UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TÉCNICAS DE RECONHECIMENTO DE IMAGEM PARA INCORPORAÇÃO EM FERRAMENTAS DE AUXÍLIO A DEFICIENTES VISUAIS

RENAN GALEANE ALBOY

ORIENTADOR: PROF. DR. MÁRCIO MERINO FERNANDES

São Carlos – SP

Junho/2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Renan Galeane Alboy, realizada em 26/06/2019:

Prof. Dr. Márcio Merino Fernandes
UFSCar

Profa. Dra. Marcela Xavier Ribeiro
UFSCar

Prof. Dr. Jose Fernando Rodrigues Junior
USP

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Jose Fernando Rodrigues Junior e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

Prof. Dr. Márcio Merino Fernandes

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TÉCNICAS DE RECONHECIMENTO DE IMAGEM PARA INCORPORAÇÃO EM FERRAMENTAS DE AUXÍLIO A DEFICIENTES VISUAIS

RENAN GALEANE ALBOY

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Arquitetura de Computadores
Orientador: Prof. Dr. Márcio Merino Fernandes

São Carlos – SP

Junho/2019

A pessoas queridas e livros.

AGRADECIMENTOS

Primeiramente agradeço a meus pais e irmão, Celso, Marcia e Felipe, avós e familiares ao apoio incondicional desde o começo dos meus estudos e incentivos para continuar neste caminho e me dar conselhos durante trajeto. Também sou muito grato a Vitória e familiares pelo carinho, apoio e boas conversas. A todas estas pessoas me faltam palavras para descrever a importância de suas presenças em minha vida.

Sou muito grato e feliz pela orientação do Professor Márcio que sempre se demonstrou disposto e deu todo apoio para o desenvolvimento do trabalho ajudando desde a escolha do tema, decisões de projeto, caminhos a seguir e correções de texto. Graças a toda a dedicação e ajuda o trabalho foi desenvolvido. Não posso deixar de falar do Professor Emerson e ao doutorando Antônio que apresentara o projeto trabalhado. E claro, agradecer ao Antônio pela paciência de sanar as dúvidas que foram surgindo e auxiliar na realização dos testes com o protótipo.

Sem esquecer claro do Marrocos e do Lucas que são grandes amigos que fiz durante o mestrado. Compartilhando desde o interesse em se reunir para estudar, discutir e aprofundar nas matérias que cursamos até as partidas de Magic e mesas de RPG.

Não posso deixar de mencionar a histórias em livros e quadrinhos. Estas sempre presentes no meu dia-a-dia e me levando para os mais diferentes mundos. Elas foram fundamentais para os mais diferentes momentos e além de auxiliar a perceber diferentes perspectivas me ensinou o quanto a ficção pode ajudar na realidade.

Aquele que luta com monstros deve acautelar-se para não tornar-se também um monstro. Quando se olha muito tempo para um abismo, o abismo olha para você.

Friedrich Nietzsche

RESUMO

A deficiência visual pode causar muitas limitações a uma pessoa, o que inclui trabalhar em tarefas domésticas básicas, atividades profissionais, assim, a percepção do mundo que os cerca é bem diferente. O advento de novas tecnologias permite o design e a exploração de novos tipos de dispositivos de auxílio para deficientes visuais. As técnicas de reconhecimento de imagem são usadas há muito tempo como uma ferramenta para compensar ou complementar as limitações visuais humanas. Mais recentemente, os avanços nas técnicas de aprendizado de máquina, como a aprendizagem profunda, ofereceram novas possibilidades para construir dispositivos de ferramentas de ajuda que podem desempenhar um papel significativo na redução das limitações sofridas pelos cegos. Este trabalho apresenta o desenvolvimento de um módulo de reconhecimento de imagem para ser usado como parte de uma ferramenta de auxílio existente para pessoas com deficiência visual. O módulo é baseado em redes neurais convolucionais e é inicialmente direcionado para o reconhecimento do estado de objetos em ambientes internos.

Palavras-chave: deep learning, redes neurais, deficiência visual, caffe.

ABSTRACT

Visual impairment can cause many limitations to a person, which includes working in basic home duties, professional activities, and also changing the way the world is perceived by him or her. The advent of new technologies allows the design and exploration of new kinds of aid devices for the visually impaired. Image recognition techniques have long been used as a tool to compensate or complement human visual limitations. More recently, advances in machine learning techniques, such as deep learning, have offered new possibilities to build aid tools devices that can play a significant role in reducing the limitations suffered by blind people. This work presents the results related to the development of an image recognition module to be used as part of an existing embedded aid tool for visually impaired people. The module is based on convolutional neural networks, and is initially targeted to the recognition of the state of indoor home objects.

Keywords: deep learning, neural network, visual problems, caffe.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO

Este capítulo tem como objetivo apresentar o contexto e a motivação do trabalho desenvolvido. Também serão apresentados os objetivos e métodos de pesquisa utilizados, além das principais contribuições e organização do trabalho.

1.1 Contexto e Objetivos

Na área de reconhecimento de imagens existem muitos tópicos a serem estudados ainda. No que diz respeito à classificação e identificação de objetos, há muitas abordagens à disposição para que se possa explorar a área. Uma dessas abordagens consiste na aplicação de tecnologias com o objetivo de auxiliar deficientes visuais a perceberem melhor o mundo que os rodeia. Na bibliografia estudada, as abordagens mais recentes fazem o uso do aprendizado de máquina, utilizando tanto redes neurais convencionais, como também técnicas de *deep learning* com camadas convolucionais para realizar o treinamento com bases de imagens a serem identificadas. O trabalho propõe o estudo e avaliação de técnicas de *deep learning* para o reconhecimento de imagens, explorando não apenas a classificação do objeto, mas também seus estados, por exemplo, identificando se uma porta está aberta ou fechada. A partir disso, avaliar quantitativamente e qualitativamente a classificação de imagens utilizando *deep learning* e desenvolver um novo módulo com esta finalidade para ser incorporado no protótipo de auxílio a deficientes visuais (DOURADO; PEDRINO, 2018). Assim, o trabalho apresenta a finalidade de auxiliar a movimentação destes em ambientes internos reconhecendo objetos como portas, gavetas, cadeiras, mesas, geladeiras, fogões, escadas e armários.

1.2 Objetivos Gerais

O trabalho tem como objetivo geral o estudo de métodos de reconhecimento de objetos utilizando técnicas de *deep learning* para o desenvolvimento de um classificador para ambientes internos, utilizando como ambiente de testes o Departamento de Ciência da Computação da Ufscar e classificando não só uma série de objetos mas também seus estados, como aberto e fechado. O desenvolvimento do classificador para estes objetos visa ser aplicado ao protótipo (DOURADO; PEDRINO, 2018) de auxílio a deficientes visuais que tem a função de identificar caminhos livres e obstáculos durante o percurso do usuário. O protótipo será descrito de forma mais detalhada no capítulo 4. Este trabalho pode impactar diretamente na vida de pessoas com deficiências visuais, aumentando a percepção do ambiente que as cerca, proporcionando também para o meio científico um estudo que mostre a viabilidade da utilização de técnicas de redes neurais profundas para a classificação de imagens.

1.3 Objetivos Específicos

Entre os objetivos específicos do projeto está a criação de bases de imagens para o treinamento dos algoritmos, a realização de testes em laboratório para simular o funcionamento do módulo e um teste prático com a utilização do protótipo do aparelho embarcado com o módulo desenvolvido. A construção das bases foi planejada para estabelecer parâmetros para serem utilizados no treinamento, sendo feitas variações na quantidade de imagens para cada classe na base e nas quantidades de épocas treinadas. A simulação visa testar o treinamento dos algoritmos que serão utilizados, sendo a escolha destes algoritmos baseado no levantamento bibliográfico realizado, e a validação dos que apresentarem melhores valores de acurácia para a criação de um novo módulo de classificação de imagens para o protótipo (DOURADO; PEDRINO, 2018). Por fim, a utilização do módulo desenvolvido embarcado para medir não só a sua acurácia, mas também o seu tempo de resposta na classificação de objetos.

1.4 Método de Trabalho

A proposta inicial do trabalho foi o desenvolvimento de um módulo de identificação de objetos para o protótipo de auxílio a deficientes visuais (DOURADO; PEDRINO, 2018) com o objetivo de realizar a implementação do reconhecimento de estado em algumas categorias de objetos ao serem identificados. Para a realização do objetivo foi feito a utilização de CNN's e também testado a classificação com método diferente, assim comparando o desempenho de

cada método testado com o alcançado com o uso de CNN e verificando se é o melhor e o método mais adequado para ser utilizado. Também foi visada a portabilidade da solução desenvolvida para a placa utilizada no projeto do protótipo.

O método de trabalho que foi utilizado para o desenvolvimento deste projeto de mestrado começou com o levantamento bibliográfico sobre o tema de interesse. As primeiras atividades foram compostas por um levantamento dos métodos mais usados para a resolução dos problemas da área e o estudo das tecnologias relacionadas a bibliotecas, ferramentas e técnicas que podem ser utilizadas como abordagem para o trabalho.

Na etapa posterior foi feita a utilização das técnicas. Em seguida, foram aplicados os métodos que apresentaram melhor desempenho como solução do problema, sendo realizados testes para obter dados do que foi utilizado. Por fim, foram analisados os resultados obtidos nas atividades, verificado sua validade, realizada a escrita e defesa da dissertação.

1.5 Justificativa do Trabalho

O que foi proposto no trabalho pode apresentar um avanço nas tecnologias para auxílio a pessoas com deficiências físicas, no caso, com deficiência visual. Além disso, também pode auxiliar no refinamento de soluções e de ferramentas para o reconhecimento de imagens por meio do levantamento de informações de quais técnicas apresentariam melhor desempenho e quais condições mais indicadas para o seu uso.

1.6 Organização do Trabalho

Este trabalho está organizado da seguinte forma: no Capítulo 2 é apresentado o levantamento bibliográfico com os principais conceitos estudados e relacionados ao trabalho. No Capítulo 3 são apresentados trabalhos e abordagens que se relacionam com este trabalho de pesquisa. No capítulo 4 é mostrada a estrutura do protótipo em que será implementado o projeto e será descrita a sua arquitetura, o seu hardware e funcionamento, explorando também o novo módulo que foi adicionado ao protótipo. No capítulo 5 são apresentados os testes realizados e outras informações acerca do desenvolvimento do trabalho. Por fim, no capítulo 6 são apresentadas conclusões.

Capítulo 2

FUNDAMENTOS E TEORIAS

Este capítulo apresenta os principais conceitos relacionados ao trabalho desenvolvido, como o de imagem digital, *machine learning*, redes neurais e *deep learning*. Também são apresentadas abordagens de trabalho, algoritmos mais utilizados e as ferramentas que podem ser utilizadas para a aplicação destas tecnologias.

2.1 Imagem digital

Uma imagem pode ser definida como uma função bidimensional, f(x,y), em que x e y correspondem a pontos no plano e formam uma coordenada em um plano. Outro valor que é relacionado a cada coordenada da imagem é sua amplitude, que corresponde a intensidade ou ao tom de cinza em um ponto na imagem (GONZALEZ; WOODS, 2000). A imagem digital é composta por um número finito de elementos, cada um possuindo uma localização e intensidade especifica. Esses elementos são chamados de pixels.

Os pixels podem assumir valores entre 0 e 1, no caso de uma imagem binária, mas também podem estar entre 0 e 255 para imagens em tons de cinza. Já no caso de imagens coloridas, é adotado o sistema de canais RGB (*red*, *green*, *blue*), em que a cor do pixel é definida pelo valor de cada componente da cor. A Figura 2.1 mostra um exemplo de imagem digital.

Como definido em (FILHO; NETO, 1999), a imagem digital pode ser obtida a partir de uma cena real. Primeiramente é necessária a transformação da imagem, originalmente em 3-D, em uma imagem 2-D, com redução em suas dimensões, e pode ser obtida por uma gama de dispositivos como câmeras fotográficas e de vídeos. No caso da imagem colorida é necessário um conjunto de filtros de cor para compor as componentes R (*red*), G (*green*), B (*blue*), sendo os canais mesclados entre si para realizar a composição das cores necessárias para a replicação

da cena real.

Figura 2.1: Representação de uma imagem digital (FILHO; NETO, 1999).

2.2 Aprendizado de máquina

O uso de imagens digitais propicia várias vantagens em relação aos demais tipos de imagem. Devido a sua composição, feita por pixels que são definidos em eixos x e y, podem ser trabalhadas em ambientes computacionais e permitem o uso de algoritmos mais sofisticados para sua manipulação. Dentre as formas de manipulação deste tipo de imagem, pode-se destacar as técnicas de aprendizado de máquina.

O aprendizado de máquina compreende o uso de técnicas que visam a utilização de algoritmos por meio dos quais um programa de computador pode aprender a partir dos dados inseridos. Assim, um programa de computador pode aprender a partir de dados de treinamento para realizar atividades sobre uma determinada classe em que ocorreu o treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). Entre as tarefas que o aprendizado de máquina permite realizar está a classificação, reconhecimento de padrões, previsões e muitas outras.

