UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE

CAMPUS AVANÇADO DE NATAL

DEPARTAMENTO DE COMPUTAÇÃO

CURSO DE CIÊNCIA DA COMPUTAÇÃO

RONALDO BARROS DA COSTA JUNIOR

REDE NEURAL ARTIFICIAL PROFUNDA PARA DETECÇÃO DE DEGRAUS

NATAL – RN

2018

RONALDO BARROS DA COSTA JUNIOR

REDE NEURAL ARTIFICIAL PROFUNDA PARA DETECÇÃO DE DEGRAUS

Monografia apresentada à Universidade do Estado do Rio Grande do Norte Como um dos pré-requisitos obrigatórios para obtenção do título de bacharel em Ciência da Computação.

Orientador: Wilfredo Blanco Figuerola

NATAL – RN
2018

Catalogação da Publicação na Fonte.

Universidade do Estado do Rio Grande do Norte.

C837r Costa Junior, Ronaldo Barros da

REDE NEURAL ARTIFICIAL PROFUNDA PARA DETECÇÃO DE DEGRAUS. / Ronaldo Barros da Costa Junior. - NATAL, 2018.

60p.

Orientador(a): Prof. Dr. Wilfredo Blanco Figuerola.

Monografia (Graduação em Ciência de Computação). Universidade do Estado do Rio Grande do Norte.

1. Deficiência Visual. 2. Redes Neurais. 3. Aprendizado profundo. 4. Reconhecimento de Degraus. I. Figuerola, Wilfredo Blanco. II. Universidade do Estado do Rio Grande do Norte. III. Título.

O serviço de Geração Automática de Ficha Catalográfica para Trabalhos de Conclusão de Curso (TCC´s) foi desenvolvido pela Diretoria de Informatização (DINF), sob orientação dos bibliotecários do SIB-UERN, para ser adaptado às necessidades da comunidade acadêmica UERN.

RONALDO BARROS DA COSTA JUNIOR

REDE NEURAL ARTIFICIAL PROFUNDA PARA DETECÇÃO DE DEGRAUS

Monografia apresentada à Universidade do Estado do Rio Grande do Norte como um dos pré-requisitos obrigatórios para obtenção do título de bacharel em Ciência da Computação

Aprovado em ____/____/____.

Banca Examinadora

_____

Prof. Dr. Wilfredo Blanco Figuerola
Universidade do Estado do Rio Grande do Norte

_____

Prof. Dr. Carlos André Guerra Fonseca
Universidade do Estado do Rio Grande do Norte

_____

Prof. Me. André Gustavo Pereira da Silva
Universidade do Estado do Rio Grande do Norte

Natal/RN

2018

A todos que tiveram participação direta e indireta na minha formação acadêmica, em especial a Deus e minha Família, por estarem sempre ao meu lado.

AGRADECIMENTOS

Tenho muito a agradecer a todos que fizeram com que a conclusão do processo de graduação se tornasse possível. Principalmente aos que estiveram próximos de mim durante o período de aproximadamente quatro anos. O meu pai Ronaldo Barros da Costa, que sem dúvida foi de muito incentivo para mim. A minha mãe Sandra Helena Lima da Costa, que esteve sempre ao meu lado me ajudando emocionalmente e sem dúvida foi indispensável na minha formação. Aos meus irmãos Richardson Lima da Costa e Helensandra Lima da Costa pela companhia e ajuda moral.

Não poderia esquecer a ajuda de grandes amigos que estiveram sempre ao meu lado nesses momentos. A Thâmara Geanny Sousa Oliveira pela ajuda em momentos tão difíceis, em especial nesse último ano. E a toda minha família na fé por estarem sempre ao meu lado.

Na universidade foi possível perceber que realmente a união faz a força, por isso tenho muito a agradecer a meus companheiros diários de turma: Paulos Sérgio Moura de Oliveira e Daniel Ferreira Costa. Com a ajuda de todos vocês esse processo se tornou possível. Aos professores e ao Prof. Dr. Wilfredo Blanco Figuerola pelas orientações e compreensão.

“Eu proponho... que nós façamos um robô capaz... de amar.”

(William Hurt, A.I, (2001))

RESUMO

A tecnologia tem se mostrado uma aliada dos portadores de necessidades especiais, especificamente para os deficientes visuais. Nesse contexto, este trabalho apresenta um sistema computacional inteligente para detectar degraus os quais são filmados com uma câmera. Uma rede neural artificial profunda com aprendizado supervisionado, foi treinada com um conjunto de imagens (frames de vídeos) contento rótulos (área retangular) com a localização dos degraus. As imagens de treinamento foram escolhidas para detectar situações diversas como: ambientes com luminosidade variada e diversos ângulos da câmera. Para manipular os frames/imagens utilizamos a biblioteca OpenCv; e para executar e avaliar do processo de treinamento foi utilizado o framework da *Tensorflow.* Na fase de teste, o sistema mostrou eficiência ao detectar e marcar com precisão os degraus nas situações previamente mencionadas. Para imagens capturadas em ambientes com maior luminosidade, a determinação dos degraus foi com mais acurácia.

Palavras-chave: Deficiência Visual, Redes Neurais Artificiais, Aprendizado Profundo, Reconhecimento de Degraus.

ABSTRACT

Technology has proven to be an ally of people with special needs, specifically for the visually impaired. In this context, this work presents a computational intelligence system for detecting steps that are filmed with a camera. A deep artificial neural network with supervised learning was trained with a set of images (video frames) containing labels (rectangular area) with the location of the steps. The training images were chosen to detect diverse situations like: environments with varied luminosity and camera views from several angles. To manipulate the frames/images we used the OpenCv library; and to execute and evaluate the training process the Tensorflow framework was used. In the test phase, the system showed efficiency and accurately detecting and marking the steps in the previously mentioned situations. For images captured in higher light conditions, the determination of the steps was more accurate.

Keywords: Visual impairment, Artificial Neural Network, Deep Learning, Detecting Steps.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIAÇÕES

OMS	Organização Mundial da Saúde
DL	*Deep learning*
CNN	*Convolutional Neural Network*
IA	Inteligência artificial
GPUS	Unidades de processamento gráfico
3d	Espaço Tridimensional
2d	Espaço Bidimensional
AM	Aprendizado de máquina
RNA's	Redes neurais artificiais
ReLU	*Rectified linear units*
CV	Computação visual
TMC	Topologia de Múltiplas Camadas

SUMÁRIO

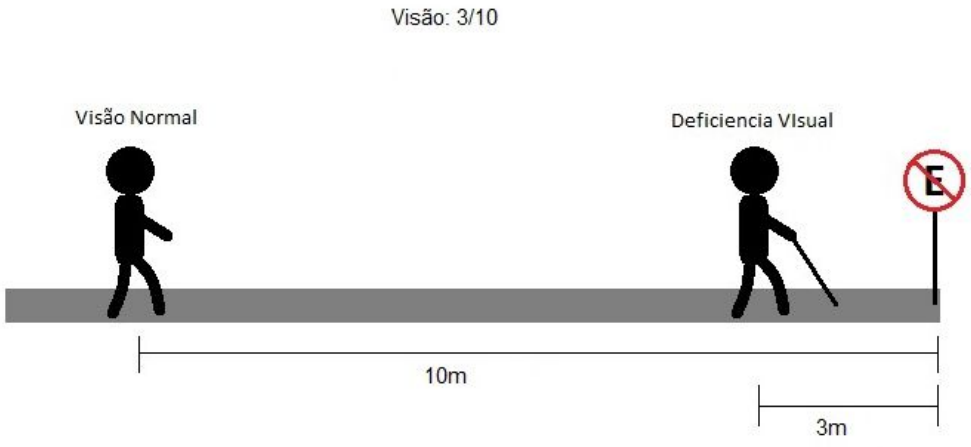
1 INTRODUÇÃO

Grande parte das atividades diárias realizadas pelos seres humanos requerem a utilização da visão; consequentemente, quando uma pessoa apresenta deficiência visual, ela provavelmente terá limitações para a execução das mesmas, prejudicando a inserção social de seus portadores, principalmente durante a sua locomoção. Por este motivo, existe um grande esforço para minimizar as consequências que esta limitação traz para a pessoa no seu convívio social.

Para definir se alguém é considerado deficiente visual, são utilizados dois parâmetros principais: A acuidade que é a capacidade de discernir dois pontos à uma determinada distância, e/ou o campo visual que é a amplitude do espaço percebido pela visão.

Com a visão do melhor olho corrigida, a acuidade visual deve ser menor ou igual a 3/10 para ser considerado deficiente visual, alguém com essa acuidade visual, tem limitação de ver a 3m o que um indivíduo com visão total vê a 10m, além disso o campo de visão deve possuir medida inferior a 20º de arco, levando em consideração o eixo de visão (Taleb *et al.*, 2013). Na Figura 1 é possível observar que alguém com visão normal consegue enxergar a placa à uma distância de 10m, enquanto o deficiente visual enxerga a 3m.

Figura 1 Acuidade visual 3/10.



Fonte: Autor

Segundo a Organização Mundial da Saúde (OMS), a deficiência visual é dividida em 5 estágios, sendo as pessoas que possuem os sintomas dos estágios 1 ou 3, consideradas deficientes visuais com baixa visão. Já as pessoas com sintomas de 4 ou 5 são consideradas cegas (Taleb *et al.*, 2013). Todos esses estágios são definidos na Tabela 1.

Tabela 1 Caracterização dos níveis de deficiência visual

Classificação Da Deficiência visual		Acuidade visual com a melhor correção possível	
		Máximo inferior à	Mínimo igual ou melhor que
Baixa visão	1	3/10	1/10
	2	1/10	1/20
	3	1/20	1/50
Cegueira	4	1/50	Percepção de luz
	5	Sem percepção de Luz	

Fonte: (Taleb *et al.*, 2013)

A tecnologia tem sido um aspecto importante para aliviar os efeitos da deficiência visual. Com o desenvolvimento da tecnologia, uma área em especial, chamada de visão computacional, tem se destacado nesse cenário. Essa é uma área da inteligência artificial que consiste em capturar cenas, e a partir delas obter informações das características da imagem, analisando cada pixel da mesma.

A utilização desta tecnologia pode trazer uma qualidade de vida melhor e auxiliar os deficientes visuais em tarefas como a locomoção e identificação de obstáculos e objetos. Porém, a visão computacional, apenas, não é o suficiente. Os sistemas precisariam aprender a partir dos dados retirados das imagens, para que possam funcionar com uma precisão maior e oferecer um alivio dos efeitos da deficiência visual.

As redes neurais se enquadram nesse contexto e por isso podem trazer benefícios para essas pessoas. Com seu grande poder de processamento, as redes neurais são definidas como técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência. Uma dessas técnicas é a *deep learning* (em português, aprendizagem profundo), que consiste em utilizar redes neurais para

aprender uma representação de um determinado tipo de informação. Uma das possibilidades para essa técnica é a percepção, ou seja, a possibilidade de identificar padrões em cenas capturadas. Sendo assim, a utilização da técnica de aprendizado profundo é uma alternativa no estudo de visão computacional.

1.1 MOTIVAÇÃO E JUSTIFICATIVA

Segundo a OMS, em 2010 existiam cerca de 285 milhões de deficientes visuais ao redor do mundo, sendo 246 milhões com baixa visão e 39 milhões com cegueira, número esse que tem crescido ano após ano (Abner *et al.*, 2010). Por se tratar de um número considerável, essa pesquisa deseja mostrar a necessidade de proporcionar uma melhor qualidade de vida para estas pessoas, que infelizmente possuem limitações com relação a tarefas diárias, principalmente as que envolvem locomoção. Esse estudo visa trazer aos deficientes visuais a possibilidade de serem auxiliados na identificação de degraus no terreno da sua trajetória, tornando a tarefa da percepção desses degraus mais simples para eles.

A realidade atual é que, para se locomoverem com maior mobilidade, muitos dos deficientes visuais precisam de bengalas ou até mesmo cães guias, que possuem um alto custo de aquisição, cerca de 30 mil reais(Globo, 2011). Além de precisarem de um longo tempo para o treinamento, os animais ainda estão sujeitos a doenças, precisam de alimentação, cuidados especiais, e como qualquer ser vivo, tem um tempo de vida. Além desses fatores, os cães guias também possuem limitações quando o assunto é locomoção, visto que eles apenas se locomovem para determinados lugares que foram pré-definidos.

Alguns estudos mostram o uso de técnicas de redes neurais em *smartphones* para identificação de objetos durante a locomoção, sendo exemplo um aplicativo para celular, *AI Scry*, que consegue descrever ações e objetos (Costa, 2016). Porém, quando se fala de locomoção, fatores como o péssimo estado de calçadas se tornam uma problemática. Esperamos que esse projeto seja o início para o desenvolvimento de uma tecnologia capaz de guiar os deficientes visuais, minimizando fatores externos que possam oferecer um impedimento para sua locomoção.

Sem dúvida, o grau de complexidade para o desenvolvimento de um guia para cegos é alto, porém se espera que um início seja construído a fim de que os deficientes visuais possam ter uma tecnologia que, dentro do possível, os ajudem nas

suas tarefas simples e essenciais como se locomover e identificar degraus. Para isso, a visão computacional junto com as redes neurais demonstram grande potencial como ferramentas e para construção de uma futura aplicação que consiga auxiliar na locomoção dos deficientes visuais.

1.2 OBJETIVOS

Focando em melhorar a qualidade de vida dos deficientes visuais, este projeto visa utilizar o campo de aprendizado de máquina, treinando uma rede neural e assim obter uma ferramenta computacional que possibilite a detecção de degraus que possam ou não oferecer riscos aos deficientes.

