momento em que se notou grande desbalanceamento das classes presentes no *dataset*, onde, diante das treze classes, seis delas estava totalmente zeradas, ou seja, nem todas as classes estavam presentes nas imagens. Diante deste problema, foi necessária a implementação de algoritmo para realização de *Class Weight* (peso das classes) que é responsável por fazer a iteração dentro da tabela das *labels* (rótulos) e calcular um peso de cada classe que é adicionado ao treinamento.

5.2 Treinamento e teste com *Autoencoder*

O autoencoder é alimentado com os *arrays* das imagens de resolução 300 x 400 pixels. Em cada iteração, há um processo de compressão das imagens, isso ocorre até a camada intermediária, que nesse caso, chegou a resolução de 75 x 100 pixels. Na figura 15 vemos a reconstrução gerada pelo Autoencoder que, neste caso, as imagens na esquerda representam as imagens originais de entrada, e na direita, sua respectiva reconstrução.

Figura 15: Reconstrução gerada pelo Autoencoder



Fonte: Próprio autor

Alguns resultados obtidos com os menores valores de perda da validação no treinamento do *autoencoder*:

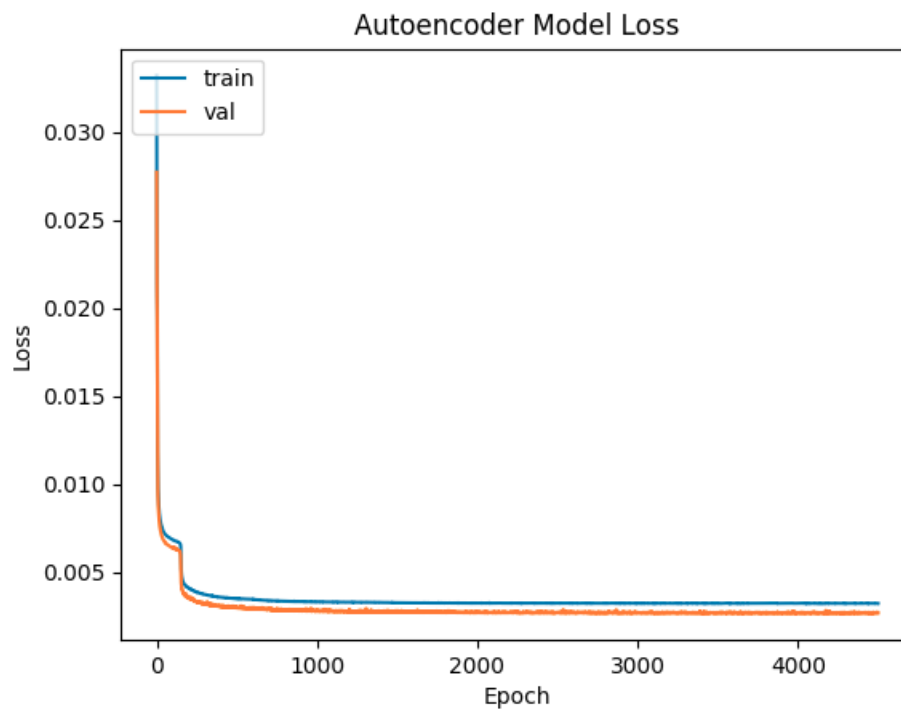
Tabela 2: Resultado de treino do *Autoencoder*

Otimizador	Função de custo	Custo de validação	Épocas
Adam	<i>Mean Squared Error</i>	0.00118	1000
Adagrad	<i>Mean Squared Error</i>	0.03445	1000
Adadelta	<i>Mean Squared Error</i>	0.00969	1000

Fonte: Próprio autor

Na figura 16 podemos ver o histórico referente à diminuição da função de custo no processo de validação, onde o Autoencoder foi treinado por mais de 4 mil épocas e a função de custo decaiu até aproximadamente 0.002.

Mesmo com o ótimo resultado no treinamento do Autoencoder, não foi implementado na versão final, pois não houve acréscimo de desempenho.