Para que seja possível realizar a execução destes algoritmos e para que alcancem o seu objetivo, existe uma série de escolhas a serem feitas. São decisões como qual algoritmo que será executado, qual abordagem para o aprendizado será escolhida e definições das etapas para os resultados alcançarem o objetivo desejado.

2.2.1 Abordagens para o aprendizado

Os algoritmos podem ser classificados segundo a sua classe de aprendizagem que são supervisionados, não supervisionado e reforçado. As classes são descritas nos tópicos abaixo.

2.2.1.1 Supervisionada

A abordagem de aprendizado supervisionada é uma das classes de aprendizado de máquina. Os algoritmos supervisionados fazem uso de dados já classificados para identificar outros que ainda não estão (ZOCCA et al., 2017). Ou seja, existem classes pré-definas que podem identificar dados com base nas características dos dados já treinados, sendo definidas saídas especificas para as classes de dados que foram aprendidos. Assim, essa classificação apresenta uma separação bem distinta entre os dados analisados, como mostra a Figura 2.2.

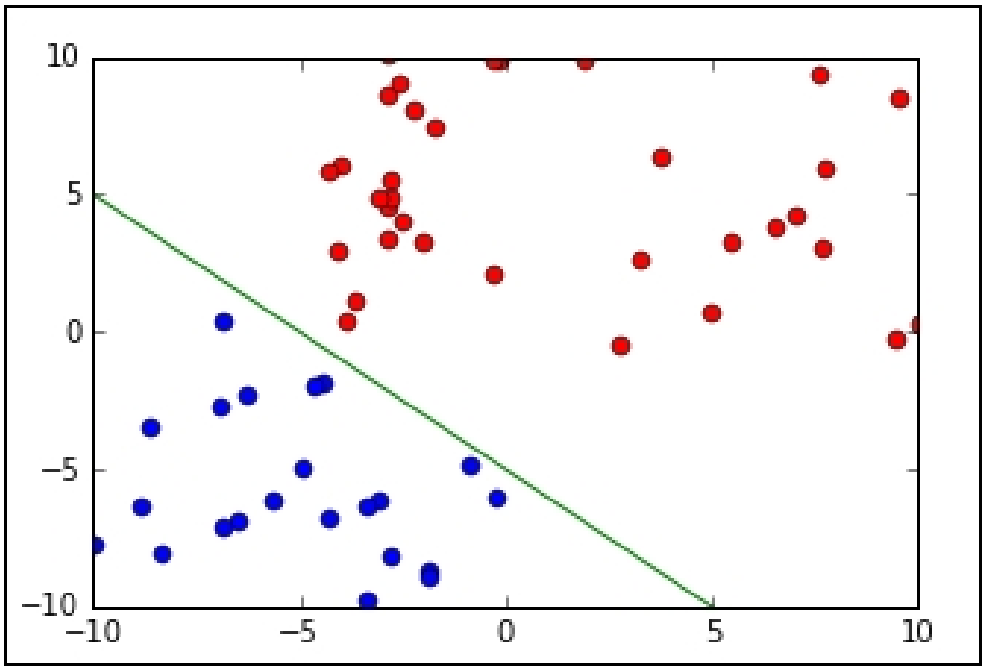


Figura 2.2: Exemplo de gráfico gerado por aprendizado supervisionado (ZOCCA et al., 2017).

Um exemplo do algoritmo supervisionado pode ser a identificação e-mails (ZOCCA et al., 2017). Pode ser utilizada a separação entre os e-mails importantes e os spans a partir da análise do corpo de cada um e fazendo com que os que forem classificados como spams sejam descartados.

2.2.1.2 Não supervisionado

Diferentemente da classe supervisionada, os algoritmos não supervisionados não recebem a pré-classificação dos dados que estão recebendo. Assim, deixando para o algoritmo identificar características semelhantes nos dados (ZOCCA et al., 2017). Um dos exemplos do uso dessa classe é o agrupamento. Esta técnica, que não utiliza supervisão tenta separar os dados em subconjuntos, como mostra a Figura 2.3.

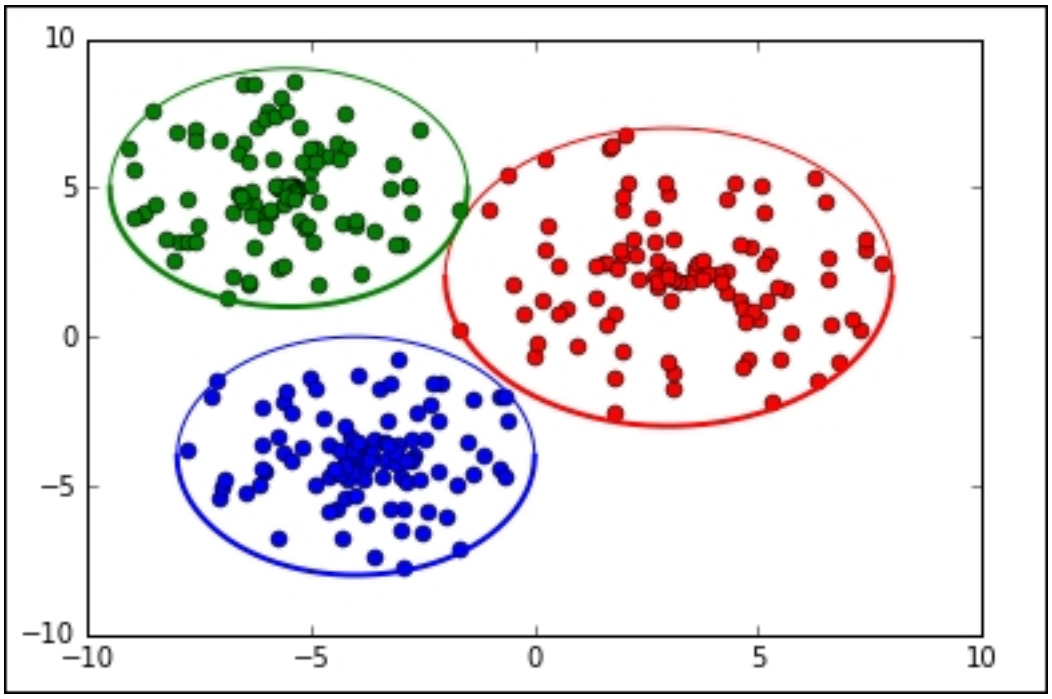


Figura 2.3: Exemplo de gráfico gerado pelo aprendizado não supervisionado (ZOCCA et al., 2017).

2.2.1.3 Reforçada

A classe de aprendizado reforçada funciona de forma diferente da supervisionada. Para aumentar seu desempenho ela faz uso do *feedback* dos elementos analisados, não os classificando como certos ou errados, mas sim recebendo pontos por suas ações de forma a ganhar ao acertar e perder no caso de erro (ZOCCA et al., 2017).

Assim, assemelha-se a um jogo, de modo que o algoritmo utilize ações já realizadas para tentar novos acertos. Porém, ao analisar dados que entram na área do desconhecido, devem mudar a abordagem para tentar alcançar um caminho de acertos.

2.2.2 Tópicos do aprendizado de máquina

Na seção 2.2.1 foram apresentadas diferentes abordagens para o aprendizado de máquina. Outro aspecto importante do aprendizado de máquina é o entendimento dos dados e dos processos, como o tratamento dos dados e a criação do caso de teste, que são realizados antes da etapa de aprendizagem. Esses e outros pontos são descritos no livro Phyton Deep Learning (ZOCCA et al., 2017), conforme elencados abaixo.

- Aprendizado: Técnica que será utilizada para o aprendizado de máquina. A técnica escolhida é diretamente relacionada com a escolha do algoritmo, pois impactará diretamente no desempenho do aprendizado.
- Treinamento: Base de interesse com os dados que servirão para a aprendizagem. Nesta etapa é definido o tipo de aprendizado, se é supervisionado, não supervisionado ou reforçado, sendo importante para o entendimento da estrutura do problema.
- Representação: Dado em termos de elementos (*features*) identificados no aprendizado. Uma boa escolha de representação impacta em resultados melhores.
- Objetivo: Representa a razão para a aprendizagem e está diretamente relacionada com o alvo. Também ajuda a definir pontos fundamentais, como o tipo de aprendizado que será utilizado e qual a forma de representação é mais interessante.
- Alvo: Representa o que será usado para a entrada do aprendizado e quais as saídas que serão identificadas. Pode ser a classificação de um dado sem a etiqueta, a representação de dados de entrada de acordo com parâmetros ou características, uma simulação para futuras predições ou também a estratégia que é usada para o aprendizado reforçado.

Mesmo com um objetivo e alvo definido, os algoritmos de aprendizagem de máquina não são exatos; assim, os resultados obtidos são aproximados, pois não são numericamente exatos (ZOCCA et al., 2017). Para a obtenção dos dados que são utilizados nos processos descritos anteriormente faz-se uso de outra série de processos, que são mostrados na Figura 2.4.

- Coleção de dados: Implica na coleta da maior quantidade possível de dados e, no caso do aprendizado supervisionado, na marcação das classes dos objetos selecionados.
- Processamento dos dados: Implica na limpeza dos dados coletados, sendo removidas as redundâncias, preenchidos os dados faltantes ou alteradas a relação entre eles.

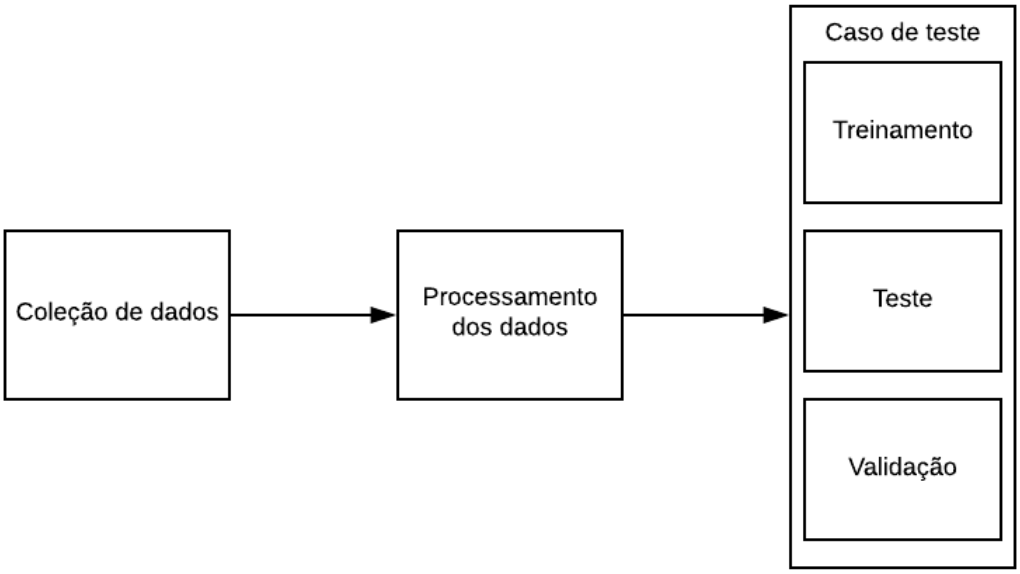


Figura 2.4: Processo para o tratamento dos dados.

- Criação do caso de teste: Os dados obtidos nos processos anteriores podem ser divididos em grupos, como um conjunto de treinamento e um de teste de conjunto de dados. O primeiro grupo serve para treinar o algoritmo e o segundo para avaliar o algoritmo e a precisão que foi alcançada com o treinamento. Também pode ser criado um terceiro grupo, o de validação do conjunto, que pode ser utilizado após o treinamento e teste e que verifica os resultados obtidos nos processos anteriores.

As técnicas de aprendizado de máquina não resultam em um resultado preciso, apenas aproximado. Por esse motivo é importante a divisão nos grupos descritos anteriormente. Além disso, o uso de uma mesma base de imagem pode resultar em um algoritmo com problema de *overfiting*, gerando um resultado com alta precisão para a base em que foi treinado, porém com resultado impreciso em bases de dados similares à utilizada no treinamento (ZOCCA et al., 2017).

2.2.3 Principais algoritmos

Os algoritmos abaixo são descritos no livro Python deep learning (ZOCCA et al., 2017).

2.2.3.1 Regressão linear

Os algoritmos de regressão linear são um tipo de aprendizagem supervisionada que utiliza dados dos valores de entrada para prever um valor. Neste tipo de algoritmo o objetivo é minimizar uma função de custo encontrando parâmetros apropriados para a função nos dados de entrada que mais se aproximam do valor alvo. A função utilizada é uma função de erro, que é o

quão longe o valor atual está do resultado correto. Uma função que pode ser utilizada é o erro quadrático médio, mostrado na Equação 2.1, em que é tomado o quadrado da diferença entre o resultado esperado e o valor previsto. A partir do valor obtido pelos exemplos de entrada é obtido o erro do algoritmo e representa a função de custo.

$$E_i(Yi\tilde{}ti)^2 \tag{2.1}$$

A utilização de algoritmo de regressão linear em aplicações pode ser observada no trabalho de (SHAILAJA; ANURADHA, 2016) que visa realizar o reconhecimento facial. Para cumprir tal objetivo, o trabalho faz a utilização de *deep learning* juntamente com regressão linear. O *deep learning* é utilizado para uma análise mais completa da imagem, assim melhorando o desempenho do uso da regressão linear.