1.2.1 Objetivos Específicos

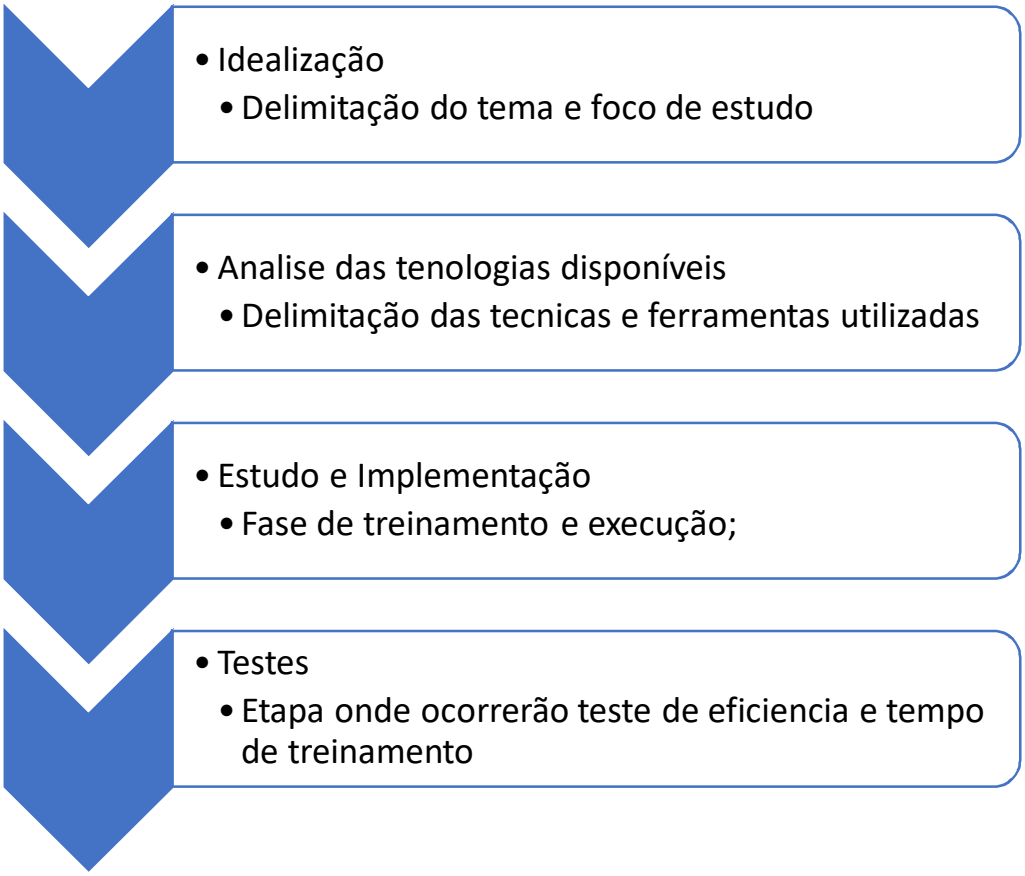
Afim de se concluir o objetivo principal deste trabalho, os seguintes objetivos foram determinados:

- Entender os conceitos de Aprendizado de Máquina;
- Estudar os conceitos de Aprendizado profundo;
- Estudar as práticas envolvidas na detecção de objetos;
- Analisar os modelos de redes utilizados para classificação;
- Estudar o modelo de rede escolhido;
- Escolher o conjunto de dados de imagens para treinar a rede escolhida;
- Rotular as imagens escolhidas;
- Treinar a rede para detectar degraus;
- Testar a rede treinada;

1.3 METODOLOGIA DA PESQUISA

Tendo em vista as dificuldades diárias dos deficientes visuais e a necessidade de uma tecnologia que os auxilie, a pesquisa tem um caráter exploratório a fim de determinar se a aplicação desenvolvida é eficiente em detectar os degraus em tempo real. Assim a pesquisa ficou dividida em quatro estágios como mostra a Figura 3: Idealização, análise das tecnologias disponíveis, estudo e implementação, e testes.

Figura 2 Diagrama da metodologia de pesquisa



Fonte: Autor

1.3.1 Idealização

A fim de se chegar a uma aplicação eficaz a pesquisa passou por um estágio de idealização, onde foi determinado o tema e foco do estudo. Nesta etapa do projeto os degraus foram escolhidos como o obstáculo principal a ser detectado. A fim de detectar os degraus em tempo real, os campos da visão computacional e das redes neurais artificiais foram escolhidas como ferramentas. Desta forma o projeto visa analisar o desempenho de uma rede neural treinada para detectar degraus.

1.3.2 Analise das tecnologias disponíveis

Tendo em vista as inúmeras arquiteturas e modelos de redes neurais, foi utilizado o conceito de aprendizado profundo (do inglês Deep Learning(DL)). Essa decisão foi tomada tendo como base alguns projetos e artigos que demonstram que o bom desempenho de redes de aprendizado profundo é, de fato, notável (Deng e Yu,

2013). Como modelo de rede foi escolhido os modelos já implementados MobileNet para o treinamento através do framework tensorflow. O modelo de rede escolhido utiliza a arquitetura Convolutional Neural Network (CNN).

Atualmente alguns frameworks estão sendo utilizados no desenvolvimento e treinamento de redes neurais com aprendizado profundo, sendo exemplos deles o *caffe deep learning framework* (Yangqing *et al.*, 2014) e o *tensorflow*(Martín *et al.*, 2015)*,* sendo escolhido o *tensorflow,* para realizar o treinamento da rede implementada com o *tensorflow*, já que possui um bom suporte com atualizações recentes e uma comunidade ativa*.*

A partir dessa decisão, foi adotada a utilização de biblioteca Opencv, essa escolha foi tomada visto que a biblioteca possui um modulo para redes neurais profundas.

1.3.3 Estudo e implementação

Nesta etapa da pesquisa, foi feita a análise das tecnologias escolhidas, a fim de se compreender o funcionamento do *framework* e as arquiteturas das redes escolhidas, bem como a criação do banco de imagens utilizadas como dados de treinamento para rede neural.

O banco de imagens é composto por cerca de 300 imagens de degraus distintos. Cada foto foi capturada em situações, horários, e ângulos diferentes. Cada foto está rotulada, como será explicado na sessão de “o sistema desenvolvido”.. O rótulo tem o objetivo de determinar em qual local da imagem utilizada para o treinamento se encontra o degrau. Com essa rotulação, foi gerado um arquivo com o mapa de rótulos que é utilizado no treinamento da rede.

Referente ao *framework Tensorflow*, foi estudado como é feito o treinamento da rede e o armazenamento dos dados do treinamento. O treinamento da rede acontecerá até que o valor de perda seja igual ou inferior a 0,005. Esse treinamento resultou em 2 arquivos que são utilizados pela aplicação desenvolvida em Python com a biblioteca OpenCV.

A aplicação é responsável por capturar as imagens do ambiente, analisar frame por frame e mostrar na tela onde se encontra o degrau. A aplicação foi testada em 3 situações distintas, sendo elas: ambientes com boa luminosidade, baixar luminosidade e rotação da câmera em relação ao degrau.

1.3.4 Testes

O primeiro teste foi realizado em ambientes com boa iluminação e em duas situações diferentes. Na primeira situação o degrau teve a tonalidade de cor variada. Dessa forma a cor da base do degrau se diferencia da cor do topo do degrau. Já na segunda situação o ambiente teve degraus que não continham uma mudança drástica na sua intensidade de cor. Objetivo desse teste é verificar a precisão da rede em detectar o degrau independentemente de sua cor.

O segundo teste foi realizado em ambientes com pouca iluminação. Nesse teste, a câmera foi apontada para degraus em que o ambiente possui iluminação desfavorável, com o objetivo de detectar se a rede continua classificando degraus mesmo em situações não propícias.

No terceiro e último teste, a aplicação foi avaliada na detecção dos degraus, mesmo com a câmera posicionada em angulações diferentes em relação ao degrau. Esse teste foi realizado com degraus diferentes e em situações adversas. Essa avaliação tem o objetivo de detectar a eficiência da rede treinada em detectar alguns degraus, em algumas situações mesmo em posições distintas.

Esses testes visam determinar a eficácia da aplicação em detectar os degraus. O resultado de cada teste será descrito a fim de validar os resultados e dessa forma, após os três testes será determinado se a aplicação atinge ou não o seu objetivo.

1.4 ESTRUTURA DO TRABALHO

Esta pesquisa está dividida em 6 capítulos, que mostram as etapas de estudo e treinamento da rede para o reconhecimento de degraus, a saber:

- No Capítulo 2 são analisados todos os conhecimentos necessários para o entendimento do projeto, tais como inteligência artificial, visão computacional, aprendizado de máquina e redes neurais.
- O Capitulo 3 descreve o processo de desenvolvimento do projeto. Neste capitulo podemos encontrar detalhes a respeito do treinamento da rede.
- O Capitulo 4 mostra testes de desempenho da rede. Nele é possível identificar a eficiência da rede e ver os resultados da detecção dos degraus.

- As considerações finais são apresentadas no Capitulo 5. Que finaliza com sugestões de trabalhos futuros.

2 REFERENCIAL TEÓRICO

Nesse capitulo será abordado o conteúdo necessário para o bom entendimento deste trabalho, dessa forma outras pesquisas que envolvam os temas analisados nessa seção, com as visões de autores sobre os temas de Inteligência Artificial, Aprendizado de Máquina, Redes Neurais, Deep Learning, Visão computacional são apresentados.

2.1 INTELIGÊNCIA ARTIFICIAL

Por volta de 1943 surge o primeiro trabalho, hoje reconhecido como IA, realizado por Warren McCulloch e Walter Pitts. Nessa pesquisa foi proposto um modelo de neurônio que alternaria entre ativado ou desativado, e que mudaria seu estágio para ativado de acordo com a intensidade dos sinais de entrada (MCCULLOCH E PITTS, 1943).

O estudo da IA continuou e por volta de 1950 surgiu o primeiro computador de rede neural, desenvolvido por Marvin Minsky e Dean Edmonds, que simulava uma rede de 40 neurônios (PETER E RUSSELL, 2013).

Alan Turing também esteve envolvido em pesquisas relacionadas a IA trazendo resultados significativos, produzindo pesquisas na área do aprendizado de máquina e algoritmos genéticos (PETER E RUSSELL, 2013). Porém todas as pesquisas e a euforia inicial reduziram significativamente nos anos seguintes, visto que os sistemas apresentaram falhas significativas ao tentar resolver problemas mais complexos.

No início da década de 80 a IA se tornou uma ciência e as pesquisas nesse ramo tem tido bons resultados (PETER E RUSSELL, 2013). Com o início dos anos 2000 o avanço das unidades de processamento gráfico (GPUS), contribuíram para o avanço da IA tornando-a capaz de resolver problemas mais complexos e mais rapidamente.

Por mais que tenha iniciado pouco antes do início da década 1950, a inteligência artificial não se encarregou de apenas armazenar e processar os dados, mas tem ganhado a incumbência de gerar informações novas, ou seja, de aprender dos dados analisados. Isso se resume na tentativa de trazer racionalidade no processamento das informações.(PETER E RUSSELL, 2013)

Em linhas gerais o estudo da IA se caracteriza por trazer a racionalidade humana para o comportamento de um sistema. Alguns autores definem a IA da seguinte forma: “[Automatização de] atividades que associamos ao pensamento humano, atividades como a tomada de decisões, a resolução de problemas, o aprendizado...” (Bellman, 1978, *apud* Peter e Russell, 2013) e “O estudo das computações que tornam possível perceber, raciocinar e agir.” (Winston, 1992, *apud* Peter e Russell, 2013).

Uma vez que é extremamente necessário atingir a racionalidade nas decisões tomadas pelo computador, se tornou paralelamente imprescindível se entender o funcionamento do cérebro humano. Com isso foi percebido que a interligação das células do cérebro (redes de neurônios) é responsável por produzir pensamentos, ações e a consciência (*apud* Peter e Russell, 2013).

Dessa forma a IA tem se tornado uma grande área de estudo que visa trazer racionalidade aos modelos computacionais e utiliza técnicas para obtenção e processamento das informações. Nesse estudo vamos analisar duas áreas da inteligência artificial, sendo elas: Visão Computacional e Aprendizado de Máquina.

2.1.1 Visão Computacional

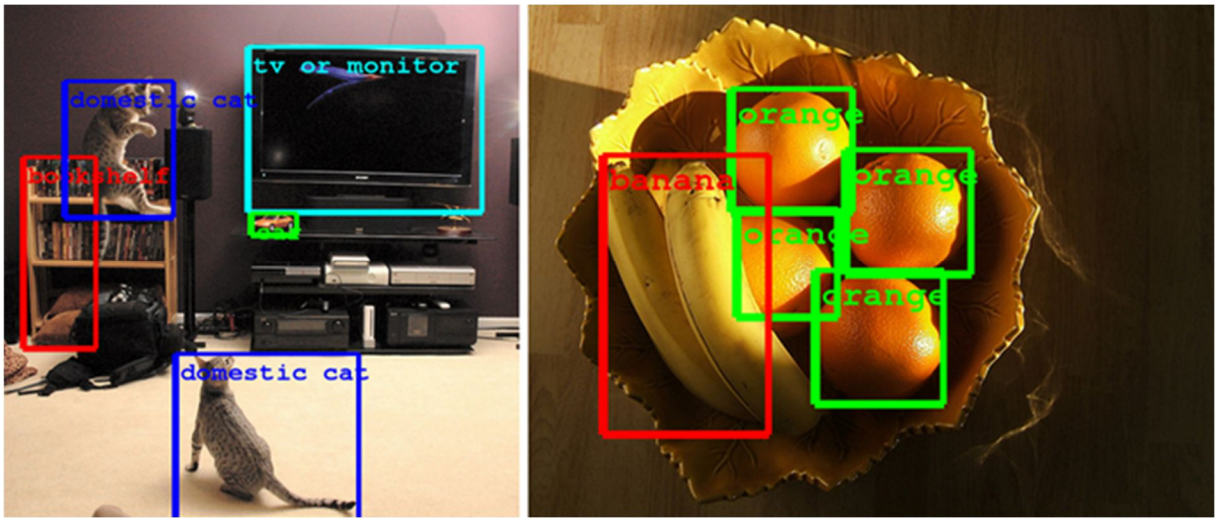
A Visão Computacional (CV) é uma das áreas da IA utilizadas para tornar os computadores mais racionais, visto que a visão é uma característica dos seres humanos. Por isso, está sendo cada vez mais comum vermos em shoppings, sistemas de segurança, nos semáforos e até mesmo nos nossos *smartphones,* aplicações responsáveis por realizar diversos tipos de detecções em imagens e vídeos.

Szeliski (2010) define visão computacional, como a tentativa de transformar o mundo real em uma imagem e reconstruir as características desse ambiente. A visão computacional também é definida como: *“*... é uma disciplina que estuda como reconstruir, interromper e compreender uma cena 3d a partir de suas imagens 2d em termos das propriedades da estrutura presente na cena” (Academy, 2017).

Uma das habilidades do sistema visual humano que mais chamam a atenção, é o reconhecimento. Com isso aplicações como reconhecimento facial em redes sociais e até mesmo reconhecimento de objetos do Google no sistema operacional mobile Android, são frutos de inúmeras pesquisas em uma área chama reconhecimento de imagem.

O reconhecimento de imagem é responsável por distinguir objetos e até mesmo expressões de sentimento como explicado por Academy (2017). A figura 3 mostra um exemplo de sistema que realiza reconhecimento de objetos.