Figura 16: Função de custo da validação do *Autoencoder*


Fonte: Próprio autor

5.3 Treinamento e Teste da Rede Neural Convolutacional

Mesmo implementando o *Class Weight* mostrado anteriormente, o algoritmo não foi capaz de generalizar para as classes nulas, porém foi possível realizar o treinamento.

Seguindo a Tabela 3, é possível notar a variação de desempenho fazendo a alteração no número de camadas de convolução e o tamanho do filtro e fixando o valor de uma camada Totalmente Conectada.

Tabela 3: Desempenho variando as camadas de convolução

		Convolução = 2		Convolução = 3	
		3x3	5x5	3x3	5x5
Acurácia	Treino	0,9128	0,9345	0,9055	0,9052
	Teste	0,9013	0,9026	0,9006	0,9019
Custo	Treino	0,1073	0,0925	0,1135	0,1106
	Teste	0,2687	0,3282	0,2394	0,2378

Na próxima tabela, os testes foram feitos variando o número de Camadas Totalmente Conectadas (em inglês – *Fully Connected Layer*), mantendo o valor de 3x3 para o tamanho da janela de convolução.

Tabela 4: Desempenho variando as camadas FC

		FC = 1		FC = 2	
		Conv = 2	Conv = 3	Conv = 2	Conv = 3
Acurácia	Treino	0,9106	0,9049	0,9128	0,9055
	Teste	0,8974	0,8994	0,9013	0,9006
Custo	Treino	0,0967	0,1085	0,1073	0,1135
	Teste	0,2453	0,2359	0,2687	0,2394

Fonte: Desenvolvido pelo autor

6 CONCLUSÃO

A Inteligência Artificial está presente em todas as áreas de atuação profissional, sendo o seu objetivo, produzir modelos capazes de automatizar e melhorar o desempenho de computadores, a fim de auxiliar a atividade humana. Com a crescente demanda e pesquisa na área, foi possível notar que é praticável a implementação em manutenção preditiva, que neste caso, através da classificação de imagens. Este trabalho demonstrou a importância dos processos de integração da tecnologia, com o aprendizado de máquina e rede neural convolucional.

Posto que houve o problema referente à falta de classes, como descrito anteriormente, a implementação da CNN trouxe resultados que podemos considerar válidos, e ressaltamos ainda que a variação de 2 ou 3 camadas de convolução não alterou de forma notável a precisão do modelo, e que, o desbalanceamento das classes, impactou negativamente o desempenho do modelo como um todo. A partir disto, podemos validar a contribuição científica com a produção desta pesquisa, que oferece um estudo base para futuros desenvolvimentos e mostra como o aprendizado de máquina se comporta de forma positiva quando introduzido à determinado problema.

6.1 Trabalhos futuros

A pesquisa tinha como objetivo a criação e validação de um modelo de aprendizado de máquina e implementar dentro de manutenção preditiva, além de explicar o funcionamento de redes neurais artificiais. Sabendo que as redes neurais de convolução são ferramentas para extração de características, podemos concluir que é possível melhorar o modelo além de 90% de acurácia, reduzindo ainda mais o custo de validação, corrigindo o problema de balanceamento de classes e realizando novos testes.

Sendo assim, segue alguns tópicos que devem ser abordados para possíveis trabalhos futuros:

- Testes como uma nova base de dados;
- Aumento artificial das imagens, através do uso da técnica *Data Augmentation*;
- Implementar técnicas para redução e controle de sobre-ajuste (overfitting), que seriam a diminuição automática da taxa de aprendizagem, normalização L2, etc.

Buscando auxiliar ainda mais os profissionais que trabalham com manutenção preditiva, é válido a implementação de um relatório final pós classificação das imagens, onde este relatório contenha possíveis ações que precisam ser tomadas diante as falhas encontradas nas amostras.

REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO, Flávio et al. **Redes Neurais Convolucionais com TensorFlow: Teoria e Prática**. 2017. Artigos e Minicursos - Capítulo 7. Disponível em <<http://www.eripi.com.br/2017/images/anais/minicursos/7.pdf>>.

AZEVEDO, C. H. B. **Metodologia para a eficácia da detecção de descargas parciais por emissão acústica como técnica preditiva de manutenção em transformadores de potência imersos em óleo isolante**. 2009. 92f. Tese (Mestrado em Engenharia Elétrica e Computação) - Universidade Federal de Goiás, Goiânia, 2009.