2.2.3.2 Árvore de Decisão

O algoritmo de árvore de decisão (QUINLAN, 1986) também apresenta o tipo de aprendizado supervisionado e cria um classificador em forma da estrutura de uma árvore de dados. A árvore de decisão é composta por nós de decisão nos quais ocorrem os testes com atributos específicos e os nós folhas indicam o valor destino. Este tipo de algoritmo classificador funciona a partir de um nó raiz e percorrendo os demais nós a partir da avaliação dos valores até chegar aos nós folhas, que são nós que não possuem outros após eles. Em (ZHANG et al., 2017b) é utilizada a árvore de decisão na utilização da técnica Random Forests, que é um método de classificação, que pode ser utilizado para estimar a pose de objetos, classificar imagens e realizar a detecção. O trabalho visa prever a localização de um objeto em uma sequência de frames, utilizando um modelo de configuração livre do algoritmo.

2.2.3.3 K-Means

O algoritmo k-mean (MACQUEEN et al., 1967) apresenta a característica de agrupamento e o tipo de aprendizado é não supervisionado. A técnica de agrupamento mais comum é o clustering k-means e agrupa cada elemento em um conjunto de dados que se dividem em k subconjuntos distintos. O algoritmo consiste na escolha de k pontos aleatórios que representem os centros dos subconjuntos k, que são chamados de centroides. Para cada centroide selecionado será criado um novo subconjunto diferente. Então é recalculado centro e feito o processo até que os centros localizados não sofram mais alterações. Para que a técnica funcione, é importante que sejam identificadas métricas para calcular a distância entre os pontos. Um exemplo de utilização do

algoritmo k-means é a análise da imagem feita pela aplicação desenvolvida por (ZHANG et al., 2017a). A aplicação tem o objetivo de detecção de pedestres para monitoramento inteligente, robótica, direção inteligente e outros. O funcionamento é realizado a partir da obtenção da área de interesse extraída por uma rede neural convolucional, sendo utilizado k-means para áreas que podem conter pedestres. Assim, a rede de detecção identifica e classifica a imagem selecionada.

2.2.3.4 Naive Bayes

O algoritmo de Naive Bayes se difere dos demais pela abordagem probabilística que ele utiliza. A maioria dos algoritmos de aprendizado de máquina utiliza a abordagem de determinar a chance de um evento Y ocorrer a partir de condições X, como mostra a Equação 2.2.

$$p(Y/X) \tag{2.2}$$

Naive Bayes é chamada de abordagem generativa e trabalha com tais informações de forma oposta, ou seja, sabendo qual é o evento Y e qual a probabilidade de X, buscando dessa maneira a probabilidade de gerar instancias de X com Y. Essa abordagem é representada na Equação 2.3.

$$p(X|Y) = p(Y|X) * p(X)/p(Y) \tag{2.3}$$

Por exemplo, com a utilização de Naive Bayes pode ser calculada a probabilidade de uma determinada configuração de pixels representar uma imagem, levando-se em consideração que já são sabidas a probabilidade e existência da imagem desejada e que uma configuração aleatória de pixels pode corresponder a ela. Um exemplo da utilização de Naive Bayes pode ser observado em (LI; SONG; LUO, 2017), em que a combinação com filtros permite a classificação de imagens com mais de uma etiqueta, ou seja, a partir de uma imagem de entrada vários objetos podem ser identificados.

2.2.3.5 Suporte a Máquinas Vetoriais

O algoritmo de suporte a máquinas vetoriais (SVM) (CORTES; VAPNIK, 1995) utiliza o aprendizado supervisionado e é utilizado principalmente na atividade de classificação. Diferentemente dos demais algoritmos de aprendizado de máquina que separam os dados em classes, o SVM busca uma separação em hiperplanos, assim maximizando a margem que separa cada ponto. Além disso, também pode trabalhar com dados que não são separáveis linearmente, uti-

lizando duas estratégias para lidar com este tipo de dado. A primeira é a introdução de margens flexíveis e a segunda é chamada de manipulação do *kernel*.

A introdução de margens flexíveis auxilia a identificação, permitindo que dados sem classificação mantenham propriedades preditivas do algoritmo e possibilitando que ele ainda seja classificado. Por meio da manipulação de *kernel* ocorre o mapeamento de espaço de recurso em outros espaços em que poderia ser definido o hiperplano; assim, quando mapeado de volta, a sua característica é a de um hiperplano não linear, possibilitando a separação de elementos (que anteriormente demonstravam ser inseparáveis) em grupos.

Um exemplo de aplicação de SVM pode ser visto em (ZHU et al., 2017). O trabalho desenvolve uma tecnologia para sonares de veículos subaquáticos para o reconhecimento de alvos de forma automática. O sonar dos veículos é responsável pela obtenção das imagens que tem suas características extraídas por uma rede neural convolucional e a classificação delas é feita por meio do algoritmo de SVM.

2.2.3.6 Método Entropia cruzada

O algoritmo de entropia cruzada (BOER et al., 2005), diferente dos demais que utilizam a aprendizagem supervisionada ou não supervisionada, faz uso da aprendizagem reforçada. Esta técnica visa resolver problemas de otimização para encontrar parâmetros tanto para maximizar como para minimizar uma função.

O método funciona a partir de uma amostra aleatória das variáveis que serão otimizadas, que podem ser, por exemplo, os pesos de uma rede neural. Então as tarefas são executadas e os valores obtidos são armazenados. Na etapa seguinte, as melhores execuções são identificadas e são selecionadas as variáveis com melhor desempenho para que possam ser usadas para o cálculo de novas variáveis e para que uma nova amostra possa ser gerada. O processo se repete até que se atinja a condição de parada, o valor desejado, por exemplo, ou até que o sistema pare de otimizar os valores.

Um exemplo de utilização da entropia cruzada pode ser observado em (HU et al., 2017). O trabalho tem como objetivo estimar a idade a partir da comparação de duas imagens da mesma pessoa. O cálculo da estimativa é feito utilizando uma rede neural convolucional que recebe como entrada um par de imagens de um mesmo indivíduo retiradas com idades diferentes. A entropia cruzada, combinada com outras funções de estimativa de idade, é utilizada para medir o valor correto das imagens que foram usadas como entrada.

2.3 Redes Neurais

As redes neurais são um algoritmo para a aplicação do aprendizado de máquina, porém seguem uma abordagem diferente dos algoritmos já citados na seção 2.2.3. Enquanto os outros métodos utilizam probabilidade e estatística, os algoritmos de redes neurais buscam imitar as estruturas do cérebro humano (SUGOMORI et al., 2017). A estrutura cerebral que o algoritmo imita é composta por neurônios que através dos dendritos recebem os impulsos e os transmitem para o próximo neurônio pelo axônio, como mostra a Figura 2.5.

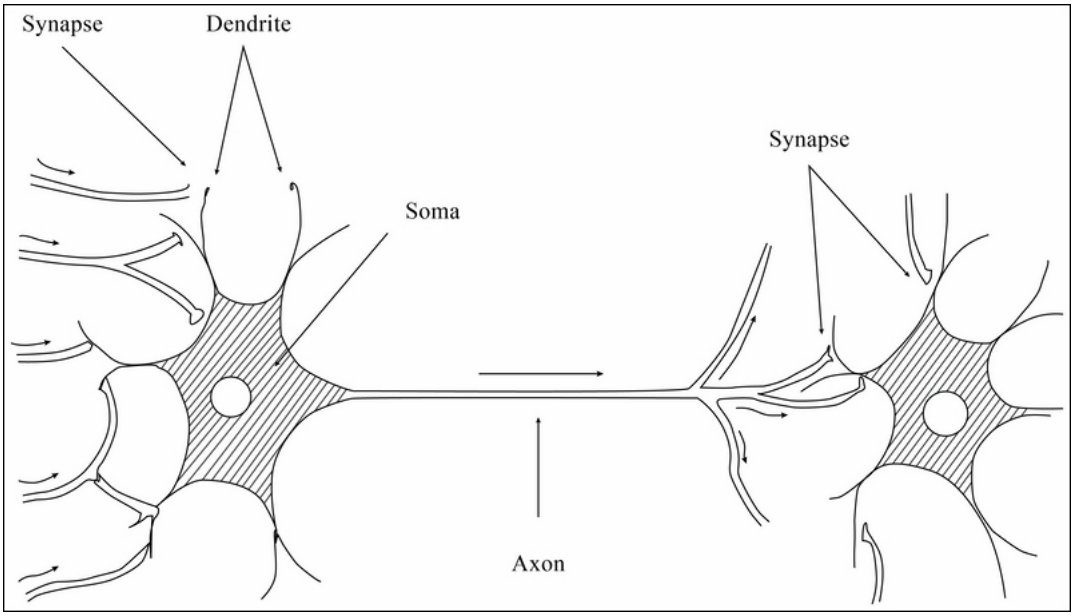


Figura 2.5: Estrutura do cérebro humano (SUGOMORI et al., 2017).

Uma rede neural é dividida em camadas, a camada de entrada, camadas internas e a de saída. A camada de entrada serve como a de entrada da informação, a interna serve como um ponto de tomada de decisão e a de output para a representação do resultado obtido pela entrada desejada. Cada camada é composta por unidades, ou neurônios, podendo possuir uma ou mais destas unidades e para que elas entrem em atividade são estabelecidos valores de ativação. O valor de ativação pode ser considerado o estado interno do neurônio, quando é submetido a um valor maior do que possui internamente ocorre a sua ativação, caso contrário não (ZOCCA et al., 2017).

O primeiro exemplo de rede neural foi chamado de *The perceptron* e possui apenas um neurônio na camada de output, tendo o objetivo de realizar classificação binária. Ela foi criada por Frank Rosenblatt em 1957. Um exemplo da primeira rede neural pode ser visto na Figura 2.6, apresentando a camadas de entrada, output e o valore de ativação representado por “w” entre as camadas.

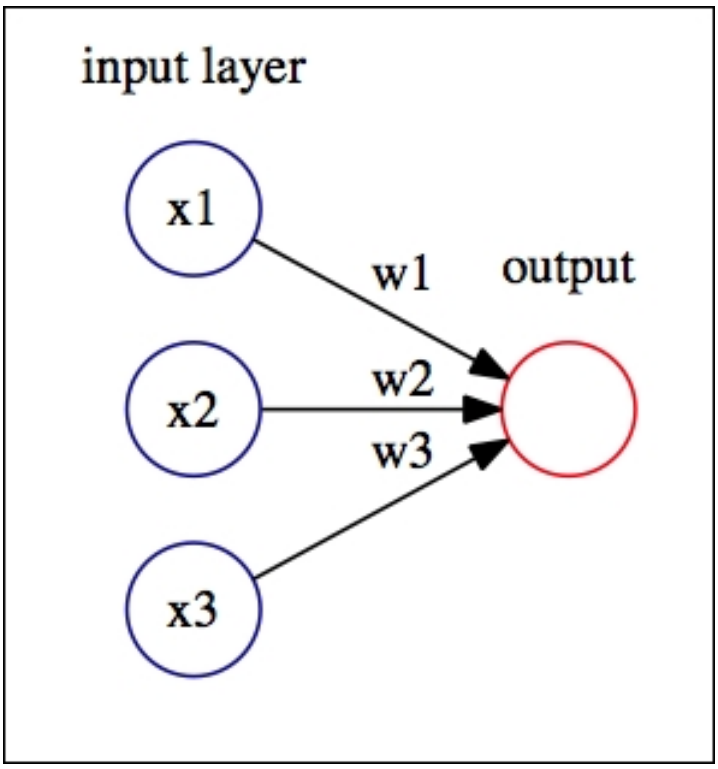


Figura 2.6: Exemplo de rede criada por Frank Rosenblatt (ZOCCA et al., 2017).

As redes neurais podem ser classificadas em categorias baseadas em suas características, sendo que essa classificação se dá de acordo com o número de camadas que elas apresentam e pelo sentido que segue o sinal dentro dela. Ambas as categorias serão descritas no subtópicos a seguir.

2.3.1 Classificação pelo número de camadas

Com relação ao número de camadas, as redes neurais podem ser classificadas em redes monocamada ou multicamadas. A rede monocamada apresenta apenas uma camada entre a entrada e a saída, como mostra a Figura 2.7. Nesta rede, a entrada alimenta os neurônios que irão produzir sinais para a saída (SUGOMORI et al., 2017).