Figura 3 Reconhecimento de objetos



Fonte: Szegedy (2014)

Para se realizar esse reconhecimento, várias técnicas de processamento digital de imagem podem ser utilizadas, dentre elas a que mais a convolução, que é uma técnica matemática muito utilizada para reconhecimento de padrões em redes profundas.

As convoluções são transformações no domínio espacial utilizadas em diversas técnicas do processamento digital de imagens, sendo exemplo delas a desfocagem e o filtro de nitidez. Uma outra função das convoluções é a detecção de características e uma imagem.

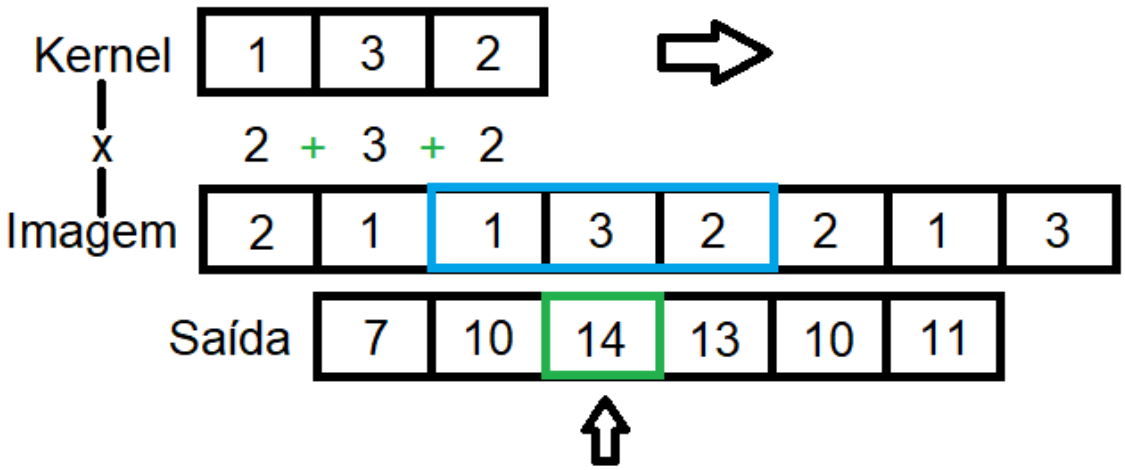
Para se realizar a detecção de características de imagens através de convoluções são necessários 2 imagens principais e uma imagem de saída: uma imagem de entrada, uma imagem de núcleo (Kernel), e uma imagem de saída que será o resultado da operação entre as duas imagens.

No que diz respeito a imagem de entrada, é aquela onde deseja-se encontrar os padrões, geralmente essa imagem é transformada em escala de cinza e após a transformação, a imagem é analisada pixel a pixel, executando operações matemáticas com o Kernel.

O Kernel é uma outra imagem de tamanho bem inferior. Essa imagem representa a característica buscada na imagem de entrada. O kernel pode ter várias dimensões, porém as dimensões mais utilizadas são 3x3 e 5x5. O processo de convolução se resume em passar o Kernel pela imagem multiplicando cada pixel do kernel pelo seu correspondente no pixel da imagem de entrada, o resultado da multiplicação é somado e todos os valores são armazenados na matriz de saída. Todo esse processo pode ser visualizado na Figura 4.

Ao realizar o processo de convolução obtém-se uma imagem de saída, onde poderemos encontrar os locais onde se encontra os padrões do Kernel. Os padrões encontrados são representados por valores, que superam a média dos pixels da imagem de saída. Dessa forma os maiores valores da imagem de saída, representam os locais da imagem de entrada onde se encontram os padrões.

Figura 4 Processo de convolução



Fonte: Autor

Como pode ser visto no simples exemplo da Figura 4, a característica do Kernel se repete na região em azul da imagem. O resultado da convolução resulta em saídas com valores mais altos na região onde se encontram as características do Kernel como está representado em verde na imagem de saída

As convoluções exercem papel fundamental na detecção de padrões em uma imagem. Como se pode perceber, o Kernel deve ser determinado previamente, visto que representa o padrão que se deseja encontrar. Dessa forma o Kernel pode ser ajustado a partir de imagens de treinamento e para isso pode-se usar uma outra área

da IA, que será responsável por aprender a partir dessas e imagens e assim ajustar o Kernel. Essa área se chama aprendizado de máquina(AM).

2.1.2 Aprendizado de máquina

Desde a década de 70 a utilização da IA tem crescido. Nas últimas décadas, os problemas têm se tornado cada vez mais complexos e com volume de dados cada vez maior. Com isso surge a necessidade de os sistemas computacionais dependerem cada vez menos da manipulação humana, para isso se torna necessário que os computadores consigam criar e aprender com base nas experiencias passadas, dessa forma surge o conceito de aprendizado de máquina. (FACELI *et al.*, 2011)

O aprendizado de máquina é uma área de IA que é responsável pelo estudo de algoritmos que sejam capazes de realizar o aprendizado e a previsão com base nos dados de treinamento, chamados de dados de entrada Um sistema de Aprendizado de máquina pode ser definido da seguinte forma: “Um sistema de aprendizado é um programa de computador que toma decisões baseado em experiências acumuladas através da solução bem sucedida de problemas anteriores”(MONARD E BARANAUSKAS, 2003).

. Dessa forma, o sistema é exposto a um conjunto de dados de entrada e tenta achar soluções gerais (modelo preditivo) a partir dos exemplos analisados. A Figura 5 mostra um esquema de arquitetura genérica de um sistema de AM. Esta etapa pode ser caracterizada da seguinte maneira:

> Assim algoritmos de AM aprendem a induzir uma função ou hipótese capaz de resolver um problema a partir de dados que representam instâncias do problema a ser resolvido. Esses dados forma um conjunto, simplesmente denominado conjunto de dados.(FACELI *et al.*, 2011)

De acordo com Faceli (2011), Sistemas de AM podem ser divididos em diferentes categorias dependendo do paradigma adotado para o sistema lidar com os novos dados. Neste contexto, o modelo se ajusta dependendo dos novos dados, e este ajuste é chamado de algoritmo de aprendizado. Se para cada instância de entrada o sistema conhece a sua saída (rótulo) desejada, o sistema AM tem um objetivo preditivo e seu algoritmo é supervisionado. Em casos onde o sistema não tem conhecimento dos rótulos das entradas, o modelo tem um papel descritivo onde o

ajuste do modelo é feito aplicando métricas que ajudam na descrição dos dados de entradas, sendo o aprendizado classificado como não supervisionado (ver esquema da Figura 6).

Figura 5 Arquitetura de um sistema de Aprendizado de máquina

Fonte: FRED (2017)

Segundo Faceli (2011), o aprendizado supervisionado se dá pela “simulação da presença de um “supervisor externo””, esse supervisor é responsável por verificar se a saída, o rótulo, é correspondente com a saída desejada para cada exemplo. Como mostra a Figura 5, cada elemento pertencente ao grupo de dados de treinamento possui um rótulo, dessa forma o algoritmo de AM é capaz de analisar se a saída corresponde ao rótulo do objeto de treino sendo, assim capaz de gerar um modelo preditivo responsável por analisar os novos dados e fazer as predições.

Como ilustra a Figura 6, o AM é dividido em dois tipos de paradigmas: supervisionado e não supervisionado. O paradigma supervisionado, ele pode ser dividido de acordo com o tipo de rótulo dos dados de treino. Se os dados são finitos, falamos de algoritmos de classificação e devem obedecer a seguinte fórmula:

$$Y = f(X) \in \{C_n, ..., C_m\} \qquad (1)$$

Dessa forma cada elemento de entrada X possui um rótulo Y correspondente, que pertence a um conjunto finito C.

Como o trabalho se enquadra nesse grupo abordando uma rede de aprendizado supervisionado, com dados de treino rotulados finitos, não será abordado o algoritmo de regressão nem o aprendizado não supervisionado.

Figura 6 Caracterização de sistemas de Aprendizado de Máquina

Fonte: Allysson Allex Araújo (2015)

2.1.3 Redes Neurais

O estudo das redes neurais surge pelo reconhecimento de que a forma como cérebro humano processa as informações é de longe mais rápido do que qualquer computador. Isso acontece pela capacidade de o cérebro processar as informações de forma não linear e paralela. Isso resulta na capacidade de processar informações sem nem mesmo nos perceber. Haykin (2001) cita o fato de o sistema visual humano realizar tarefas de reconhecimento, como por exemplo reconhecer familiares, em um tempo médio de 150ms, enquanto um computador convencional pode demorar dias para realizar tarefas mais simples.

Isso ocorre, visto que o cérebro tem a capacidade de formar suas próprias regras através da experiência, o que significa que o cérebro vai obtendo essa experiência com o tempo, e vai se ajustando afim de conseguir realizar essas tarefas com mais precisão e velocidade.

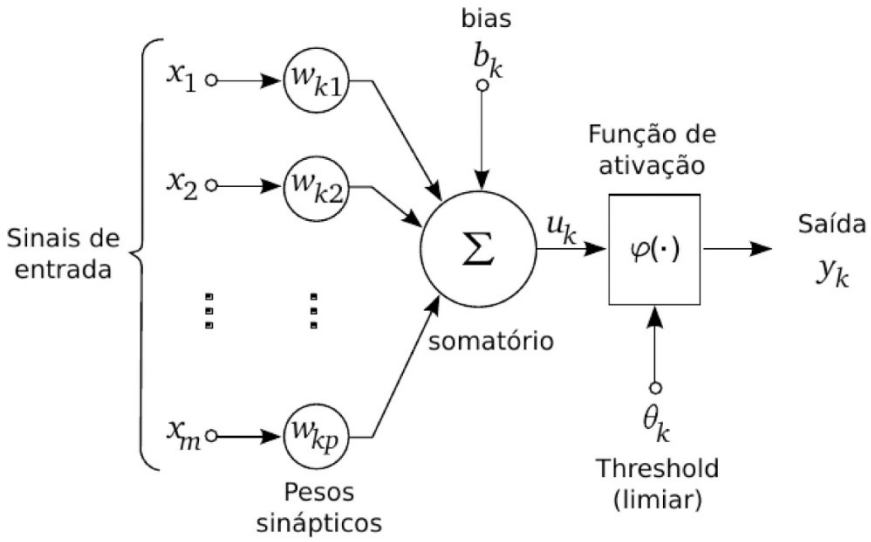
Dessa forma as redes neurais artificiais são uma tentativa de modelar esse comportamento do cérebro ao realizar as tarefas. Por isso as redes neurais, tem sido definida da seguinte forma:

> Uma rede neural é um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torna-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:
>
> 1 O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem;
>
> 2 Forças de conexão entre neurônios. Conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido.(HAYKIN, 2001)

Como mencionado na definição, as RNA'S podem ser formadas por centenas de unidades, responsáveis pelo processamento e armazenamento do conhecimento obtido. Essas unidades são chamadas de neurônios artificiais e o seu funcionamento se baseia no comportamento do neurônio biológico, que ao ser exposto a um estímulo, pode ou não ser ativado.

Cada neurônio pode possuir uma ou mais entradas, cada uma dessas conexões de entrada é chamada de sinapse. A Figura 7 mostra que cada sinapse é multiplicada por um valor de peso é feito um somatório/integração entre todas as sinapses daquele neurônio, o resultado dessa operação matemática será analisado por uma função de ativação que determinará a ativação ou não do neurônio.

Figura 7 Estrutura de um neurônio artificial



Fonte: (Rodrigues E Silva e Schimidt, 2016)

A ativação de um neurônio ocorre através de uma função denominada função de ativação ou “função restritiva” que adotam um limiar a partir do bias, para ativação do neurônio e caracteriza a saída do neurônio como podemos observar na Figura 7.(Esteu, 2014)

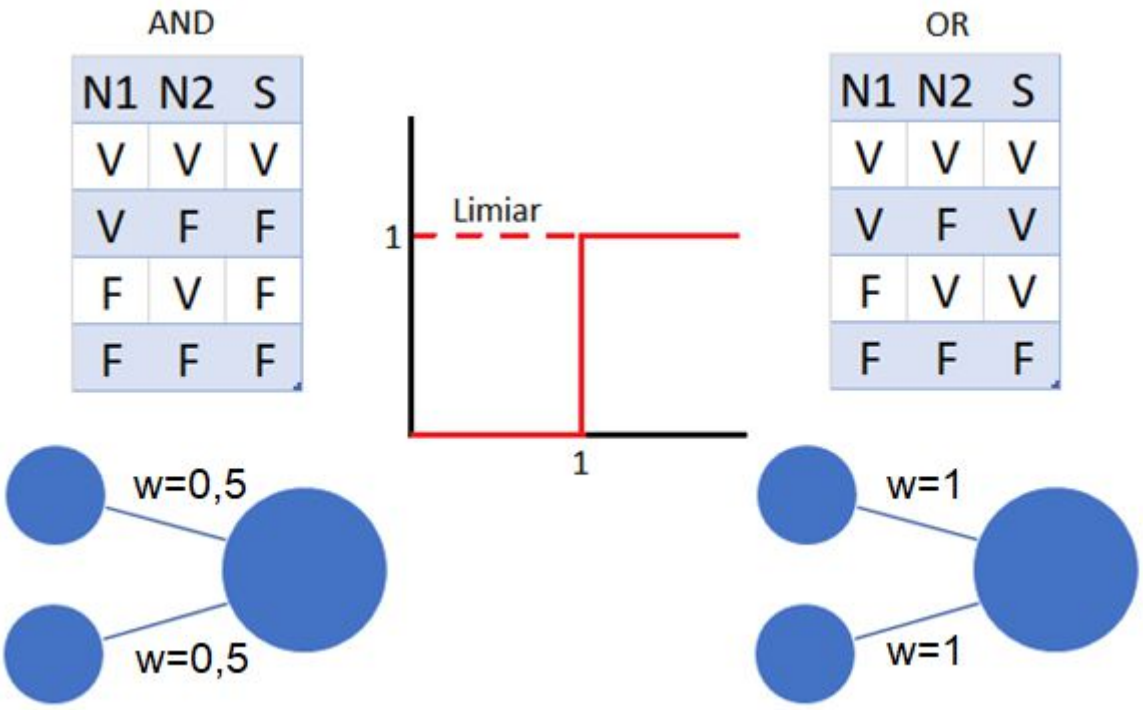
Dessa forma o neurônio artificial pode ser descrito com a seguinte formula matemática:

$$U_i = \Sigma W_{ij} * X_j \qquad (2)$$

Onde o “ U_i ” é o campo local induzido “ i “, vai ser determinado pelo somatório da multiplicação entre o estímulo “ X_i “ pelo peso sináptico ” W_{ij} “ do neurônio pré-sináptico “ $_j$ “.