COSTA, Augusto H. et al. **Manutenção Preditiva**. Disponível em <http://wwwp.feb.unesp.br/jcandido/manutencao/Grupo_8.pdf>.

FERNEDA, Edberto. **Redes neurais e sua aplicação em sistemas de recuperação de informação**. Disponível em <<http://www.scielo.br/pdf/ci/v35n1/v35n1a03.pdf>>.

FERREIRA, A. E. T. **Estimação do ângulo de direção por vídeo para veículos autônomos utilizando redes neurais convolucionais multicamadas**. 2017. Trabalho de Conclusão de Curso (Graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. Disponível em: <<http://www.deeplearningbook.org>>.

HAFEMANN, L. G. **An analysis of deep neural networks for texture classification**. 2014. Master's degree Dissertation. M. Sc. Program in Informatics, Universidade Federal do Paraná.

KARPATHY, A. **Convolutional neural networks for visual recognition**. 2017. Disponível em <<http://cs231n.github.io/convolutional-networks/>>.

KARPATHY, A. **Convolutional neural networks**. 2014. Disponível em <<http://andrew.gibiansky.com/blog/machine-learning/convolutional-neural-networks/>>.

KRISHNA. **Introduction to Exponential Linear Unit**. Disponível em: <<https://medium.com/@krishnakalyan3/introduction-to-exponential-linear-unit-d3e2904b366c>>

MARTINS, Ronilson de Carvalho et al. **Proposta de uma nova metodologia aplicada a manutenção preditiva baseada na análise de óleo lubrificante**. Disponível em <http://www.conemi.org.br/download/TT11_XIV_CONEMI-003.pdf>.

Max-pooling / Pooling. Computer Science Wiki. Disponível em <https://computersciencewiki.org/index.php/Max-pooling/_Pooling>.

MIYAZAKI, C. K. **Redes neurais convolucionais para aprendizagem e reconhecimento de objetos 3D**. Disponível em: <http://www.tcc.sc.usp.br/tce/disponiveis/18/180500/tce-22022018-121624/publico/Miyazaki_caio_tcc.pdf>.

MONTEIRO, Caio Italiano et al. **Manutenção Corretiva**. Disponível em <http://wwwp.feb.unesp.br/jcandido/manutencao/Grupo_6.pdf>.

NORVIG, P.; RUSSELL, S. **Artificial Intelligence: A modern approach**. Artificial Intelligence. Prentice-Hall, Englewood Cliffs, 1995.

O que é neurônio?. Brasil Escola. Disponível em: <<https://brasilescola.uol.com.br/o-que-e/biologia/o-que-e-neuronio.htm#>>>.

ODO, Fábio. **A importância da filtragem**. Disponível em <<http://www.fluidcontrol.com.br/library/pdf/importanciadafiltragem.pdf>>.

PONTI, A. Moacir; COSTA. G. B. P. **Como funciona o Deep Learning**. Disponível em: <http://conteudo.icmc.usp.br/pessoas/moacir/papers/Ponti_Costa_Como-funciona-o-Deep-Learning_2017.pdf>.

SHIRURU, Kuldeep. (2016). **An introduction to Artificial Neural Network**. International Journal of Advance Research and Innovative Ideas in Education. 1. 27-30.

SOUZA, Alien Vlganô et al. **Qualidade da mão de obra na manutenção**. Disponível em <http://wwwp.feb.unesp.br/jcandido/manutencao/Grupo_5.pdf>.

TURING, A. M. (1951). **Can digital computers think?**. Disponível em: <<http://www.turingarchive.org/browse.php/B/5>>.

TURING, A. M. (1950). **Computing Machinery and Intelligence**. Disponível em: <<https://academic.oup.com/mind/article/LIX/236/433/986238>>.

WEHLE, Hans-Dieter. (2017). **Machine Learning, Deep Learning, and AI: What's the Difference?**.

ZUCCONI, Alan. **An Introduction to Autoencoders**. Disponível em: <<https://www.alanzucconi.com/2018/03/14/an-introduction-to-autoencoders/>>.