Já nas redes multicamadas, as camadas são apresentadas de forma paralela e os dados das camadas posteriores recebem os dados alterados de sua camada anterior até que alcance a saída. Um exemplo de rede multicamada é mostrado na Figura 2.8.

2.3.2 Classificação Fluxo Interno

Com relação ao fluxo, é possível classificar a rede neural pela sua arquitetura, que pode ser *feedforward* ou *feedback*. Na primeira, o dado é recebido pela entrada e segue para a camada

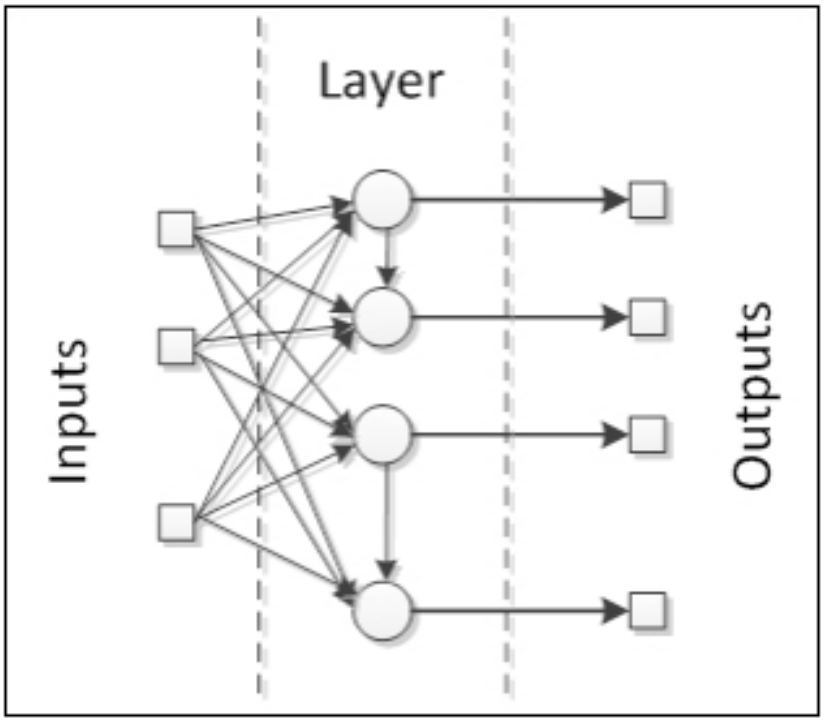


Figura 2.7: Exemplo de rede neural monocamada (SUGOMORI et al., 2017).

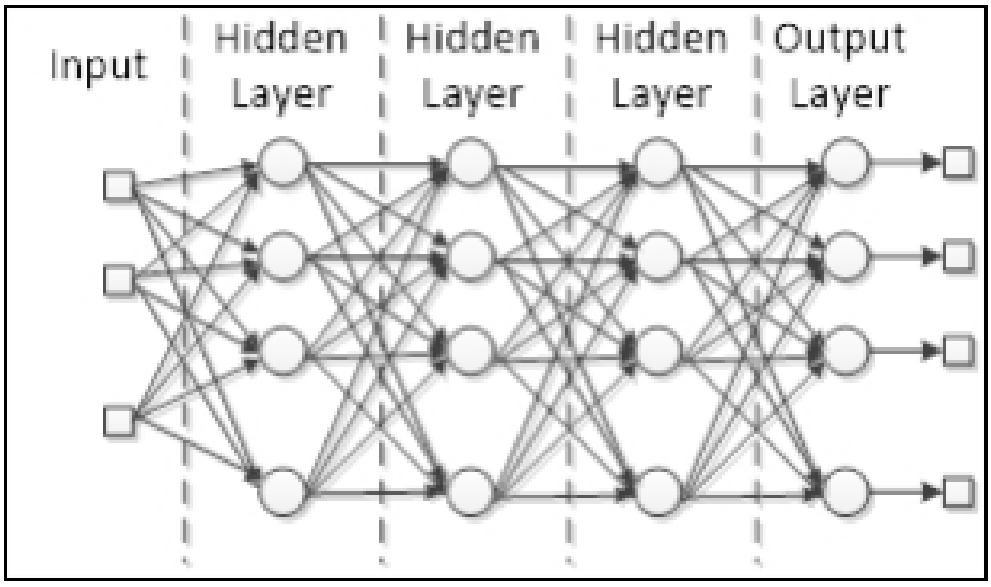


Figura 2.8: Exemplo de rede neural multicamadas (SUGOMORI et al., 2017).

posterior (SUGOMORI et al., 2017), como mostrado na Figura 2.8. Já na segunda, *feedback*, pode ocorrer recorrência internas da informação que já passou pela camada (SUGOMORI et al., 2017), assim, o dado é reprocessado como mostrado na Figura 2.9. O uso do *feedback* promove maior dinâmica na rede, sendo benéfico para o uso do aprendizado reforçado pela repetição e reprocessamento do dado, sendo comumente utilizada em redes monocamadas (SUGOMORI et al., 2017). Porém, redes com essa característica podem apresentar dificuldades no treinamento

devido a possíveis ocorrências de recursão.

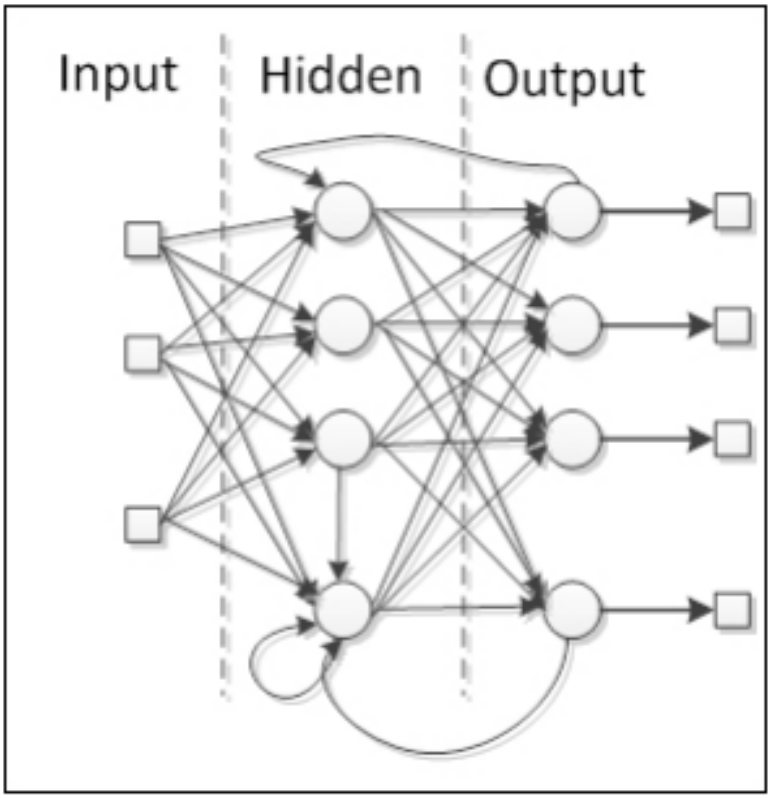


Figura 2.9: Exemplo de rede neural com fluxo feedback (SUGOMORI et al., 2017).

2.4 Deep Learning

O *deep learning*, ou aprendizado profundo, é um dos ramos do aprendizado de máquina e utiliza um conjunto de algoritmos que tentam modelar abstração de alto nível dos dados utilizando grafos com várias camadas de processamento (também chamado de grafos profundos). Neste grafo podem ocorrer transformações lineares e não lineares (GOODFELLOW; BENGIO; COURVILLE, 2016).

Entre algumas aplicações que fazem uso de *deep learning* pode ser citado como exemplo o desenvolvido por (UÇAR; DEMIR; GÜZELIŞ, 2016), em que tem foco específico em identificação de pedestres para veículos autônomos. Outra aplicação é mostrada em (HUI-BIN et al., 2016) que faz a utilização de técnicas de *deep learning* para o reconhecimento através de frames de vídeos para o reconhecimento de objetos em grupos de pessoas. Também, em (EITEL et al., 2015) é feita a utilização de *deep learning* para reconhecimento robusto de imagens com padrão RGB. A arquitetura utilizada é composta por duas redes que posteriormente se unem em uma única, assim, gerando um único resultado com duas entradas de dados.

Um parâmetro muito importante que se destaca para que o treinamento ocorra corretamente com o uso do deep learning é o parâmetro "épocas". Este parâmetro indica a quantidade de vezes que a base de dados destinada ao treinamento irá passar por toda a rede. Assim, para que o aprendizado da rede ocorra corretamente, é importante que a quantidade de épocas seja suficiente para possibilitar a extração de características e ser feita a classificação de forma correta.

2.4.1 Redes Neurais Convolucionais

As redes neurais convolucionais (*Convolutional Neural Network* - CNN) são um tipo de rede neural profunda, sendo chamadas por esse nome pelo tipo de operação linear com matrizes que ocorre nelas que são chamadas de convolução. As CNN são multicamadas e, além das camadas de convolução, também podem apresentar camada com outras operações - como cálculos não lineares, por exemplo (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Com a melhoria do hardware, as redes neurais convolucionais passaram ser mais viáveis e a se tornarem mais populares nas áreas de visão computacional e aprendizado de máquina. Como descrito em (ALBAWI; MOHAMMED; AL-ZAWI, 2017), os elementos básicos das CNNs são a convolução, os passos, o preenchimento e a fórmula da convolução. Considerando uma imagem 32x32 em uma rede em que cada camada tenha uma espessura de 3 unidades, a quantidade de parâmetros gerados seria de 32x32x2 para aquela camada. A convolução permite que a saída desta camada e que será entrada da próxima possua uma conexão menor, assim, reduzindo o número de parâmetros. O passo (*stride*) é justamente a quantidade em que é reduzida a entrada. Por exemplo: em um passo 2 para uma imagem 5x5, a saída seria uma imagem 3x3. A matriz que percorre a imagem é chamada de filtro (ou *kernel*). Os processos serão descritos de maneira mais detalhada futuramente. A Figura 2.10 mostra um exemplo de como ocorre o processo em uma rede que trabalha com esta arquitetura.

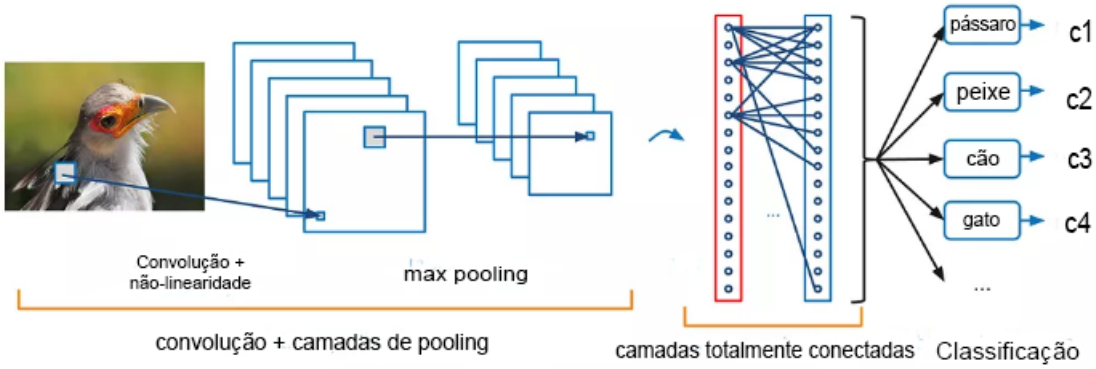


Figura 2.10: Exemplo do processo em uma rede convolucional.

A Figura 2.11 mostra um exemplo de como ocorre o processo em de convolução entre a

imagem e o *kernel*.

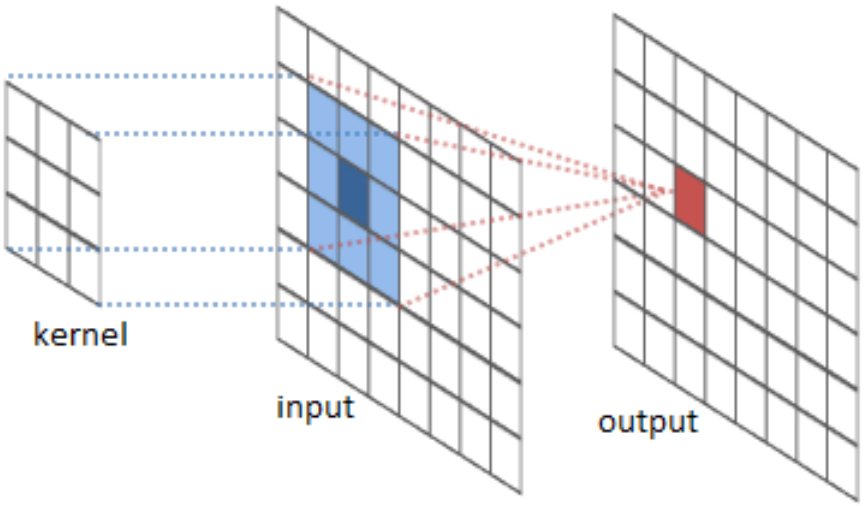


Figura 2.11: Exemplo do processo de convolução de uma imagem.