O ajuste dos pesos sinápticos é responsável pelo aprendizado de uma RNA e isto acontece através do processo de treinamento da rede para obter o resultado mais próximo do esperado(Haykin, 2001).

Figura 8 Neurônios para detectar comando lógico AND e OR



Fonte: Autor

Na Figura 8 podemos observar 2 exemplos, um referente a porta lógica AND e outro para porta lógica OR. No centro observamos a função de ativação que determinará o limiar de operação para o neurônio. O que significa que o neurônio artificial será ativado a partir do momento em que o valor de entrada for igual ou maior ao seu limiar.

Considerando que na Figura 8, que a saída de um neurônio ativado, que significaria verdadeiro na tabela verdade, é representada pelo nível lógico 1, o neurônio desativado tem nível lógico 0, o que significa falso, ou seja, V corresponde a 1 e F corresponde a 0.

Ainda na Figura 8 é possível observar, a tabela verdade para uma porta lógica " E ", bem como um neurônio que fará a mesma função, ou seja, o neurônio de saída será ativado só quando as duas entradas forem iguais a 1. Caso contrário o valor de saída será falso (0).

Também consta na Figura 8 a tabela verdade para uma porta lógica " ou ", onde se pode observar que o único momento que o neurônio de saída não será ativado é quando as duas entradas forem iguais a 1.

Logo os pesos desempenham um papel importantíssimo no funcionamento da rede. Dependendo dos pesos sinápticos a rede terá um determinado comportamento. Dessa forma os pesos representam o conhecimento armazenado e obtido pela rede através do treinamento(HAYKIN, 2001).

Afim de se obter os valores adequados para os pesos sinápticos, a rede passa por um processo chamado de: processo de aprendizagem. Um tipo de processo de aprendizagem, é a aprendizagem por correção de erro. Que consiste em comparar a saída real com a saída esperada(HAYKIN, 2001). Dessa forma os pesos vão se ajustando conforme for necessário, e uma das formas de se realizar o ajuste é através do algoritmo *backpropagation.* O algoritmo consiste em realizar o ajuste dos pesos partindo da saída até as entradas da rede. Esse ajuste ocorre através da derivada da função de erro(Fleck *et al.*, 2016).

Haykin (2001) chama esse processo de aprendizagem com um professor e descreve como esse processo funciona em RNA. Segundo Haykin (2001), o professor, responsável por ter o conhecimento que é representado pelo conjunto de dados de treinamento, é capaz de oferecer a resposta correta visto que já tem um conhecimento prévio. Essa resposta, seria a saída desejada a ser dada pela rede. Dessa forma, toda

vez que a rede é exposta aos dados de treinamento é feito uma comparação com o saída desejada oferecido pelo professor. Sendo assim, a diferença entre a saída desejada e a respostada dada pela rede corresponde ao sinal de erro. A rede por sua vez, ajusta os parâmetros afim de que o conhecimento fornecido pelo professor, seja passado para a rede neural.

O processo de treinamento ocorre através de iterações para cada saída da rede até que o algoritmo atinja a convergência. A convergência do algoritmo significa o mínimo para aquela função de erro, ou seja, o resultado da taxa de erro já começa a sofrer pouca alteração ao passar das iterações.

As RNA também podem ser agrupadas de acordo com sua arquitetura. A arquitetura de uma RNA representa a forma como os neurônios estão agrupados e como eles se conectam. Dessa forma, a arquitetura da rede está intimamente ligada ao algoritmo utilizado para treinar a rede.

Haykin (2001), menciona três classificações de rede: Redes Alimentadas Adiante com Aamada Única, Redes Alimentadas Diretamente com Múltiplas Camadas e Redes Recorrentes.

2.1.3.1 Redes com múltiplas camadas

Um único neurônio não é capaz de trazer respostas significativas e complexas, porém a conexão de várias unidades neuronais aumenta consideravelmente as funcionalidades de uma RNA. Há várias topologias/arquiteturas de redes, entre as topologias mais conhecidas, está a topologia de múltiplas camadas e totalmente conectada. Esta subseção foca na descrição da topologia de múltiplas camadas (TMC), já que o sistema desenvolvido utiliza esta arquitetura.

Em redes com TCM cada unidade neuronal das camadas se conecta com todas as unidades da próxima camada, ou seja, a saída de uma camada é a entrada da próxima camada. Dessa forma o sinal de entrada passa de uma camada para outra até a camada de saída onde passará pela função de erro, nunca um sinal retorna para a camada anterior (Haykin, 2001).

2.1.3.2 Deep Learning

Desde o início dos anos 2000, pesquisas e estudos em redes profundas tem revolucionado várias áreas do aprendizado de máquina, uma dessas áreas é a visão computacional. Isso tem acontecido pelo aumento da quantidade de dados abertos, que possuem milhões de imagens e também o avanço da própria tecnologia que proporciona *hardwares* mais robustos, capazes de processar diversos dados por segundo descreve Ponti e Costa, (2017)

Com o grande número de pesquisas e o grande potencial das redes de aprendizado profundo *(DL)*, muitas áreas tem se beneficiado de suas aplicações. O sucesso das DL's não se restringe a apenas aplicações de visão computacional, mas se expande para reconhecimento de fala, texto e até mesmo computação gráfica. Junto com suas aplicações surgiram arquiteturas como por exemplo a redes neurais de convolução (CNN), explicadas na seção 2.1.3.1. Ponti e Costa, (2017) lembra que apesar de tamanho sucesso, pesquisadores estão tendo dificuldades de explicar questões como por exemplo as limitações que essas redes possuem.

Ainda segundo Ponti e Costa (2017), para um sistema ser considerado como utilizando DL, ele deve buscar um modelo gerado através dos exemplos previamente analisados e um algoritmo para guiar o aprendizado da rede a partir dos próprios dados de treinamento da rede. O resultado do processo de treinamento, gera uma função capaz de a partir dos dados brutos de entrada, oferecer a resposta que represente de forma adequada a solução do problema, a partir do aprendizado da função.

Moacir A. Ponti e Costa(2017) chama a atenção ao fato de que em métodos DL, a forma como a rede aprende a função é o diferencial. Isso se dá pelo fato de que redes "superficiais" ou "rasas", como são chamados os métodos que não são DL, possuem uma única função que, a partir de um conjunto de dados, conseguem gerar o resultado esperado. Diferentemente, as funções dos métodos DL são formadas a partir de composições de funções, como vemos na função:

$$F_L(\cdots f2(f1(x1, W1); W2)\cdots) W_L) \qquad (3)$$

O índice " $_L$ " corresponde a camada da função específica, já o índice " X " corresponde ao vetor de entrada da camada " $_L$ ". o resultado da função irá gerar um

outro vetor “X_L + 1” que será a entrada da próxima camada. Cada W_L corresponde a um vetor de pesos que irão modificar o conjunto de dados de entrada como descreve Ponti e Costa (2017).

Dessa forma, Ponti e Costa (2017) define uma das ideias principais da DL como sendo: “aprender sucessivas representações dos dados”. Com issso a profundidade se refere as representações sussevias e permite aprender funções que transformam o vetor de entrada.

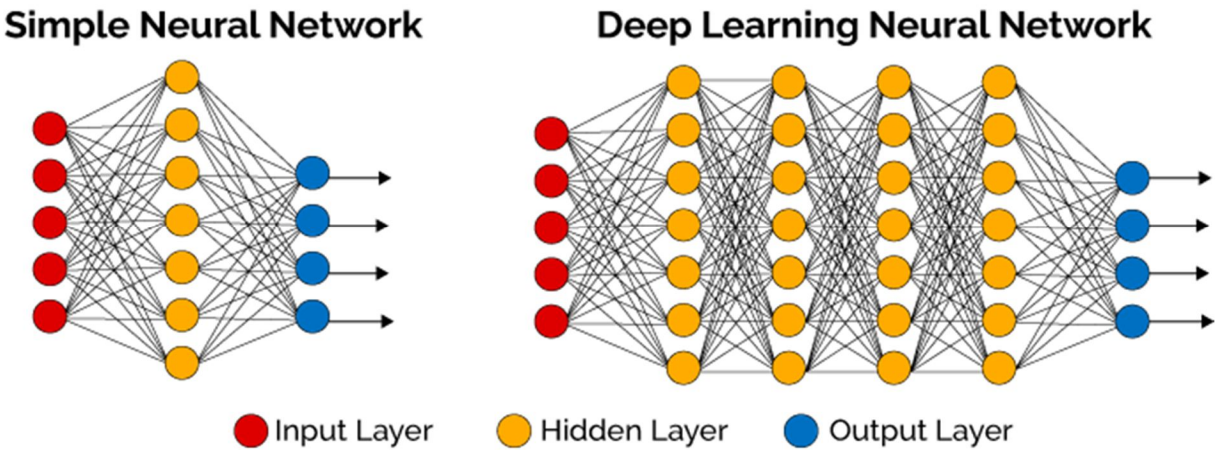
Para que o sistema seja um classificador eficiente, Ponti e Costa (2017) define as caracteristicas de comtribuem para o melhor funcionamento em DL da seguinte forma:

> Se tivermos um número suficiente de camadas L, espaços com dimensionalidade alta o suficiente, i.e., o número de parâmetros W em cada função, e dados suficientes para aprender os parâmetros WL para todo L, então conseguiremos capturar o escopo das relações nos dados originais, encontrando assim a representação mais adequada para a tarefa desejada.

Ponti e Costa (2017) que os métodos DL podem ser definidos como sendo métodos que utilizam camadas de neurônios para analisar os dados do vetor de entrada e dar uma resposta aproximada a esperada na camada de saída. As informações saem da camada anterior para a camada seguinte e seguem esse processo até a camada de saída. Sendo assim cada camada da rede DL fará transformações nos dados de entrada através da função estabelecida para sua camada, a transformação gerará uma respresentação que será passada para a camada seguinte. Essa arquitetura é mostrada na Figura 9.

Com o sucesso das redes de aprendizado profundo inúmeras arquiteturas surgiram e um dos tipos de arquitetura é a *convolutional neural network(CNN)* (Deng e Yu, 2013). As CNN’s são redes de múltiplas camadas, onde cada camada é responsável por realizar convoluções no seu vetor de entrada e dar a saída para a próxima camada.

Figura 9 Diferença entre uma rede neural simples e *Deep Learning*

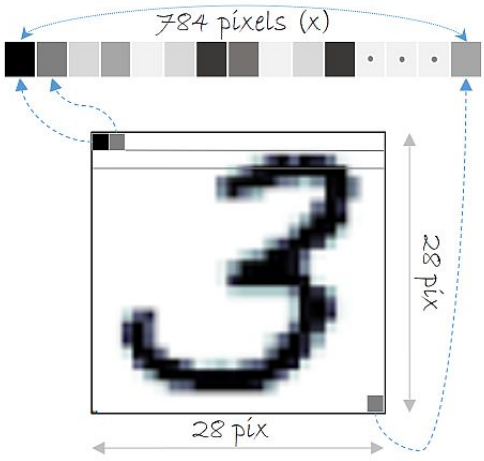


Fonte: (Gomez, 2018)

Grande parte das aplicações envolvendo CNN, tem como objetivo reconhecimento de objeto em imagens segundo Ferreira, (2017). Os dados de treinamento dessas redes são grupos de imagens rotuladas que sofrerão convoluções sucessivas no decorrer da rede, no processo de treinamento.

Para a imagem ir para as camadas internas da rede ela precisará ser transformada em um vetor como mostrado na Figura 10. Sendo assim se uma imagem de entrada possui 784 pixels, a camada inicial possuirá 784 entradas que passarão por convolução nas camadas ocultas, até a camada de saída. O número de neurônios na saída é definido em função da quantidade de saídas que deseja-se representar, ou seja, se deseja-se representar 3 rótulos distintos, serão 3 neurônios para saída da rede.

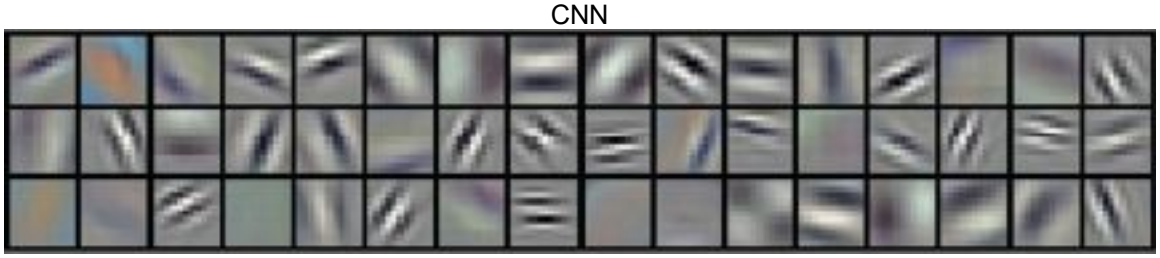
Figura 10 Processo de transformar em vetor



Fonte: (Microsoft, 2017)

As CNN's também possuem um Kernel que passará por convolução com a imagem. O Kernel representa a característica que deseja se encontrar, e na rede ele é representado pelos pesos sinápticos entre os neurônios. Visto que o objetivo do treinamento é ajustar os pesos sinápticos para ter menor erro possível, o treinamento de CNN significa ajustar o kernel afim de se encontrar o melhor agrupamento possível. O ajuste dos pesos resulta em imagens(Kernels) como as da Figura 11.

Figura 11 Exemplo de 48 Kernels aprendidos no treinamento do projeto *ImageNet classification With* CNN



Fonte: (Krizhevsky, Sutskever e Hinton, 2012)

Em alguns casos, principalmente nas imagens coloridas é possível se encontrar três canais/matrizes que representam três cores: vermelho, verde e azul (RGB), dessa forma a convolução deve ser aplicada em todas as camadas.

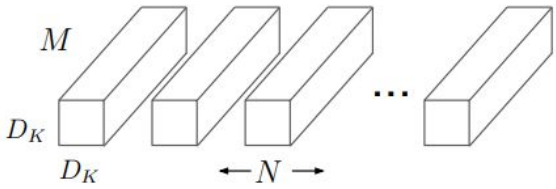
As CNN's têm se popularizado na visão computacional, e o desempenho obtido tem sido satisfatório. Howard *et al* (2017) traz atenção ao fato de que a plataforma *Mobile* tem crescido, o que tem aumentado a necessidade de que essas tecnologias atuem até mesmo em *hardwares* mais limitados. Outra necessidade também são dispositivos embarcados, como por exemplo placas Arduino, que também possuem hardwares limitados e conseguem processar aplicações que são mais limitadas.