Durante o processo de convolução pode ocorrer perda de informações que podem existir nas bordas, que não são utilizadas pelo fato de o filtro nunca passar por ela. Para a solução deste problema uma abordagem eficiente é o preenchimento das bordas com zero (*zero-padding*) como pode ser visto na Figura 2.10. Por fim, a fórmula de convolução é aplicada e é feito o cálculo do pixel para a próxima camada. Os exemplos das Redes Convolucionais serão mostrados na próxima seção.

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Figura 2.12: Exemplo zero-padding (ALBAWI; MOHAMMED; AL-ZAWI, 2017).

2.4.2 Funções

Os processos que compõe as CNNs têm como finalidade a acentuação e regulação dos valores após a convolução e ocorrem antes do envio da imagem convolucionada para as camadas

posteriores da rede. Entre os processos, pode-se citar o *Pooling*, *Local Response Normalization* (LRN) e as funções de ativação como ReLU e Softmax.

2.4.2.1 Pooling

A camada de pooling realiza o agrupamento de um conjunto de neurônios a partir do uso de um filtro. Normalmente, as vizinhanças resumidas por unidades adjacentes não costumam se sobrepor (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Desta forma, ocorre a redução da dimensionalidade da imagem categorizando melhor cada sub-região por conectar as características de cada região. Além disso, o processo também realiza a diminuição na quantidade de parâmetros.

A utilização do filtro implica em uma alteração que irá ocorrer na região. Na utilização do *Max-Pooling*, utilizada nas CNNs AlexNet e GoogLeNet, é escolhido como representante da região o ponto que possui o maior valor. Supondo, por exemplo, que uma matriz 4x4 irá ser sobreposta por um filtro 2x2 com passo 2. Após a sobreposição, a matriz resultante sofrerá uma redução de duas unidades nas dimensões x e y, assim, resultando em uma matriz 2x2. A aplicação deste processo no exemplo citado acima é mostrado na Figura 2.13. Outras métricas como a média dos valores dos pixels agrupados também podem ser aplicada a operação de *pooling*.

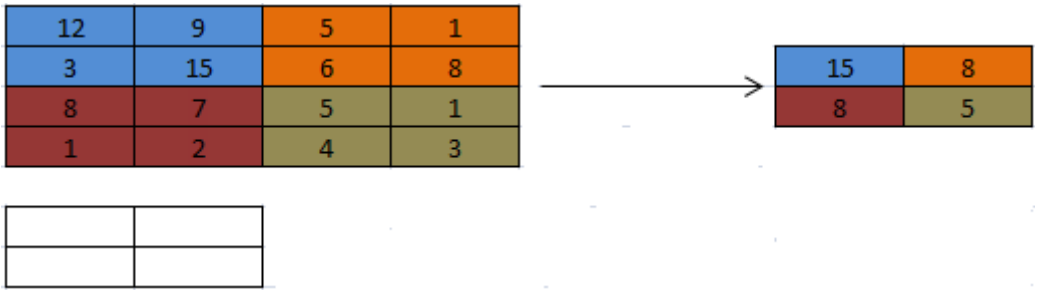


Figura 2.13: Exemplo da aplicação do *max pooling*.

Na Figura 2.14 é mostrado um exemplo da aplicação do pooling após a convolução de uma imagem.

2.4.2.2 Local Response Normalization (LRN)

A LRN é uma forma de normalização dos valores dos neurônios. A normalização dos valores é interessante quando é utilizada a função de ativação ReLU (*Rectified Linear Unit*), que possui valor de ativação ilimitado. Com o uso da LRN é feita uma padronização dos valores e o aprendizado ocorre de forma mais fácil pela melhor distribuição nos valores de ativação.

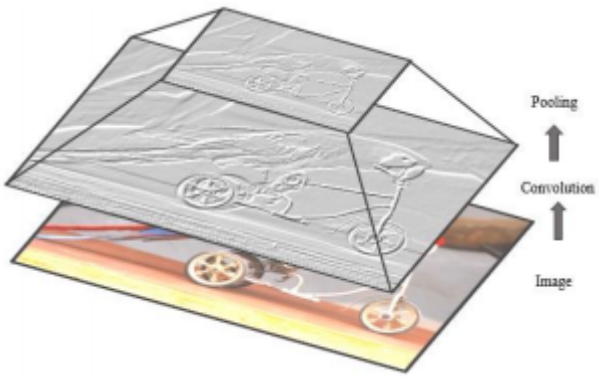


Figura 2.14: Exemplo de convolução seguida da operação de pooling (ALOM et al., 2018).

Essa função de ativação é utilizada na arquitetura da AlexNet nas suas últimas três camadas anteriores à classificação, por exemplo.

A normalização é mostrada na Equação 2.4. Nela é representado o valor de ativação que será computado, sendo i o filtro aplicado e (x,y) a posição. O valor normalizado é dado por b (KRIZHEVSKY; SUTSKEVER; HINTON, 2012).

$$b^i_{x_y} = a^i_{x_y}/(k+\alpha \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)})^\beta \tag{2.4}$$

2.4.2.3 Funções de Ativação

A função de ativação atua no sentido de permitir ou não a passagem da informação por um determinado neurônio. Ela pode ser de vários tipos, como linear ou não-linear, e influenciar no valor transmitido para os neurônios posteriores da rede. Entre as funções de ativação amplamente utilizadas com as CNNs estão a *Rectified Linear Unit* (ReLU) e a *Softmax*.

A função de ativação ReLU é não linear e, quando usada, o treinamento da CNN ocorre mais rápido do que com outras funções(KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Isso se deve aos valores que são obtidos com o seu uso, pois a função não trabalha com valores negativos. Assim, a chegada de um valor negativo acarreta um resultado 0 e a não ativação de um determinado grupo de neurônios. Com parte inativa, os que obtiveram valores para transmitir suas informações podem ser ativados e processados de uma maneira mais fácil (ACADEMY, 2019). A função ReLU é definida pela Equação 2.5.

$$f(x) = max(0,x) \tag{2.5}$$

A camada *Softmax* é uma função de ativação que utiliza uma função sigmoidal e é útil para lidar com problemas de classificação. Ela é responsável pela adequação do valor final entre 0 e 1. Assim, o valor após adequado é dividido pela soma das saídas. Através deste cálculo é dada a probabilidade de a entrada pertencer a uma determinada classe (ZOCCA et al., 2017). A função é definida pela Equação 2.6.

$$\delta(z)_j = e^{z_j} / \sum_{K=1}^{K} e^{z_k} \tag{2.6}$$

Por possibilitar a definição de valores característicos para cada classe, a função Softmax é comunmente usada em camadas de saída das redes. Isso pode ser observado, por exemplo, nas redes AlexNet e GoogLeNet.

2.5 Exemplos de CNN

Nesta seção são apresentadas algumas das redes CNN mais utilizadas, como por exemplo, LeNet, AlexNet e GoogLeNet. Todas as três possuem suas peculiaridades na composição de sua arquitetura, utilizando diferentes recursos em busca de melhor desempenho e acurácia.

2.5.1 LeNet

A LeNet (LECUN et al., 1998) foi proposta na década de 1990, porém devido à limitação computacional da época, como por exemplo a capacidade de memória, sua implementação foi impossibilitada até o ano de 2010. Porém, Lecun (1998) propôs a LeNet com a utilização do algoritmo com a utilização de retro propagação, testando a eficácia em um conjunto de dados de dígitos escrito à mão (ALOM et al., 2018). Na Figura 2.15 é mostrada a arquitetura da rede LeNet (ALOM et al., 2018).

A arquitetura básica da rede convolucional LeNet é composta por duas camadas de convolução, duas de amostragem, duas camadas totalmente conectadas e a camada de saída com conexão gaussiana. A rede que recebe como entrada uma imagem na proporção 32x32 pixels passará por um processo de convolução com 6 camadas de profundidade e será utilizado um filtro 5x5 com passo 1. Desta forma, serão obtidas imagens 28x28 com 6 camadas de profundidade que seguirão para a etapa de *subsampling*, em que ocorre o *pooling*. Esta camada também apresenta profundidade igual a 6 e nela é utilizado um filtro 2x2 com passo 2; assim, as imagens resultantes na camadas serão de proporção 14x14 pixels.

Figura 2.15: Arquitetura da rede LeNet.

Nas duas camadas posteriores o processo se repete, porém a camada de convolução apresenta 16 no valor de sua profundidade e a seguinte semelhante a segunda camada. Na terceira e quarta camada são mantidos os mesmos valores de filtro e passos utilizados. Ao final da quarta camada, o valor da matriz, que inicialmente era de 32x32, passa a ser de 5x5.

As últimas três camadas correspondem a duas de classificação que são totalmente conectadas e a saída que usa como função de ativação a *Softmax*. As responsáveis pela classificação recebem os parâmetros da quarta camada e direciona para o valor para a camada de Output que indica a resposta.

Os processos de funcionamento da LeNet são representados na Figura 2.16

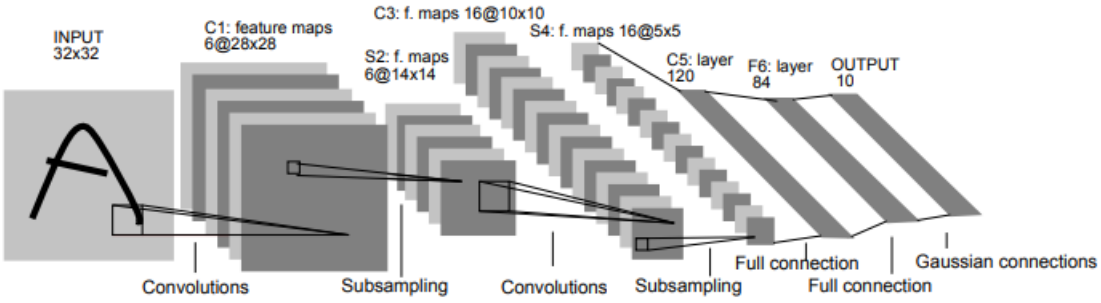


Figura 2.16: Processo interno da rede LeNet (LECUN et al., 1998).

2.5.2 AlexNet

O modelo de CNN AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) foi proposto em 2012, apresentando um modelo de rede diferente quando comparado ao LeNet e foi ganhador do desafio ImageNet para reconhecimento visual de objetos (ILSVRC) neste ano. Foi desen-

volvido por (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) e trouxe um avanço significativo no campo de aprendizado de máquina e visão computacional para classificação e reconhecimento visual, pois atingiu uma precisão de reconhecimento melhor do que as abordagens tradicionais de aprendizado de máquina e visão computacional. Graças aos resultados obtidos com a AlexNet houve um aumento no interesse em aprendizado profundo (ALOM et al., 2018). A arquitetura da rede é mostrada na Figura 2.17.

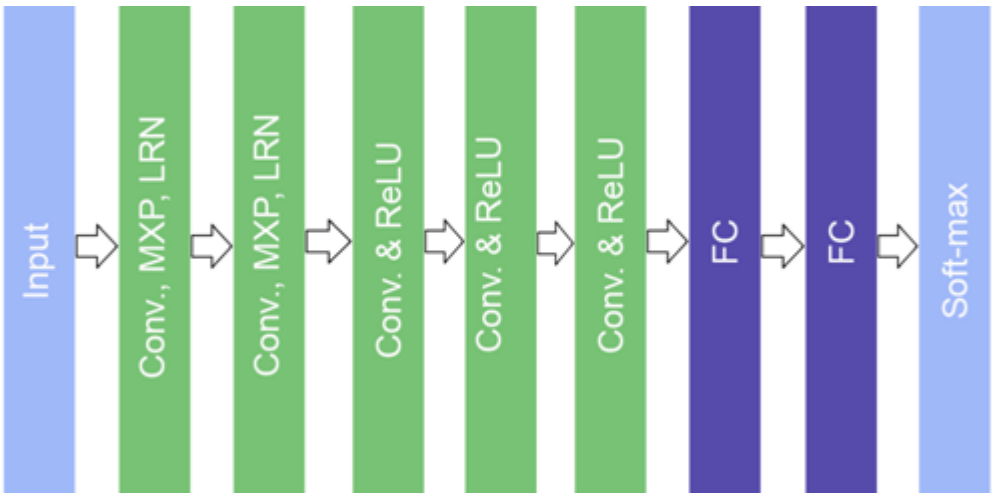


Figura 2.17: Arquitetura da rede AlexNet.

A rede AlexNet é composta por uma arquitetura de 3 camadas de convolução e 2 camadas totalmente conectadas. Antes da primeira camada de convolução são aplicadas duas camadas de normalização dos valores com o objetivo de diminuir os valores e estimular regiões de neurônios, passando por filtros de tamanhos 11x11 e 5x5 respectivamente. Em ambas as camadas são utilizadas as funções de *Max-Pooling* e LRN, assim, ocorre a redução do número de parâmetros pela ação do pooling e a normalização dos valores. A normalização é importante para a função de ativação que é utilizada nas camadas posteriores.

Nas camadas três, quatro e cinco da arquitetura, são utilizados filtros 3x3 com os neurônios utilizando como função de ativação a ReLU. Após a passagem por estas cinco camadas, são finalizadas as extrações de características e os dados são classificados por duas camadas totalmente conectadas. Com a entrada classificada, ocorre a utilização da camada com *softmax* para calcular qual a saída correta para a entrada (ALOM et al., 2018).

2.5.3 GoogLeNet

A CNN GoogLeNet (SZEGEDY et al., 2015) foi proposta por Christin Szegedy, do Google, em 2014, e foi a rede vencedora da ILSVRC. A proposta foi feita com o objetivo de reduzir a complexidade computacional em comparação às demais CNN's tradicionais. O diferencial da

proposta da GoogLeNet foi a incorporação de camadas de inception, que são camadas compostas por diferentes tamanhos de *kernel* e que têm por objetivo criar operações que capturam padrões de correlação esparsos que criam uma pilha de recursos, e possuem um número variável de campos (ALOM et al., 2018). A Figura 2.18 apresenta a arquitetura da rede GoogLeNet.

Na Figura 2.18 são indicadas as camadas com a representação em azul a operação de convolução, as vermelhas o pooling, amerelo softmax e em verde a normalização e concatenação dos valores para serem enviados para a camada posterior. As camadas desta rede são chamadas de *Inception* e a atuação em conjunto das convoluções e pooling nas camadas permitem colocar cada camada como um módulo na rede.

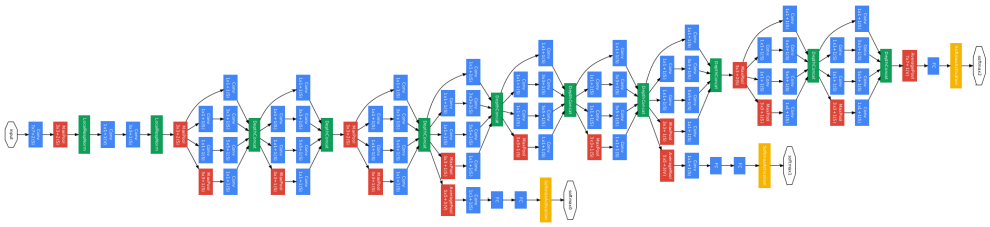


Figura 2.18: Arquitetura da GoogLeNet (SZEGEDY et al., 2015).

A GoogLeNet possui ao todo 22 camadas, assim tendo mais camadas que qualquer outra rede anteriormente proposta, entretanto apresenta menor número de parâmetros do que a sua rede antecessora, a AlexNet.

A GoogLeNet é inspirada na LeNet (LECUN et al., 1998) e, antes da aplicação dos módulos de *Inception*, apresenta camadas semelhantes de convolução, pooling e normalização como o padrão que a inspirou. Então, após a passagem por etapas mais convencionais são aplicados os módulos Inceptions, que são compostos por uma série de convoluções e operações de max-pooling. Os filtros são de tamanhos 1x1, 3x3 e 5x5, sendo o pooling realizado com passo 2 (SZEGEDY et al., 2015).

Desta forma, antes da entrada em uma nova camada, o processo interno consiste na passagem da entrada por quatro convoluções: 1x1, uma 3x3, uma 5x5 e um pooling. O final de um Inception ocorre com a aplicação de um filtro 1x1 e normalização dos valores (ALOM et al., 2018). Para a classificação, também é utilizada uma rede totalmente conectada com uma camada *Softmax*.

Um exemplo de camada *inception* é mostrado na Figura 2.19.

Entre os benefícios da arquitetura da GoogLeNet está em possibilitar o uso de várias camadas com grande quantidade de componentes em cada, sem o aumento do custo computacional

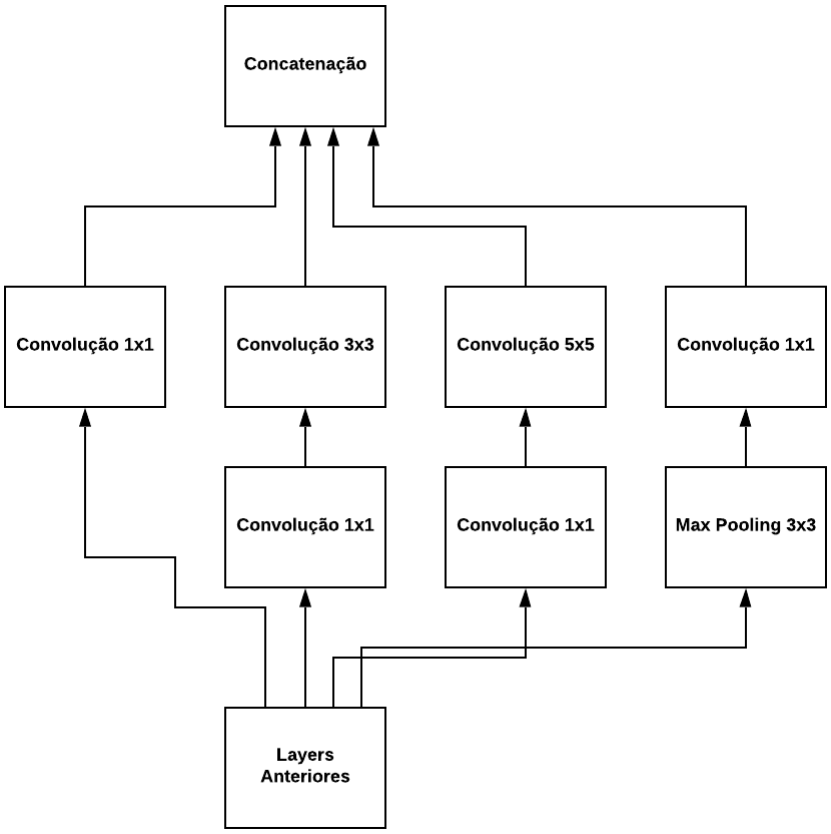


Figura 2.19: Arquitetura de uma camada *Inception*.

exagerado (SZEGEDY et al., 2015). Desta maneira a maior quantidade de convoluções e a redução de parâmetros gerados com a utilização do pooling em cada módulo e os dos filtros de concatenação permitem que seu desempenho seja melhor que a da rede convolucional AlexNet.

2.6 Frameworks

Existem vários frameworks para a implementação de redes neurais, convolucionais e aplicação para *deep learning*. Cada um deles contempla uma ou mais linguagens e apresentam formas distintas de serem programados. Entre os existentes está o Caffe, Tensorflow, Theano e Torch. Os frameworks auxiliam a implementação das redes pois possui funções básicas prontas que facilitam o desenvolvimento de novos projetos. Também apresentam a implementação de redes já existentes com o objetivo de permitir a sua utilização e configuração.

2.6.1 Tensorflow

É uma biblioteca de código aberto para computação numérica desenvolvida em 2011 pelo Google Brain Research Group para estudos de aprendizado de máquina e redes neurais profun-

das (FONNEGRA; BLAIR; DÍAZ, 2017). Sua vantagem está em sua arquitetura flexível que pode ser implementada em uma ou mais CPUs ou GPUs em desktops, servidores e sistemas mobile através de uma única API. Além disso, o Tensorflow oferece suporte para o uso nos serviços de nuvem da Google (FONNEGRA; BLAIR; DÍAZ, 2017). O Tensorflow trabalha com a construção de grafos para a computação dos dados. Nestes grafos, os nós representam as operações matemáticas, enquanto que as bordas (tensor) representam os vetores multidimensionais que realizam a comunicação.

2.6.2 Theano

O Theano foi desenvolvido pelo grupo LISA da Universidade de Montreal, Canadá. Este framework não é específico para *deep learning*, mas sim uma biblioteca em python para a computação de expressões matemáticas complexas e pode ser executada tanto em CPU como em GPU, com o uso de CUDA. Mesmo não sendo específica para *deep learning*, possui módulos para facilitar o desenvolvimento de diferentes redes. Também pode utilizar vetores multidimensionais de forma a aumentar o desempenho do hardware na construção de grafos para que os dados sejam mais bem explorados (FONNEGRA; BLAIR; DÍAZ, 2017).

2.6.3 Torch

O Torch é uma biblioteca de código aberto para aprendizado de máquina, funcionando tanto com CPU como com GPU. Em questão da linguagem, a biblioteca é desenvolvida em uma linguagem baseado em LUA e funciona com a linguagem C e CUDA (FONNEGRA; BLAIR; DÍAZ, 2017). Este framework é utilizado para aplicações cientificas, comerciais e para projetos industriais sendo utilizados por empresas como IBM e Facebook.

2.7 Caffe

O Caffe é uma biblioteca de código aberto e é dedicada a *deep learning*. A biblioteca é escrita em C++, mas também apresenta implementações em python e matlab apresentando suporte tanto para CPU como para GPU. Apresenta várias vantagens como teste de precisão, rigor experimental e uma instalação rápida (CENGIL; ÇINAR; ÖZBAY, 2017). Outro ponto de destaque desse framework é a configuração da rede que pode ser feita sem a necessidade de código, sendo feita no arquivo de configuração que também permite a otimização do módulo. Devido a suas vantagens, é muito útil na classificação de imagens que requerem o processamento de milhões

de imagens.

2.8 Considerações Finais

Neste capítulo, foram abordados os principais conceitos fundamentais relacionados ao desenvolvimento do trabalho, apresentando não só os conceitos das teorias, mas também as formas de implementar a tecnologia mostrando os frameworks mais utilizados.

Capítulo 3

TRABALHOS CORRELATOS

Este capítulo apresenta algumas abordagens encontradas na literatura que utilizam tecnologias e técnicas que se assemelham com as utilizadas neste trabalho.

3.1 Considerações Iniciais

Na literatura foram encontradas diversas abordagens para o reconhecimento de imagem utilizando *deep learning* para as mais diversas categorias de objetos. O tema mostrou-se muito relevante pela sua abrangência e pelas suas utilidades nos ambientes mais diversos. A seguir são apresentadas abordagens relacionadas com este trabalho.

3.2 Abordagens

Em (YANG; YUAN; TIAN, 2011) é desenvolvido um método com misturas de técnicas para auxiliar deficientes visuais na escolha de suas vestimentas, visando tornar essa atividade mais fácil e independente, classificando as roupas em categorias como listras, treliças, especiais e sem padrão. A partir da obtenção da imagem com o uso de câmeras, por exemplo, são extraídas características das texturas com o uso dos métodos SIFT e STA juntos. Então é feita a classificação da imagem segundo o padrão extraído, transmitindo o resultado via áudio para o usuário. Entre os resultados obtidos, o uso de ambos os métodos mostrou como um pode ser complementar ao outro, obtendo taxas mais altas de reconhecimento com os métodos SIFT e STA combinados, apresentando um valor de 87,50% com o uso de 30% da base de treinamento em comparação a 78,63% e 80,70%, com os métodos STA e SIFT, respectivamente, utilizados de forma individual.

Também no intuído de prover maior percepção do ambiente a deficiente visuais (FILIPE et al., 2016) utiliza o Microsoft Kinect e marcadores para facilitar a locomoção em ambientes internos. Para que o Kinect identifique os marcadores, esses são colocados a cerca de cinquenta centímetros do chão em pontos estratégicos no ambiente, como em escadas, pilares ou outros locais de interesse. O trabalho utiliza uma rede neural com três camadas treinada com *backpropagation*, classificando as imagens obtidas em sem obstáculos, com obstáculos, subindo escada ou descendo. Os resultados obtidos atingiram aproximadamente 99% de reconhecimento, obtendo 100% de reconhecimento na categoria obstáculo devido ao auxílio das marcações que guiam a classificação.

Em (YANG et al., 2010) foi desenvolvido um sistema para o auxílio de deficiente visuais em ambientes internos não familiares. O sistema realiza a detecção de portas com informações textuais a partir de câmeras. São classificados portas de escritórios, laboratórios, banheiros, saídas e elevadores, por exemplo. Para a detecção da porta é utilizado um algoritmo desenvolvido para este fim e em conjunto é utilizado o algoritmo OCR para a identificação de caracteres nestas portas. Como resultado foi obtido foi obtida uma precisão média de 89,5% para imagens com fundo mais complexos e 92% para as mais simples. Após a detecção e identificação da porta e do conteúdo textual o resultado é retornado ao usuário através de áudio.

Com a mesma motivação de (YANG et al., 2010), (KUNENE; VADAPALLI; CRONJE, 2016) também propõe um sistema de auxílio para a locomoção de deficientes visuais em ambientes internos não familiares através da detecção de sinais que estejam com um plano de fundo padrão. Estes sinais podem ser as mais variadas placas para simbolizar, por exemplo, locais de banheiro, saídas e outras sinalizações. Com a imagem capturada é localizado o ponto de interesse e aplicado o método SURF para então ser classificado pelas cores e desenhos identificados na imagem. A resposta é enviada por meio de áudio ao usuário.