Diante disso Howard *et al* (2017) propôs um modelo de rede convolucional que fosse eficiente e consiga atingir os requisitos de sistemas com hardwares limitados, como é o caso de dispositivos *mobile* e sistemas embarcados. Esse modelo é a *Mobilenet*. Explica também, que o modelo *mobilenet* foi construído com base em um modelo de convolução chamado: *Depthwise separeble convolution.* Esse modelo de convolução tem como objetivo reduzir os custos computacionais em algumas camadas iniciais da rede.

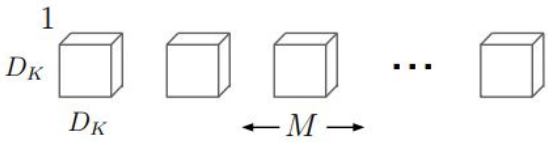
A *Depthwise* consiste em aplicar convoluções na imagem de forma fatorada, o que significa que uma convolução normal é realizada em apenas uma camada da imagem de entrada. Essa convolução é seguida de uma convolução pontual chamada

pointwise, que consiste em aplicar uma convolução com kernel 1x1 em todas as camadas e combinar o resultado com o *depthwise* Figura 12.

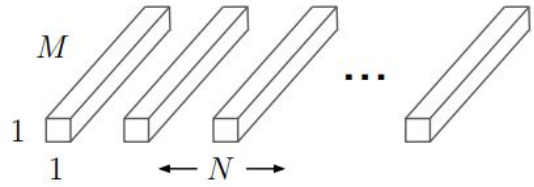
Figura 12 O modelo convolucional normal (a), é substituído por duas camadas: depthwise em (b) e pointwise (c)



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



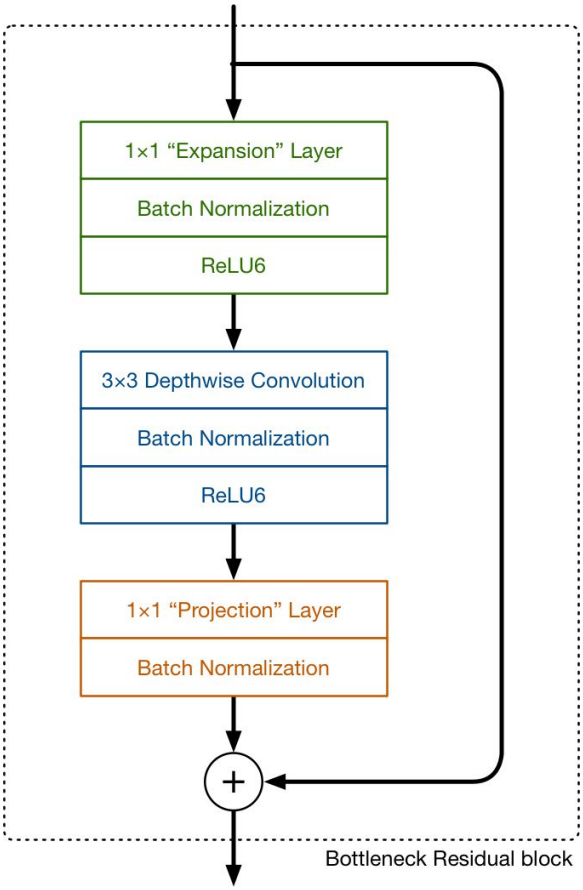
(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Fonte: (Howard *et al.*, 2017, p 3)

Na Figura 12(a), vemos a estrutura de um kernel de tamanho Dk e profundidade M, onde pode ser escolhido quantos filtros N for necessário. Essa estrutura é referente a convoluções comuns. Já na Figura 12(b) e (c), as estruturas se referem aos kernel das duas camadas citadas: *depthwise* e *pointwise* A Figura 12(b) retrada o kernel com dimensão Dk e profundidade 1 já que a convolução só é feita em um canal da imagem e pode ser escolhido M filtros, o resultado é combinado com um filtro 1x1 de profundidade com a profundidade limitada da saída da camada Figura 12(c), o objetivo é reduzir a quantidade de dados que passa para próxima camada.

Como já foi destacado, *depthwise* e *pointwise* tratam-se de camadas diferentes. Entre elas ocorre uma normalização dos valores de entrada e os valores passam por uma função de ativação chamada ativação linear retificada (ReLU). Antes de cada camada *depthwise* existe uma camada de convolução 1x1 e função de ativação ReLU. Essa estrutura pode ser observada na Figura 13

Figura 13 Modelo de camada *Bottleneck* utilizado na rede *MobileNet*



Fonte: (Matthijs Hollemans, 2018)

A Figura 13 retrata o modelo de camada chamada de *bottleneck,* esse modelo possui uma camada inicial de convolução 1x1. Ao todo a rede possui inicialmente uma camada densamente conectada seguida de dezenove camadas de *bottleneck.* A camada também possui uma conexão entre as entradas e as saídas, utilizada para passar o gradiente para ajuste de pesos. Essa conexão só é utilizada quando o número de canais na imagem de entrada é idêntico ao número de canais da imagem de saída.

2.1.3.3 Treinamento

O treinamento é o momento em que a RNA aprende dos dados de entrada, esse aprendizado ocorre através do ajuste dos pesos sinápticos, e para os pesos serem ajustados é necessário que a RNA calcule o erro de cada treino.

A fim de se calcular o valor do erro, o algoritmo de treinamento utiliza um método não publicado chamado RMSprop, proposto por Geoff Hinton na aula 6e do curso Cursera (Ruder, 2016).

O método consiste em dividir a taxa de aprendizado, pela raiz quadrada da função de custo para a atualização dos pesos da rede. A função de custo pode ser analisada na equação a seguir:

$$\mathrm{E}[g^2]_t = 0.9\mathrm{E}[g^2]_{t-1} + 0.1g^2_t \qquad (4)$$

Na Equação 4 nota-se a função de custo E calculada a partir do resultando anterior "$E[g^2]_{t-1}$" mais o gradiente do erro atual. O resultado será utilizado para modificar os parâmetros da rede como mostra a equação a seguir:

$$\theta_{T+1} = \theta_T - \frac{n}{\sqrt{E[g^2]_T + \epsilon}} \qquad (5)$$

Na Equação 5 cada peso da rede "θ(T+1)" recebe o "θ_T" atual, subtraindo da divisão entre a taxa de aprendizado "n" pela raiz do erro calculado na Equação 4.

Dessa forma, a cada etapa do treinamento é calculado o erro em relação a saída esperada. No projeto desenvolvido o erro foi adotado como parâmetro para o término do treinamento da rede.

2.2 TRABALHOS RELACIONADOS

Afim de se compreender os modelos de arquiteturas das RNA's e eleger uma que fosse eficiente para detecção de degraus, foram analisados alguns projetos na área de detecção de objetos e RNA's, afim de se ter compreensão da área estudada e quais ferramentas estão sendo mais utilizadas.

Na pesquisa feita por Ferreira (2017), foi utilizado redes neurais convolucionais, a fim de se reconhecer ervas daninhas em imagens de lavouras de soja. Para o treinamento da RNA desenvolvida foi utilizado o framework *CaffeNet* e um modelo de rede baseado na estrutura da rede *AlexNet*. Em seu trabalho foi desenvolvido uma aplicação para construção do banco de imagens. Os resultados foram excelentes sendo possível alcançar 99% de precisão nas classificações.

Para se reconhecer cerca de quinze milhões de imagens, divididas em vinte e duas mil classes rotuladas, localizadas em um banco de imagens chamando *ImageNet,* Krizhevsky, Sutskever e Hinton (2012) utilizou uma CNN com 8 camadas, sendo as 5 primeiras camadas convoluções e as 3 ultimas camadas densamente conectadas. O treinamento ocorreu através do método stochastic gradient descent durando cerca de 6 dias para o treinamento através de duas GPUs. O resultado é que a rede obteve 37,5% de taxa de erro durante os teste.

OLIVEIRA, (2017) propôs o treinamento de CNN para reconhecimento de 7 expressões faciais utilizando a biblioteca *lasagne*, sendo elas: Irritado, Desgosto, Medo, feliz, Triste, Surpreso e Neutro. Para os teste foram utilizadas 3 redes convolucionais diferentes, tendo desempenho com precisão de 65,48% no reconhecimento das expressões faciais.

Os trabalhos relacionados tiveram grande importancia para definição da estrutura do projeto. A partir da analise dos trabalhos foi possível decidir qual arquitetura de rede utilizar, e qual *framework* utilizar para o treinamento da rede. Na análise foi identificado que as CNNs são ideiais para classificação de imagens e que a utilização do framework *Tensorflow* seria ideal para o projeto desenvolvido.

3 O SISTEMA

Neste capitulo é descrito o desenvolvimento do sistema que faz o reconhecimento de degraus. O treinamento da RNA foi realizado através do *framework Tensorflow* em sua versão para a linguagem de programação Python e placa gráfica. Com o objetivo de se ter um melhor desempenho, foi utilizado uma placa gráfica Nvidia 920M.

A etapa de treinamento do sistema pode ser subdividida em 3 passos. Essas etapas são: Aquisição de imagens, rotulação das imagens, e treinamento da rede com base no banco de imagens rotuladas montado.

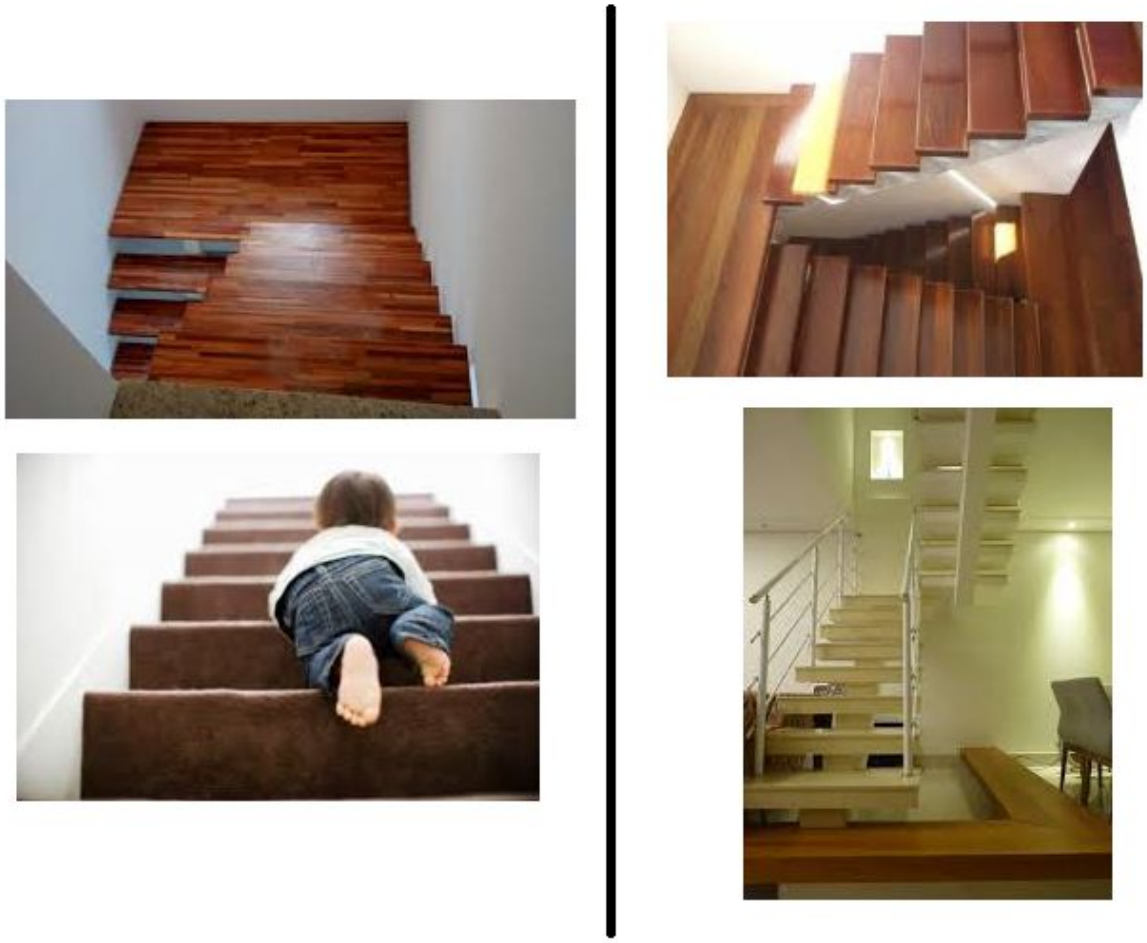
3.1 AQUISIÇÃO DAS IMAGENS

Como foi analisado na seção 2.1.2 é necessário um grupo de dados de treinamento, que será utilizado para passar o conhecimento para o sistema de AM. No caso desta pesquisa, o conjunto de dados de treinamento é composto por cerca de 115 imagens de degraus em perspectivas distintas. Essas imagens estão divididas em dois parâmetros, um grupo de imagens é utilizado para o treino e outro grupo de imagens, é utilizado para testes.

Para se caracterizar o objeto a ser detectado, as imagens de treinamento são compostas por degraus ao lado de outros objetos distintos, afim de que a rede possa diferenciar as características do objeto alvo. A Figura 14 mostra exemplos do banco de imagens de treinamento e teste formado a partir das imagens selecionadas,a esquerda imagens de banco de testes e a direita imagens de treino.

Um dos objetivos para o treinamento da rede, é a utilização de cerca de 300 imagens para o treinamento, porém por limitações de hardware, o sistema não conseguiu treinar um número maior de imagens. Por esse motivo foram utilizadas cerca de 115 imagens divididas entre 23 imagens para teste e 92 para treino. Para um bom funcionamento da rede, as imagens de teste devem ser em situações distintas das imagens de treino a fim da rede conseguir ter resultados precisos.

Figura 14 Exemplo de imagem do banco de treinamento.



Fonte: Autor

Para que cada degrau possa ser detectado e localizado na imagem é extremamente necessário rotular essas imagens, por isso na próxima seção será descrito o processo de rotulação das imagens do banco de treino e de teste.