Em (CONTRERAS; ROSA, 2016) é buscado o reconhecimento de estruturas físicas em ambientes internos, como escadas, elevadores e corredores, para a navegação segura de robôs a partir o uso de aprendizado de máquina utilizando redes neurais profundas para o reconhecimento de padrões. Foi criado um sistema de rede genérica que pode ser customizada de acordo com o dado de entrada e integrada a robôs móveis. Como resultado foi desenvolvido uma rede neural genérica que com um treinamento de 1343 iterações e taxa de aprendizagem de 0,001 foi alcançada uma precisão de 94%, porém na identificação da imagem em tempo real o acerto foi baixo. Após um novo treinamento com a mesma taxa de aprendizagem e 2999 iterações foi alcançada uma taxa de reconhecimento de 80%.

No intuito de evitar que o deficiente visual encontre obstáculos no caminho em ambientes

internos (BAI et al., 2017) propõe a criação de um aparato multissensor e um algoritmo que une a utilização das imagens em profundidade com a emissão de ondas. Utilizado como se fosse um óculos, o aparelho multissensor é composto por um sensor ultrassônico e um de profundidade com o objetivo principal de resolver o problema da detecção de pequenos obstáculos. A imagem de profundidade é obtida com uma câmera que possui um campo de visão de até 2.692 metros com uma angulação de 30 graus. O senso ultrassônico envia sinais para a região que está sendo coberta pela câmera de profundidade e caso encontre algum desnível tem retorno sonoro ao usuário e indicando o caminho livre. Nos testes foram abordados os cenários de um supermercado, escritório e em casa. Foram obtidos bons resultados, reconhecendo inclusive obstáculos menores e transparentes.

Também no âmbito de auxílio a deficientes visuais (JABEEN; MUHAMMAD; ENRÍQUEZ, 2015) propõe um sistema que auxilie as pessoas com esse impedimento no momento da realização de compras de vestimentas e outras indumentárias utilizando técnicas de processamento de fala e imagem em conjunto. O sistema consiste na adaptação do espaço da loja, sendo utilizado o dispositivo de capturar a imagem para obter informações do usuário. A partir destas imagens são extraídos dados como o perfil físico, tom de pele e outras características, além disso, o sistema também obtém dados através de informações verbais. As informações obtidas servem como treinamento para uma rede neural composta por três camadas, entrada, camada oculta e saída. A seleção do tipo de objeto ocorre por meio de uma combinação do que foi extraído das imagens com a categoria pedida pelo usuário. O projeto visa trazer mais independência ao deficiente visual.

O trabalho proposto em (MONTEIRO et al., 2017) faz utilização de redes neurais convolucionais (CNN) e SVM para classificar o movimento do cão guia e descrevendo as ações do animal, assim, auxiliando na orientação do deficientes visuais. No trabalho são utilizadas redes neurais convolucionais, AlexNet e GoogLeNet em conjunto, para a extração de características e posterior classificação com a utilização de SVM. A obtenção dos dados ocorre com o uso de uma câmera acoplada próxima a coleira do cão e retorna as ações de forma auditiva ao usuário. A partir da posição da cabeça do cão é possível identificar a direção e situações como a espera por estar passando um carro, se ele está se alimentando, tomando água ou cheirando algo. A combinação das CNN mostrou-se vantajosa para o projeto, resultando em uma acurácia de 74%.

Em (BASHIRI et al., 2018) é desenvolvido um estudo para a criação de um sistema que auxilie pessoas com deficiências visuais na navegação em ambientes desconhecidos, mais especificamente em ambientes de tratamento de saúde como hospitais, clínicas e unidades de emergência. O sistema busca realizar a classificação de portas, escadas e sinais característicos de ambientes

hospitalares. No trabalho foi utilizada a CNN AlexNet para a extração de características da base de dados e algoritmos como SVM e KNN como classificadores. Para o projeto foi planejado a captura de imagens do ambiente com a utilização do dispositivo Google Glass que envia a imagem via wifi local para ser classificada e retorna a classificação ao usuário de forma sonora. O estudo mostrou que a junção das tecnologias pode proporcionar uma boa precisão e acurácia na identificação, maior do que 98% por exemplo. Também foi concluído que a utilização da AlexNet para extração de características se mostrou suficiente para alcançar bons resultados.

Com uma abordagem para ambientes externos, o trabalho proposto por (CHUANG et al., 2018) visa o desenvolvimento de um robô que cumpre o papel de cão guia com a movimentação baseada em linhas. A proposta é que o robô seja autônomo e consiga identificar o caminho, evitando possíveis obstáculos nas mais variadas texturas do piso, condições de iluminação e na variação da linha na mudança de um ambiente para o outro. Para a identificação do caminho é utilizada uma rede convolucional chamada TrailNet e seu treinamento ocorreu tanto em ambiente virtual e real. No virtual sendo utilizadas simulações em computador e no ambiente real o treinamento da movimentação do robô de forma independente por um percurso pré-definido. O hardware é composto por uma câmera para a captura da imagem, para o processamento é utilizado uma RespberriPy e alimentada por uma bateria. Como teste foram chamados dez participantes para utilizar o aparelho e responder um questionário com perguntas referentes a usabilidade, custo, portabilidade e outros pontos. a resposta dos usuários foi positiva, mostrando também que a CNN apresentou um bom desempenho para as possíveis variações.

Já (SHIM; YUAN; TAN, 2017) tem um objetivo semelhantes a (CONTRERAS; ROSA, 2016). Porém, faz uso do método SURF e de redes neurais profundas com o diferencial de que o treinamento de cada camada ocorre uma por vez, para prover a robôs, como robôs sociais, de serviço e de resgate, capacidades como a identificação de objetos, obstáculos, planejamentos de rotas e navegação em ambientes interiores utilizando para a entrada das imagens uma câmera RGB-D. Como resultado foi obtido um bom desempenho na extração de características do ambiente pelo método SURF, assim como uma boa estimativa da posição do robô. Também mostrou-se satisfatório o uso da rede neural para fazer o desvio dos obstáculos identificados a partir do único dispositivo de entrada de dados do ambiente.

Em (DING et al., 2017) busca-se a detecção de objetos para a robótica em ambientes interiores, porém utilizando redes neurais convolucionais pré treinadas de forma off-line com base de dados de ambientes internos públicos e frames de vídeos de ambientes privados. Foram obtidos resultados eficientes para a detecção de objetos com a mistura de bases quando comparada com o modelo de treinamento que utiliza somente a base de ambientes internos. No

trabalho de (DENG; ZHU; REN, 2017) é buscada a detecção multiclasse de objetos em ambientes internos com imagens panorâmicas utilizando o método SURF, o algoritmo *K-Nearest Neighbors* (K-NN) para a medição da distância euclidiana entre o vetor da imagem e os pontos mais próximos, RANSAC para o alinhamento correto e redes neurais convolucionais com base de imagens criada a partir de três câmeras olhos de peixe para criar imagens panorâmicas de qualidade. Como resultado foi obtido uma precisão de 68,7%, porém, que pode aumentar com o incremento da base de treinamento.

O trabalho desenvolvido por (VELLA et al., 2016) tem como objetivo pesquisar um sistema que auxilie pessoas idosas em ambientes domésticos realizando o monitoramento e classificação das atividades realizadas. Para a classificação são utilizadas redes neurais profundas, sendo realizada uma comparação entre duas CNN's e como base para a comparação dos resultados uma rede MPL (*Mult Perceptron Layer*) para a análise das saídas. Ambas as CNN's apresentaram melhor desempenho do que a MLP. Entre as CNN's a rede que apresenta uma camada a mais entre as camadas convolucionais apresentou maior estabilidade e resultados melhores.

Em (ZHANG et al., 2017c) é realizado um estudo sobre a criação de bases de ambientes internos utilizando redes neurais convolucionais e uma base com imagens 3D reais para gerar imagens sintéticas, essas que podem ser utilizadas para o treinamento de aplicações, como recursos para sistemas de auxilio e monitoramento, estimativa de superfície, segmentação de imagem e detecção de objetos. Para a criação da base sintética foi utilizada a base SUNCG que é composta por cerca de 45 mil cenas, 2644 objetos e 84 categorias oferecendo também informações como texturas, superfície e iluminação das imagens. A base sintética é criada a partir de diferentes posicionamentos das imagens reais, sendo composta por 500 mil imagens em 2D a partir do processamento da variação do ambiente real.

3.3 Considerações Finais

Neste capítulo foram apresentados outros trabalhos que exploram o desenvolvimento de base de dados ou tecnologias para a área da robótica e auxílio de deficientes visuais. Estes trabalhos utilizam diferentes técnicas para alcançar seus objetivos, entre essas técnicas estão as CNNs para obtenção de característica e treinamento de redes.

Capítulo 4

MÓDULO DE RECONHECIMENTO DE IMAGEM

Este capítulo apresenta o protótipo em que foi implementado o módulo de reconhecimento de imagens e estados. Assim, serão detalhados os aspectos de hardware e a arquitetura do protótipo. Também será explicado o novo módulo implementado e como ele se encaixa na arquitetura do sistema do protótipo.

4.1 Descrição do Protótipo

A deficiência visual causa limitações na vida das pessoas e um destes fatores limitantes é a locomoção em ambientes desconhecidos. Com a finalidade de desenvolver uma forma de auxiliar no deslocamento de deficientes visuais em ambientes internos desconhecidos e aumentar a percepção destas pessoas do ambiente que as cerca foi desenvolvido o projeto de (DOURADO; PEDRINO, 2018).

O projeto resultou na criação de um protótipo, e do sistema, para auxiliar deficientes visuais verificando se o caminho está livre ou não. O protótipo é utilizado como um colete e nele é anexado as peças de hardware necessárias para seu funcionamento. Nas subseções 4.1.1 e 4.1.2 são descritos com mais detalhes o hardware utilizado e a arquitetura do sistema que foi desenvolvido para ser embarcado no protótipo.

4.1.1 Descrição do Hardware

O protótipo utiliza como câmera o dispositivo Kinect do Xbox 360 que consegue obter imagens RGB e também com profundidade. Para o realização do processamento é utilizado uma placa Jetson TX1 da NVIDIA (NVIDIA, 2017), mostrada na Figura 4.1. O usuário é informado se o caminho está livre ou não através de um fone de ouvido de condução óssea. Para alimentar

o sistema são utilizadas duas baterias, a primeira é uma LiPo 3S de 2200 MAh que fornece energia para o dispositivo Kinect, já a segunda é uma LiPo 4S de 3000 MAh que é utilizada como fonte de energia para Jetson TX1.

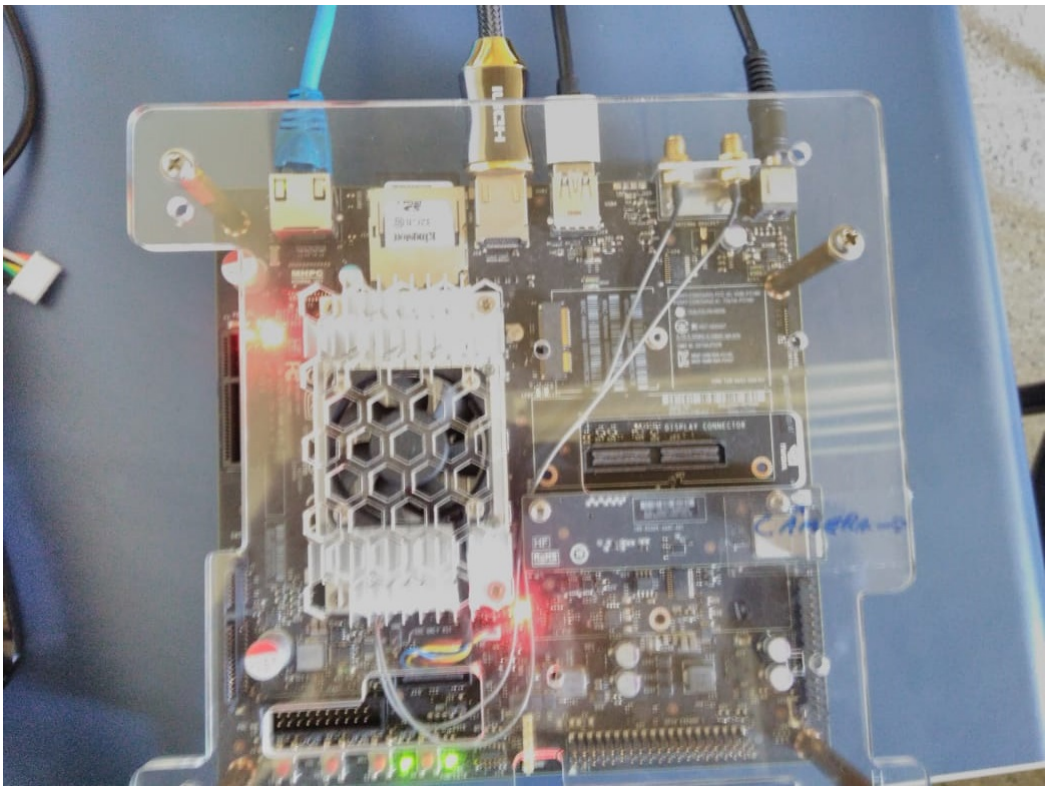


Figura 4.1: Placa NVIDIA Jetson TX1.