3.2 ROTULAÇÃO DAS IMAGENS

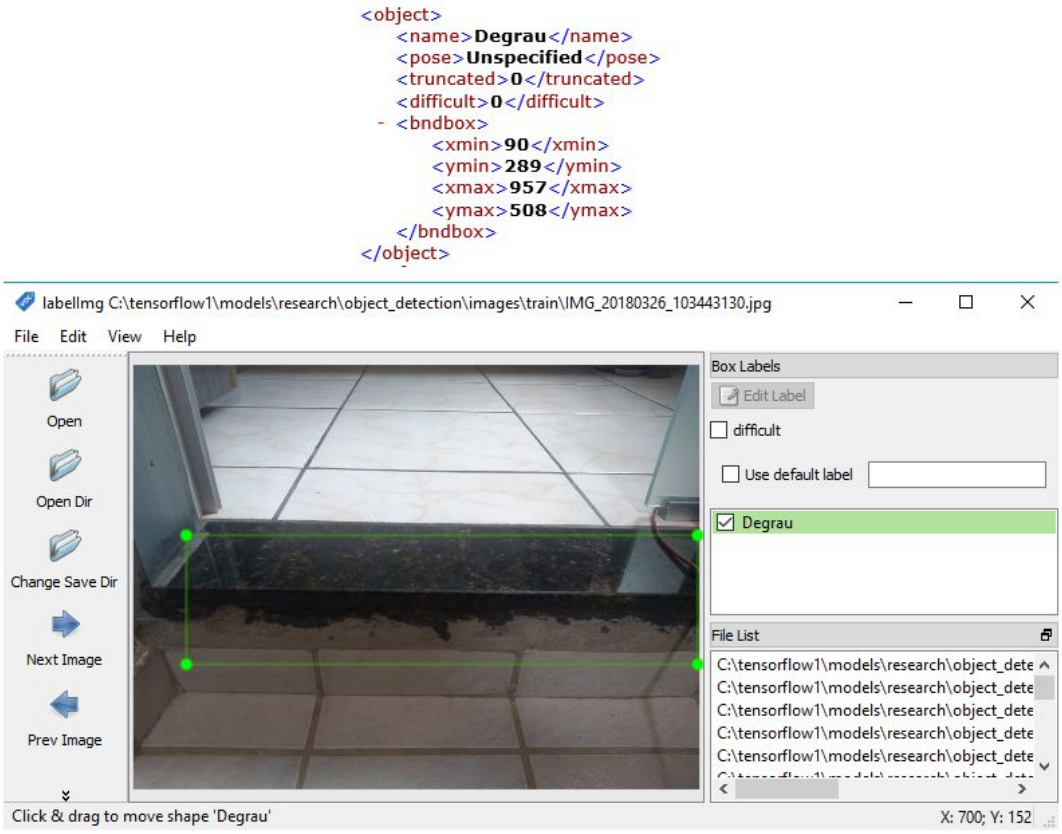
Como foi analisado na seção 2.1.2, uma das características encontrados em sistemas AM é a rotulação, dessa forma é necessário que os dados de treinamento sejam rotulados, o que significa que cada imagem do banco de imagens deve ser rotulada indicando onde se encontra o degrau, o objetivo desse procedimento é que

a rede possa receber como entrada o mapa de rótulos, dessa forma a rede pode comparar a saída real com a saída esperada.

Para a rotulação das imagens foi utilizado o *software* LabelImg (Tzutalin, 2015). Esta aplicação desenvolvida em Python, tem como finalidade rotular as imagens salvando os rótulos dessas imagens em um arquivo em formato XML como o da Figura 15.

O processo de rotulação é extremamente necessário para o treinamento da rede, já que, como foi visto na seção 2.1.3, é neste momento que os pesos sinápticos são alterados. A Figura 15 descreve o processo de rotulação da imagem usando o *software* LabelImage, onde é manualmente desenhado, através de uma caixa cada objeto alvo presente na imagem e selecionado a classe que ele pertence. A cima o arquivo XML de uma imagem e a baixo o processo de rotulação. Esse processo é repetido para todas as imagens, consequentemente cada imagem terá seu próprio mapa de características, o que torna necessário unir esses dados em um arquivo só, que servirá como entrada da rede.

Figura 15 Processo de rotulação com *LabelImg.*



Fonte: Autor

Para se unir todos os arquivos XML das imagens, foi utilizado um *script* em Python para criar uma tabela no formato CSV, com base em todos os arquivos XML das imagens de treino e outra tabela com base nos arquivos XML das imagens de teste. Após executar o script, todas as características rotuladas das imagens de treino e de teste ficam agrupadas em 2 arquivos, um de treino e outro de teste. Para que o *framework Tensorflow* possa entender esses arquivos é necessário gerar outros arquivos com formato *record.*

Para a execução dessa tarefa foi utilizado um outro *script* para gerar um arquivo *record* para o CSV de testes, e outro para o CSV de treino. Os arquivos gerados pelo *script* são responsáveis por passar para a rede o identificador de cada rótulo alvo, que no caso desta pesquisa será apenas os degraus.

Visto que há a possibilidade de existirem vários rótulos, é necessário criar um arquivo, que foi chamado de *labelmap,* que indique todos os rótulos e os ids que correspondem a cada rótulo. A Figura 16 mostra o arquivo *labelmap* com o rótulo para os degraus. Esse arquivo também será utilizado pela aplicação OpenCV afim de que a aplicação possa utilizar o nome do rótulo bem como seu id.

Figura 16 Exemplo do LabelMap

```
item {
  id: 1
  name: 'Degrau'
}
```

Fonte: Autor

Após a configuração dos arquivos que serão utilizados pela rede no treinamento, eles podem ser passados para o arquivo disponibilizado pela *Tensorflow* com as configurações da rede que será responsável por determinar toda a estutura da rede, bem como as informações para o treinamento da rede, dessa forma é extremamente necessário alterar o arquivo de configuração, para adicionar a quantidade de rótulos que se deseja determinar, bem como adicionar os caminhos para os dados de treinamento da rede.

Sendo feitas todas essas configurações o treinamento da rede pode ser realizado utilizando a *tensorflow*.

3.3 TREINAMENTO DA REDE

Uma das tarefas mais importantes para o bom desempenho de uma rede neural é o treinamento. Essa etapa é responsável por ajustar os pesos da rede e dessa forma caracterizar o objeto que será detectado, em função do coeficiente de perda como analisado na seção 2.1.3.

A *tensorflow* dá suporte para essa tarefa por disponibilizar um *script* que é responsável por realizar esse treinamento e gerar dados importantes como desempenho da rede ao longo do treinamento, demonstrando os números do treinamento da rede. Dessa forma nesta seção será analizado como o treinamento da rede foi realizado, e como a rede se comportou nesse treinamento tendo como referência o coeficiente de perda ao longo dos passos de treinamento da rede.

Para realizar o treinamento da rede, a *Tensorflow* disponibiliza o arquivo *train.py* que é responsável por gerenciar o aprendizado do modelo neural utilizando o arquivo de configuração da rede que contém todos os dados para o treinamento gerados bem como o rótulo para cada imagem.

O *script* de treino, ao ser inicializado começa o treinamento realizando comparações entre a saída, obitida a partir da imagem de entrada da rede, e o resultado esperado, obtido a partir dos rótulos de cada imagem. Dessa forma determina a taxa de erro daquela comparação, cada comparação será chamada de passos como vemos na Figura 17.

Figura 17 Processo de treinamento da rede.

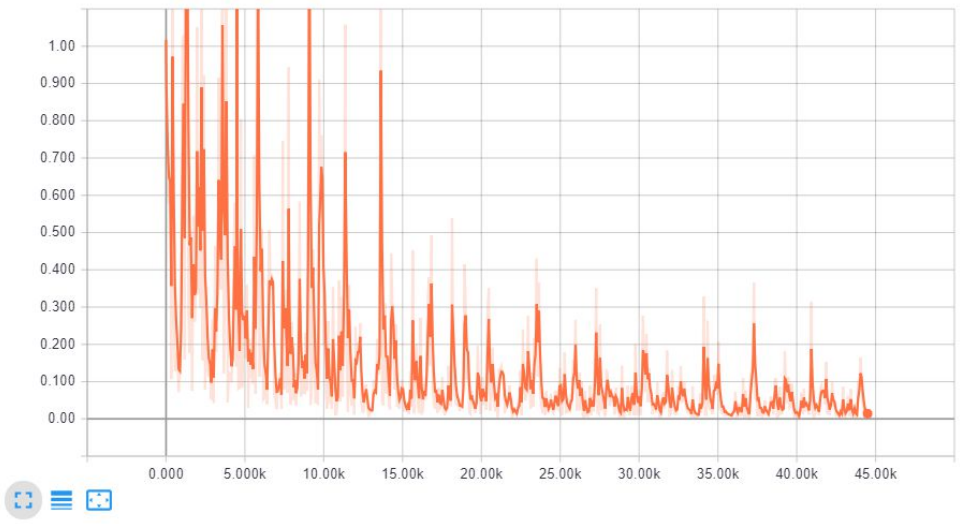
```
INFO:tensorflow:global step 8072: loss = 0.0617 (1.552 sec/step)
INFO:tensorflow:global step 8073: loss = 0.0562 (1.525 sec/step)
INFO:tensorflow:global step 8074: loss = 0.0370 (1.545 sec/step)
INFO:tensorflow:global step 8075: loss = 0.0580 (1.552 sec/step)
INFO:tensorflow:global step 8076: loss = 0.0550 (1.535 sec/step)
INFO:tensorflow:global step 8077: loss = 0.0574 (1.559 sec/step)
INFO:tensorflow:global step 8078: loss = 0.0598 (1.571 sec/step)
INFO:tensorflow:global step 8079: loss = 0.0448 (1.543 sec/step)
INFO:tensorflow:global step 8080: loss = 0.0485 (1.573 sec/step)
INFO:tensorflow:global step 8081: loss = 0.0524 (1.547 sec/step)
INFO:tensorflow:global step 8082: loss = 0.0360 (1.545 sec/step)
INFO:tensorflow:global step 8083: loss = 0.0514 (1.550 sec/step)
```

Fonte: Autor

O processo de aprendizagem é registrado em um arquivo de log, que armazena todos os dados referente ao apredizado da rede. Este arquiovo pode ser lido pela *tensorboard*, sistema da *tensorflow* que é responsável por mostrar o desempenho da rede durante o treinamento. Dessa forma o *script train.py* recebe como entrada o enderenço onde irá armazenar o arquivo de log e o endereço onde se encontram o arquivo de configuração da rede.

O treinamento da rede durou um periodo de 12 horas, que foi interrompido por a rede ter convergido, porém o objetivo esperado para a taxa de erro no treinamento da rede era de fosse inferior a 0,005, porém o algoritimo convergiu antes, tendo seu mínimo de taxa de erro igual a 0,009, por a taxa de erro não diminuir siginificativamento, o algoritmo foi interrompido gerando o gráfico da Figura 20, que representa o erro de cada passo do treinamento.

Figura 18 Taxa de erro(y) por cada passo(x) de treinamento realizado.



Fonte: Autor

Como vemos na Figura 18, é possível se notar uma curva acentuada de aprendizado, no começo do treinamento a taxa de erro que tinha uma variação alta, possuía valores bem superiores do que ao final do treinamento. No decorrer da aprendizagem da rede, é possível se notar que tanto a taxa de erro diminui como a variação do erro, sendo no final do treinamento uma variação quase que constante e o erro bem inferior do que no início do treinamento.

Após o treinamento da rede ser concluído, pode-se gerar um arquivo que será utilizado pela aplicação opencv como classificador. Esse arquivo vai servir como uma rede pré-treinada e terá como objetivo apenas a classificação. O arquivo gerado terá formato binário e é utilizado apenas para a comunicação entre a OpenCV e a *tensorflow*.

Para gerar este arquivo, foi utilizado um *script* em *Python,* que receberá como entrada: o tipo de dado de entrada que a rede trabalha, que no caso da pesquisa são imagens e é chamado de *image_tensor*, o arquivo de configuração da rede, o último arquivo de log realizado pelo treinamento e por fim o diretório onde será armazenado o arquivo gerado pelo *script*.

O arquivo gerado pelo *script,* tem como objetivo armazenar a estrutura da rede bem como alguns pesos obtidos durante o treinamento, esse arquivo que foi utilizado pela OpenCV sendo lido e gerando a classificação e criando caixas de classificação que serão mostradas na tela.

Após se obter todos os arquivos utilizados no treinamento da rede e gerar o arquivo que fará a classificação junto com a aplicação OpenCV, o sistema está pronto para passar por testes e verificar o seu comportamento diante dos degraus encontrados. Esses testes são analisados no próximo capítulo.

4 RESULTADOS

Após o treinamento da rede, o sistema está pronto para ser validado, passando por testes que verificam a eficiência do sistema neural em detectar os degraus em situações distintas.

O objetivo do sistema, é detectar os degraus que estão à frente da câmera e informar na tela onde se encontram. Os degraus serão marcando com um retângulo verde na região da imagem onde foram detectados. O sistema deve detectar um ou mais degraus em cada imagem.

Os testes descritos nesse capitulo vão estar divididos em três sessões. Na primeira sessão será analisado a eficiência da rede em detectar os degraus expostos a uma boa iluminação, porém com variações nas cores dos degraus. Em seguida a segunda sessão trará os resultados da rede em detectar os degraus expostos a uma baixa iluminação, os degraus também sofrerão variação nas cores. Por fim o terceiro teste avaliará o desempenho da rede ao detectar degraus em ângulos diferentes.

Os testes realizados visam determinar se a rede neural treinada consegue desempenhar seu papel em situações comuns no dia a dia, dessa forma, será possível avaliar o desempenho da rede e determinar sua eficiência em realizar a tarefa.

4.1 PRIMEIRO EXPERIMENTO

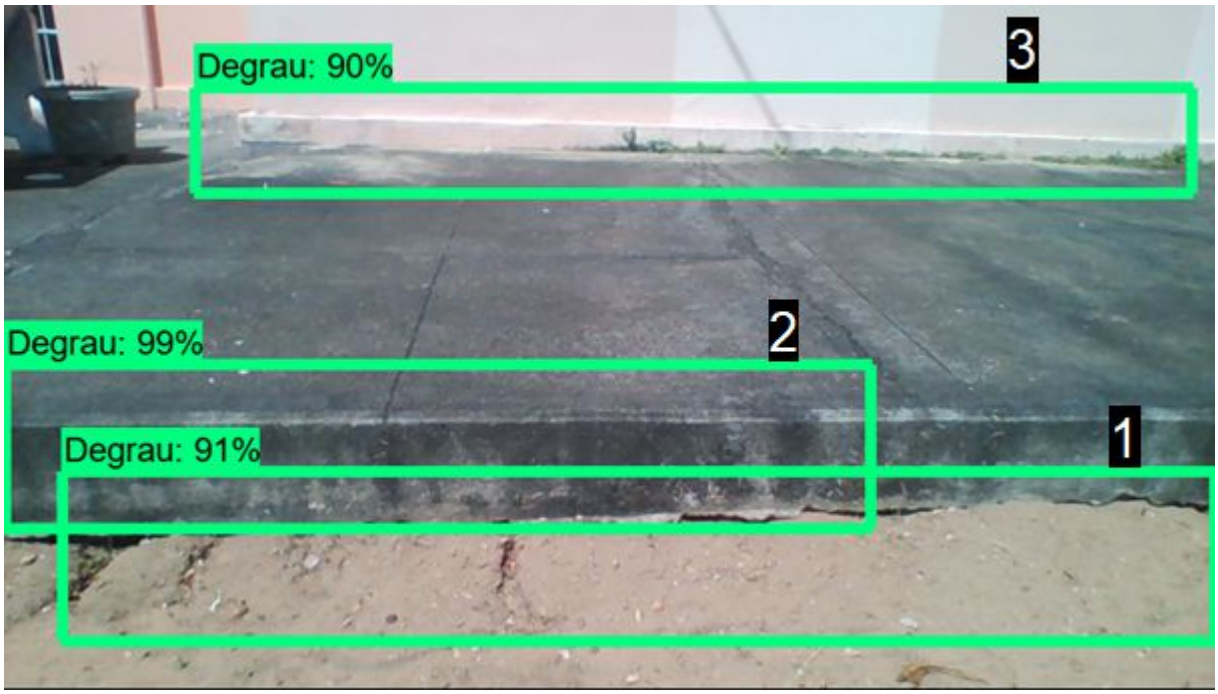
O primeiro experimento consiste em direcionar o foco da câmera para degraus expostos a uma boa iluminação. Dessa forma esses experimentos foram realizados durante a parte da manhã do dia, o que tornava os objetos alvos bem iluminados e destacados na imagem.

Para um bom resultado nesse experimento o sistema deve detectar os degraus em duas situações distintas: primeiramente a câmera foi apontada para degraus que variam em sua tonalidade de cor, dessa forma a rede treinada deve detectar maioria dos degraus contidos na imagem. Na segunda situação os degraus não possuem variação significativa em sua tonalidade de cor, dessa forma ao se apontar a câmera em direção dos degraus o sistema também deve detectar o máximo que conseguir de degraus que estiverem na imagem.

Na Figura 19 observa-se um exemplo referente ao primeiro experimento e as previsões realizadas pela rede referente a essa situação. Nessa imagem pode-se

observar degraus que contém algumas variações em sua tonalidade, na imagem é possível notar que a parte superior do degrau, difere em cor de sua base.

Figura 19 Câmera voltada para parte frontal de um degrau.



Fonte: Autor

É possível se observar na Figura 19, a visão da parte frontal dos degraus. Na imagem pode-se notar a existência de um primeiro degrau na parte inferior da figura. A base do degrau é composta por uma cor diferente da do corpo do degrau. Já na parte superior da imagem é possível se notar a existência de um outro pequeno degrau próximo a uma parede, novamente nesse pequeno degrau nota-se que a base do degrau difere do seu corpo.

Ainda referente a Figura 19, foi observado que a rede fez 3 previsões. Na primeira previsão nota-se que a rede calculou uma probabilidade de 91% da existência de um degrau naquela região da imagem. Por mais que a rede não tenha errado, visto que existem características de um degrau na imagem, o objetivo é que a rede detecte apenas o degrau, como o que acontece nas detecções 2 e 3.

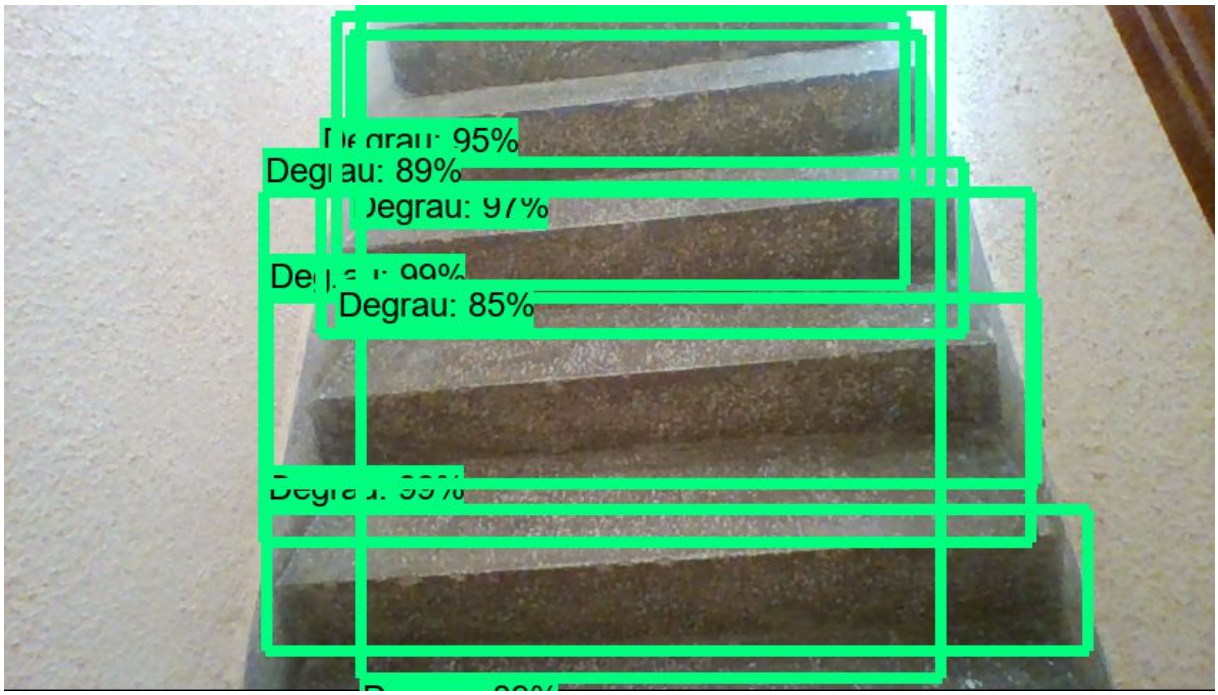
Na detecção 2 da Figura 19, o sistema fez uma excelente previsão, detectando com probabilidade de 99% a existência de um degrau na região da figura. Dessa forma

a rede fez a detecção da forma correta, marcando na tela o posicionamento do degrau na imagem.

Ainda na Figura 19 observamos a detecção 3. De forma correta a rede marcou na imagem o local onde se encontra o degrau. Por mais que represente um pequeno degrau, aquela região possui características de um degrau e demonstra que a rede teve um bom desempenho nessa tarefa já que a rede fez uma detecção com a probabilidade de 90% da existência de um degrau na região da imagem.

Com o objetivo de validar a eficácia do sistema proposto, nesta seção também será analisada degraus com pouca variação na tonalidade de cor. Na Figura 20 é possível observar uma escadaria onde os degraus possuem a mesma tonalidade de cor, sofrendo influencia apenas da iluminação ambiente.

Figura 20 Câmera posicionada na parte frontal e direcionada a uma escadaria



Fonte: Autor

Analisando a Figura 20 é possivel se observar que existem 4 degraus na imagem. Cada degrau recebeu sua marcação, porém o sistema detectou tres outros objetos que não se caracterizam como sendo um degrau. Por mais que o sistema tenha tido essa variação, o resultado não foi insatisfatório visto que todos os degraus da imagem foram detectados e os objetos detectados que não são degrau possuem porcetagem a baixo de 90%.

Após a conclusão dos testes representados pelas figuras 19 e 20, foi possível avaliar o desempenho da rede como satisfatório, visto que a rede realizou detecções precisas mesmo quando exposta a períodos de baixa de luminosidade.

4.2 SEGUNDO EXPERIMENTO

No segundo experimento o sistema foi utilizado situações com degraus sob pouca iluminação, para isso os testes foram realizados no período do anoitecer. Por sua vez, o teste passou pelas mesmas situações do primeiro experimento, em uma ocasião os degraus possuem variação em sua tonalidade, já em outra situação o degrau possui a tonalidade de cor aproximadamente constante.

Analisando a Figura 23, podemos observar uma situação semelhante à da Figura 21, analisamos dois degraus com variações de cor. Um na parte inferior da imagem e outro na parte superior da imagem.

Figura 21 Imagem da parte frontal do degrau, com baixa iluminação.

Fonte: Autor

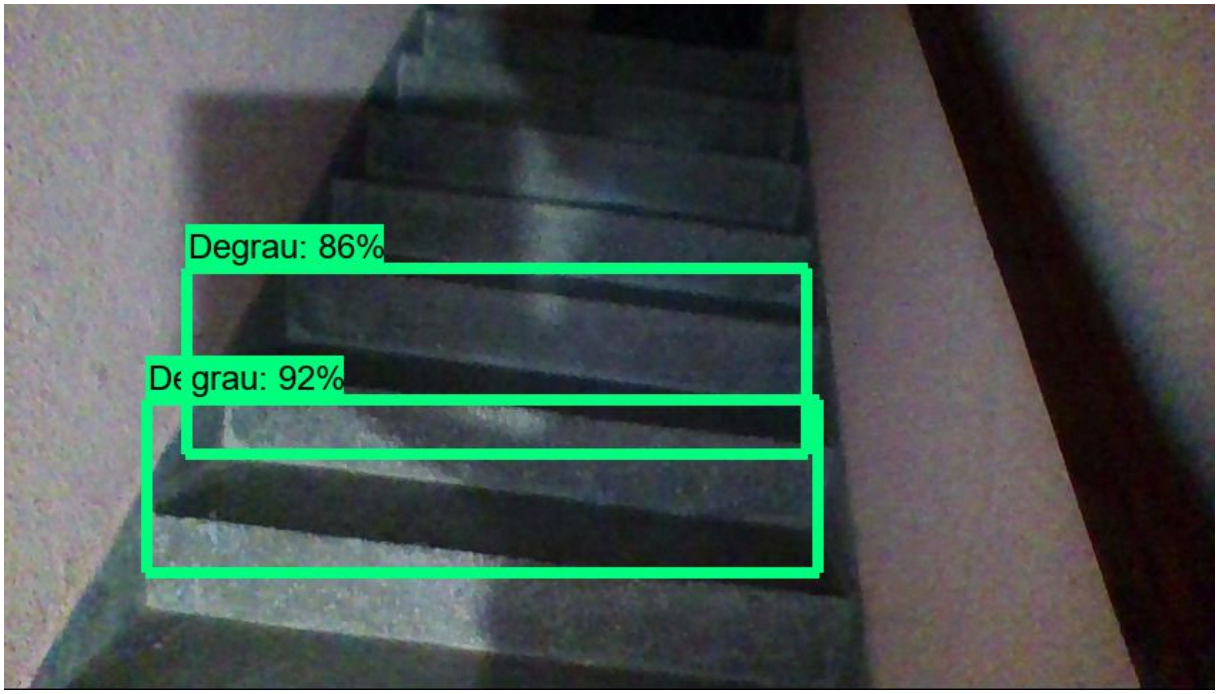
É possível se observar na Figura 21 que o sistema realizou duas previsões na imagem. A primeira previsão o sistema detectou corretamente o degrau no centro da

imagem. Nessa previsão o sistema deu corretamente uma probabilidade de 97% de chance de naquela região da imagem conter um degrau.

Na segunda previsão o sistema não foi muito preciso em determinar onde está localizado o degrau. Por mais que tenha acertado não se pode afirmar que o sistema encontrou o degrau na imagem

Na Figura 22 pode-se nota uma escadaria semelhante à da Figura 20, onde a rede conseguiu detectar a presença de dois degraus com precisão de 86% e 92% de probabilidade de ser ou não um degrau. Nesse teste em particular foi percebida uma dificuldade em detectar os degraus localizados na parte com menos iluminação do corredor.

Figura 22 Parte frontal de uma escadaria no período da noite



Fonte: Autor

Nos testes envolvendo pouca variação na tonalidade de cor. O sistema apresentou muita dificuldade na detecção. Os efeitos da baixa luminosidade afetam consideravelmente o desempenho da câmera reduzindo a qualidade das previsões do sistema. Mesmo com baixa iluminação o sistema conseguiu fazer a detecção de alguns degraus. Sendo assim o desempenho do sistema pode ser considerado como satisfatório durante o teste.

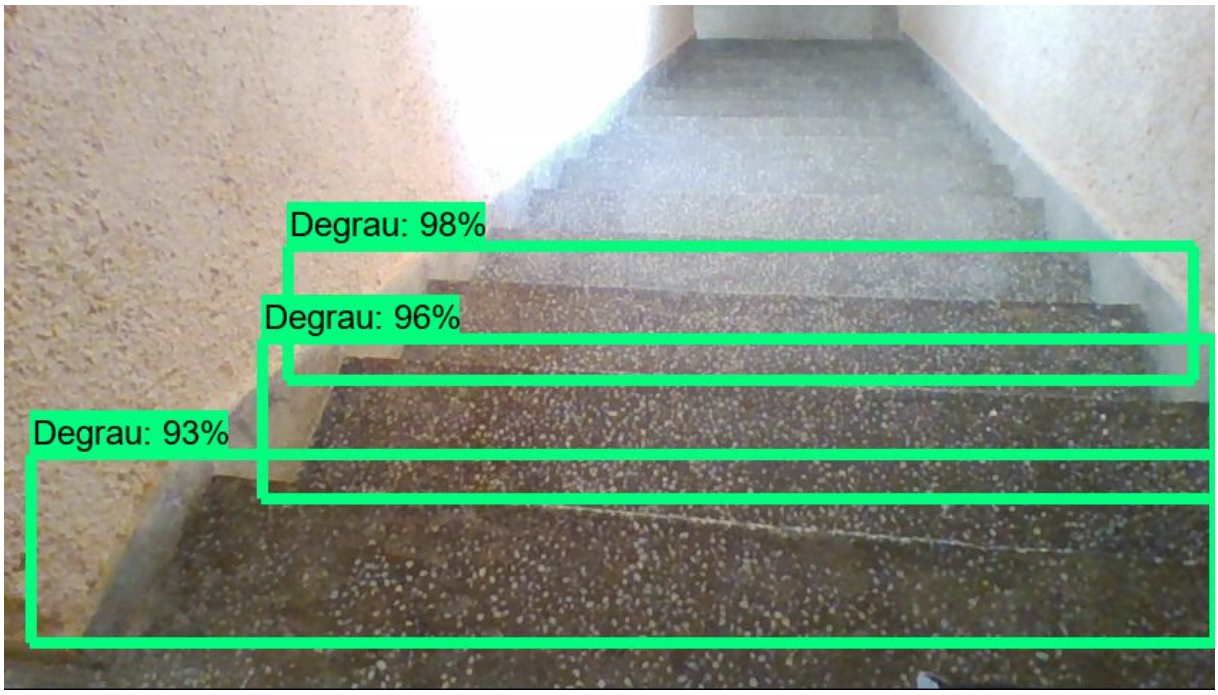
4.3 TERCEIRO EXPERIMENTO

O terceiro e último teste visa analisar o desempenho do sistema a partir de ângulos distintos. Esse teste visa determinar se a precisão das detecções é alterada com a variação dos ângulos entre a câmera e o objeto.

Diferentemente dos experimentos passados, o teste não passará por avaliação de tonalidade de cor dos degraus. Dessa forma o sistema será avaliado apenas em relação a posição da câmera em relação ao objeto alvo, e a precisão da detecção.

Visto que o desempenho do sistema é consideravelmente melhor durante o dia, os testes foram realizados na parte da manhã do dia, a fim de se ter o melhor desempenho do sistema, e consequentemente resultados mais precisos as Figura 23, 24 e 24 caracterizam os resultados do teste.

Figura 23 Três detecções obtidas a partir da visão superior dos degraus



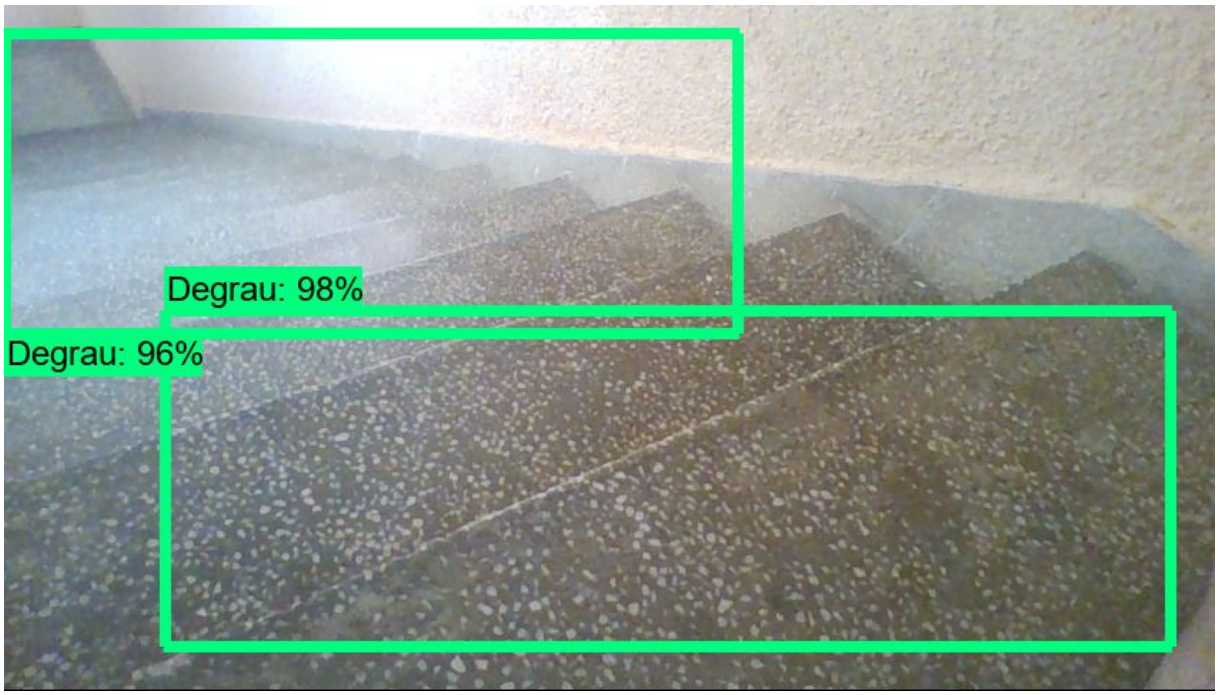
Fonte: Autor

Na Figura 23 podemos observar que a câmera está direcionada para a parte superior do degrau. Nessa imagem vemos um excelente desempenho do sistema, visto que os degraus não possuem variação de tonalidade. Diante disso a rede detectou com precisões de 93%, 96% e 98% três degraus da escadaria.

É possível observar na Figura 23 também, que alguns degraus localizados na parte superior da imagem não foram detectados, porém esses degraus sofrem muita distorção da luminosidade o que pode dificultar a detecção, já que até com a visão humana é difícil de caracterizar essa região da escadaria.

A fim de se ter uma boa avaliação do desempenho da rede, a posição da câmera em relação ao degrau foi modificada, esperando que os resultados sejam similares. Os testes geraram a Figura 24, onde observamos a visão superior esquerda da mesma escadaria da Figura 23,

Figura 24 Imagem superior esquerda da escadaria



Fonte: Autor

Como é possível observar na Figura 24 , A rede teve um pouco de dificuldade em detectar os mesmos degraus da Figura 23. Mesmo tendo essa pequena variação a rede não deixou de detectar degraus na imagem, detectando com precisão de 98% o primeiro degrau do lance de escadas.

Na Figura 24 também notamos uma outra previsão da rede. Nessa imagem é possível observar que o sistema marcou uma região da imagem com probabilidade de 96% de conter um degrau. Essa região é referente ao centro da escadaria e podemos caracterizar como um acerto da rede.

Nota-se que nesse teste ocorreu uma dificuldade devido a troca de angulação na imagem, mas esse resultado não transforma o desempenho do sistema como sendo insatisfatório, visto que além de detectar degraus, não foi dado nenhum resultado inconsistente.

O teste passou por mais uma mudança de posição, nessa ocasião a câmera foi posicionada a frente da escadaria visualizando os dois últimos degraus da escadaria. O resultado desse teste pode ser analisado pela Figura 25.

É possível observar na Figura 25, a mesma escadaria das figuras 23 e 24 nesta situação a câmera está voltada para a parte frontal dos últimos degraus da escadaria. Nessa detecção o sistema conseguiu detectar os dois últimos degraus da escadaria. O degrau inferior, o sistema detectou com precisão de 94%, e o superior o sistema calculou a probabilidade de 88% de existir um degrau naquela região da imagem.

Figura 25 Visão Frontal da escadaria

Fonte: Autor

Com os resultados obtidos nesse teste, o sistema mostrou um resultado excelente diante do experimento, nas modificações de posicionamento da câmera o sistema sofreu alterações no resultado, porém a capacidade de detecção do sistema não deixou a desejar.

Dessa forma após todos os testes, ficou claro o bom desempenho do sistema mesmo ao enfrentar situações adversas obtendo bons resultados em todos os experimentos realizados.

5 CONSIDERAÇÕES FINAIS

Neste trabalho foi apresentado um sistema que utiliza a CV e redes RNA's, para desenvolver um ferramenta que é capaz de detectar degraus. Sendo a aplicação capaz de reconhecer degraus em uma imagem e indicar onde esses degraus se encontram em cada frame do vídeo capturado pela câmera do dispositivo.

Para o desenvolvimento de uma aplicação que utilize CV, foi utilizado a biblioteca OpenCV, para a aquisição das imagens e processamento *frame* a *frame* do vídeo. O sistema desenha um retângulo na região da imagem em que se encontra o degrau. Para o desenvolvimento, o sistema passou por três etapas: aquisição de imagens de treino. Rotulação das imagens e treinamento da rede.

O sistema utilizou o conceito de DL para detecção dos degraus. Utilizando o modelo de rede *MobileNet*, a rede foi testada em três situações distintas. Primeiramente a rede foi exposta a um ambiente com boa iluminação em duas condições, uma contendo degraus que possuíam variação de tonalidade de cor e outra contendo degraus que não variam em sua tonalidade de cor. O segundo teste consistiu em expor o sistema a um ambiente com iluminação desfavorável, passando pelas duas situações dos testes anteriores. E por fim o terceiro teste visava analisar a precisão da rede em detectar degraus em ângulos diferentes.

No primeiro teste a rede obteve um desempenho satisfatório, conseguindo detectar os degraus de forma precisa, porém algumas das classificações foram distintas do que era esperado. Já o segundo teste, obteve resultados razoáveis, visto que a falta de luminosidade afetou consideravelmente o desempenho da câmera e consequentemente do sistema. Por fim o terceiro teste obteve um excelente resultado, a rede conseguiu detectar com precisão os degraus mesmo com angulações diferentes.

Após a execução de todos os testes, foi possível se analisar que o sistema cumpriu com seu objetivo. A rede obteve um bom desempenho até mesmo quando esteve em situações que não são favoráveis no processamento de imagens como por exemplo baixa luminosidade. Dessa forma o sistema se mostrou eficiente em todos os testes.

5.1 TRABALHOS FUTUROS

Diante dos resultados dos testes com o sistema proposto, é possível se perceber algumas melhorias que podem ser realizadas em trabalhos futuros. Umas das tentativas seria aumentar a precisão das detecções do sistema utilizando um conjunto de dados maior.

Outra modificação seria o desenvolvimento de uma rede neural especifica para realizar a detecção dos degraus. Essa modificação deixaria o sistema menos pesado e aumentaria precisão nas previsões realizadas pelo sistema.

Uma das implementações possíveis seria o cálculo da distância entre o degrau e a câmera, para isso poderia ser utilizado um sensor sonar, ou até mesmo uma câmera estéreo.

Com o bom funcionamento da aplicação, pode-se delegar outras tarefas ao sistema, como por exemplo diferenciar uma escadaria de uma rampa de acesso, o que poderia ser uma excelente aplicação para cadeira de rodas autônomas.

REFERENCIAS

ABNER, G. H. *et al.* EFA Global Monitoring Report Regional Overview-Monitoring the Education for All goals:Sub-saharan Africa. **Journal of Visual Impairment & Blindness**, v. 1, n. 2, p. 1–14, 2010.

ACADEMY, E. D. S. **O QUE É VISÃO COMPUTACIONAL?** Disponível em: <http://datascienceacademy.com.br/blog/o-que-e-visao-computacional/>. Acesso em: 5 jun. 2018.

COSTA, R. **App utiliza câmera para descrever situações e objetos de forma inusitada**. Disponível em: <https://www.tecmundo.com.br/apps/102244-app-utiliza-camera-descrever-situacoes-objetos-forma-inusitada.htm>. Acesso em: 16 set. 2017.

DENG, L.; YU, D. Deep Learning: Methods and Applications. **Foundations and Trends® in Signal Processing**, v. 7, n. 3–4, p. 197--387, 2013.

ESTEU, B. R. M. Clusterização de dados de vibração na perfuração de poços de petróleo através de redes neurais não supervisionadas. p. 47–54, 2014.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. **Inteligência Artificial: Uma Abordagem de Aprendizado de MáquinaLivros Técnicos e Científicos**, 2011.

FERREIRA, S. Redes Neurais Convolucionais Profundas na Detecção de Plantas Daninhas em Lavoura de Soja. p. 70, 2017.

FLECK, L.; TAVARES, M. H. F.; EYNG, E.; HELMANN, A. C.; ANDRADE, M. A. DE M. Redes Neurais Artificiais: Princípios Básicos. **Revista Eletrônica Científica Inovação e Tecnologia**, v. 1, n. 13, p. 47–57, 2016.

FRED, L. **O Machine Learning e seus principais algoritmos de aprendizagem supervisionadale**. Disponível em: <https://pt.linkedin.com/pulse/o-machine-learning-e-seus-principais-algoritmos-de-luís-fred>.

GLOBO. **Preparar um cão guia custa em média 30mil e demora cerca de 2 anos**. Disponível em: <http://g1.globo.com/globo-news/noticia/2011/08/preparar-um-cao-guia-custa-em-media-r-30-mil-e-demora-cerca-de-2-anos.html>. Acesso em: 17 set. 2017.

GOMEZ, A. **Deep Learning In Digital Pathology**. Disponível em:

<http://www.global-engage.com/life-science/deep-learning-in-digital-pathology/>. Acesso em: 7 jun. 2018.

HAYKIN, S. Redes neurais: princípios e prática. **Bookman**, p. 900, 2001.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. **ArXiv**, p. 9, 2017.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet Classification with Deep Convolutional Neural Networks. **Advances In Neural Information Processing Systems**, p. 1–9, 2012.

MARTÍN, A. *et al.* **TensorFlow: Aprendizado de Máquina em Grande Escala em Sistemas Heterogêneos**. Disponível em: <https://www.tensorflow.org/>.

MATTHIJS HOLLEMANS. **MobileNet version 2**. Disponível em: <http://machinethink.net/blog/mobilenet-v2/>.

MCCULLOCH, W. S.; PITTS, W. A Logical Calculus of the Idea Immanent in Nervous Activity. **Bulletin of Mathematical Biophysics**, v. 5, p. 115–133, 1943.

MICROSOFT. CNTK 103: Parte A - Carregador de Dados MNIST. 2017.

MOACIR A. PONTI; COSTA, G. B. P. DA. **Como funciona o Deep Learning**. [s.l: s.n.].

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre Aprendizado de Máquina. **Sistemas inteligentes: fundamentos e aplicações**, p. 89–114, 2003.

OLIVEIRA, H. S. DE. Redes Neurais Convolucionais para Classificação de Expressões Faciais de Emoções. 2017.

PETER, N.; RUSSELL, S. **Inteligência Artificial**. [s.l: s.n.].

RODRIGUES E SILVA, S.; SCHIMIDT, F. Redução de variáveis de entrada de redes neurais artificiais a partir de dados de análise de componentes principais na modelagem de oxigênio dissolvido. **Quimica Nova**, v. 39, n. 3, p. 273–278, 2016.

RUDER, S. An overview of gradient descent optimization algorithms. 2016.

SZEGEDY, C. **Building a deeper understanding of images**. Disponível em:

<https://ai.googleblog.com/2014/09/building-deeper-understanding-of-images.html>. Acesso em: 24 maio. 2018.

SZELISKI, R. Computer Vision : Algorithms and Applications. **Computer**, v. 5, p. 832, 2010.

TALEB, A.; FARIA, M.; AVILA, M.; MELLO, P. As condições da saúde ocular no Brasil. **Journal of Chemical Information and Modeling**, v. 53, n. 9, p. 1689–1699, 2013.

TZUTALIN. **LabelImg**. Disponível em: <https://github.com/tzutalin/labelImg>. Acesso em: 19 jun. 2018.

YANGQING, J.; EVAN, S.; JEFF, D.; SERGEY, K.; JONATHAN, L.; ROSS, G.; SÉRGIO, GUADARRAMA TREVOR, D. Caffe: Convolutional Architecture for Fast Feature Embedding. **arXiv preprint arXiv:1408.5093**, 2014.

PETER, N.; RUSSELL, S. Inteligência Artificial. [s.l: s.n.].