Na Figura 4.2 é mostrado o protótipo do colete sendo utilizado e pode ser observado a estrutura do equipamento de uma visão frontal e lateral. Na parte frontal da vestimenta é colocado o dispositivo Kinect que é anexado ao protótipo com a utilização de um tecido, enquanto que na parte traseira são colocadas as baterias e a placa TX1. O material do colete consiste em uma veste tática (de airsoft) adaptada para comportar as peças de hardware.

4.1.2 Descrição da Arquitetura do Sistema

O sistema que foi feito para a utilização no protótipo foi desenvolvido em C++ e é dividido em módulos que realizam desde a aquisição da imagem, passando pelo tratamento, processamento da imagem e *feedback* ao usuário. Os módulos atuam separadamente, mas gerando dados para o módulo posterior com a finalidade de processar a imagem para verificar se o trajeto está livre ou não.

A arquitetura do sistema, antes da inclusão do módulo de classificação de objetos e estados, é mostrado na Figura 4.3.

Figura 4.2: Visão frontal e lateral do protótipo sendo utilizado.

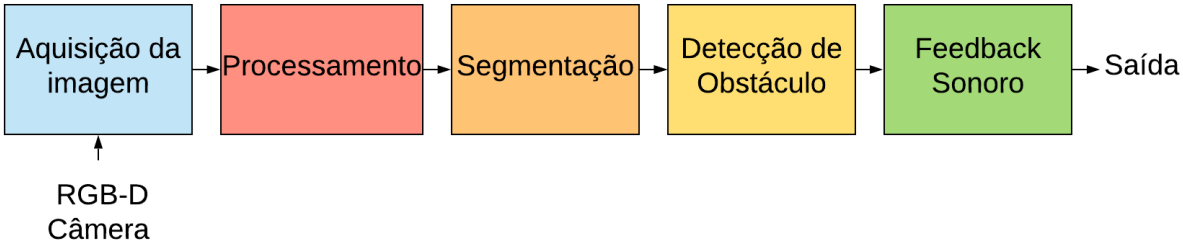


Figura 4.3: Arquitetura do sistema (DOURADO; PEDRINO, 2018).

As funções dos módulos são:

- Aquisição da imagem: Utilizando o dispositivo Kinect é obtida a imagem nos formatos RGB e com profundidade.
- Processamento: Recebem as imagens RGB e com profundidade e é responsável por fragmenta-la para a utilização nos módulos posteriores.
- Segmentação: Utiliza a parte da imagem fragmentada com profundidade para identificação do caminho livre.
- Detecção de Obstáculos: A detecção ocorre utilizando a outra parte da imagem que verifica se há obstáculos.
- Feedback Sonoro: Responsável por retornar o caminho que não apresenta impedimento para o usuário.

O processo tem início com a aquisição da imagem. Ela ocorre com uso do dispositivo kinect

e obtém a imagem no formato RGB e com profundidade na proporção de 640 x 480. Após a obtenção da imagem entra em ação o módulo de processamento em que ocorre a divisão de cada imagem em duas partes. Os 75% superior de cada imagem é utilizado na detecção e os 25% inferior na segmentação. Exemplos das imagens obtidas com o protótipo são mostrados nas Figuras 4.4 e 4.5.

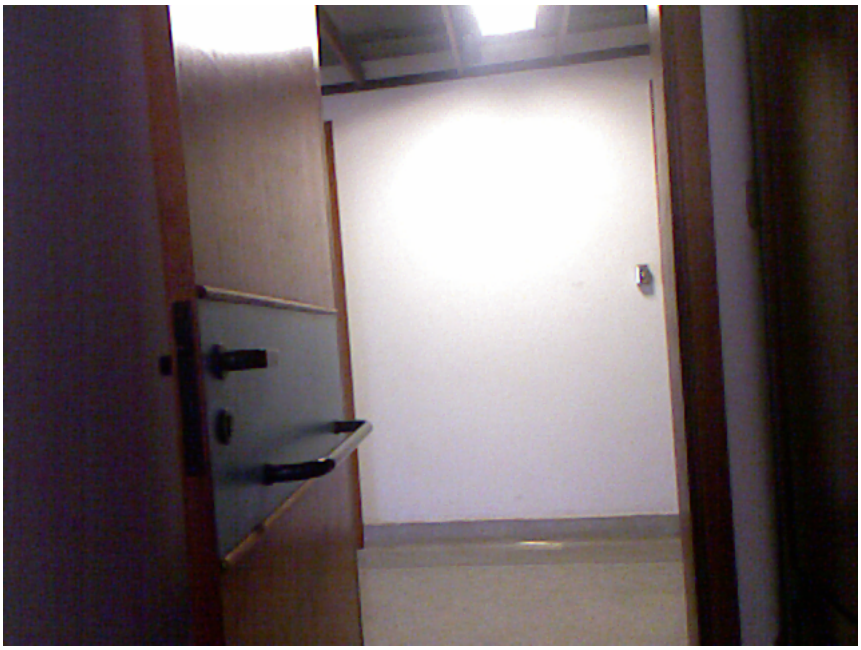


Figura 4.4: Exemplo de imagem RGB-D obtida com uso do protótipo.

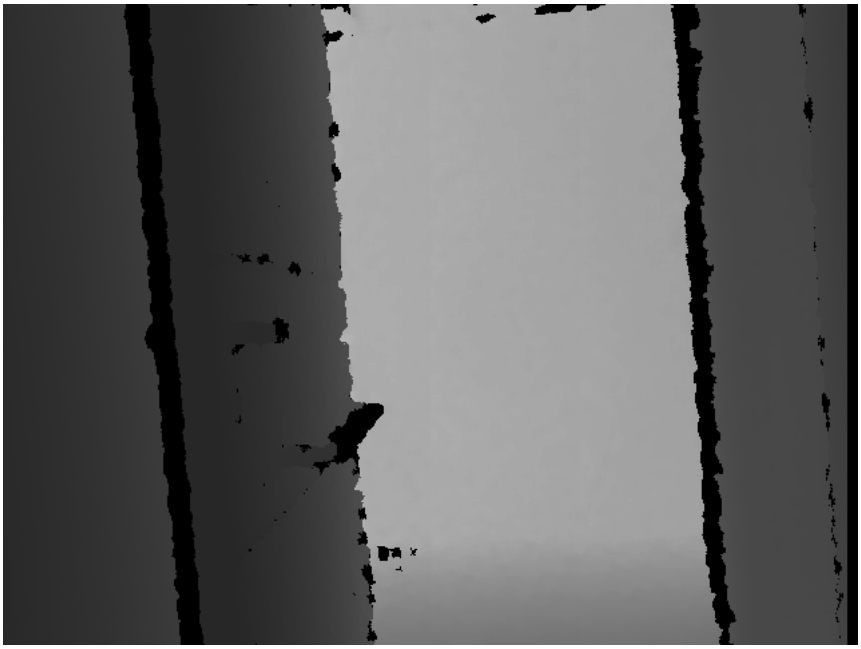


Figura 4.5: Exemplo de imagem com profundidade obtida com uso do protótipo.

Com a imagem fracionada, o módulo de segmentação recebe e utiliza os 25% inferior da imagem com profundidade para verificar se há caminho livre, enquanto que os outros 75%

servem para averiguar se existe algum obstáculo nesta parte do trajeto. Com a passagem por estes dois módulos o caminho livre de impedimentos é indicado ao usuário, ou avisa ao mesmo que existe obstáculo pelo trajeto. O resultado é retornado por meio de *feedback* que tem como saída o fone de ouvido de condução óssea.

O sistema apresenta a característica de ser adaptável ao usuário, apresentando a possibilidade de configuração de *feedback* para a forma que mais se adequada ao perfil de quem está usando o protótipo. Sendo possível alterar a forma com que o sistema retorna o caminho livre.

4.2 Implementação do Módulo

O módulo implementado teve como objetivo a classificação de imagens e a identificação de seus estados, como aberto e fechado. A inserção do novo módulo na arquitetura já existente ocorre com o intuito de complementar a detecção de obstáculos, assim, possibilitando que passagens impedidas em ambientes internos possam ser identificadas e propiciar maior reconhecimento do local que cerca o deficiente visual.

Como linguagem de programação foi utilizada C++ que é a mesma empregada para a construção do sistema do protótipo. Como a proposta de sua funcionalidade é realizar a classificação a partir do requerimento do usuário, a estrutura do novo módulo pode ser dividida em três partes que são recebimento de dados, classificação e resposta.

As funções de cada parte são:

- Recebimento de dados e obtenção de imagem - A partir da identificação da existência de um obstáculo, o usuário pode requerer a classificação via comando de voz, por exemplo. Esta primeira parte faz a requisição da imagem RGB já capturada pelo sistema cuja imagem com profundidade identificou o obstáculo.

- Classificação - Nesta etapa são utilizados os arquivos contendo a rede que realiza a classificação. Estes arquivos são a identificação da rede e rótulos das classes treinadas. Ao final é dado o resultado.

- Resposta - Com a classificação definida é enviado o rótulo da imagem para o módulo de *Feedback* Sonoro.

Inicialmente foi implementado um novo comando de voz na lista de palavras aceitas pelo sistema. A interpretação da voz é feita com uso da biblioteca pocketsphinx e o comando é

convertido para texto dentro do sistema pela biblioteca sphinx. Estas bibliotecas cumprem a função de converter o áudio para string, assim, possibilitando o seu uso no sistema como um comando já reconhecido. O reconhecimento de comandos é processado pelo sistema uma vez que na arquitetura anterior a implementação do módulo o recebimento de comando de voz já existia.

Para estabelecer a comunicação deste novo módulo com os já existentes no sistema foram criadas novas funções que realizam o recebimento do sinal que mostra que houve a detecção de obstáculo, o requerimento do usuário, a passagem da imagem para a CNN e o envio da resposta da classificação para o módulo de *Feedback*. Para a comunicação das funções criadas com a CNN não foi necessária nenhuma conversão de código uma vez que foi utilizada a versão C++ do Caffe no protótipo. Inclusive o Caffe já possui função de adequar a imagem para o tamanho da entrada apropriada da rede utilizada, no caso 256 x 256 pixels para a AlexNet e 224 x 224 pixels para a GoogLeNet.

Também houve o reaproveitamento de função já existente no sistema pela aproximação de funcionalidades. Tal uso ocorreu com a função responsável por buscar a imagem para a classificação, sendo alterado o parâmetro da imagem que deveria ser buscada, pois é utilizada a imagem RGB e não a com profundidade.

O processo pelo qual a imagem é submetida durante o início da classificação até a obtenção do resultado depende da CNN que está sendo utilizada. No caso, foram utilizadas as CNNs AlexNet e GoogLenet e a forma como o processamento da imagem ocorre em ambas as redes foi descrito em detalhes no Capítulo 2.

Na Figura 4.6 é mostrada a arquitetura do protótipo com a adição do módulo de reconhecimento de imagem e estado.

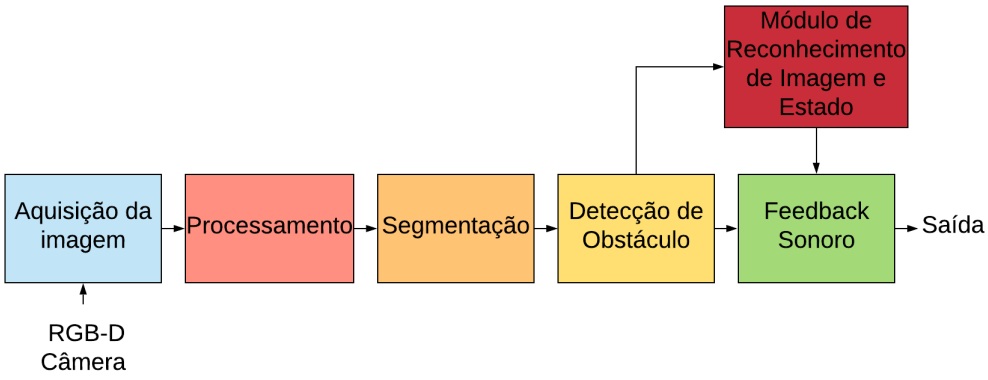


Figura 4.6: Arquitetura do sistema com o módulo implementado.

Na Figura 4.7 e Figura 4.8 são mostradas, respectivamente, a arquitetura do módulo de forma geral e de suas subdivisões.

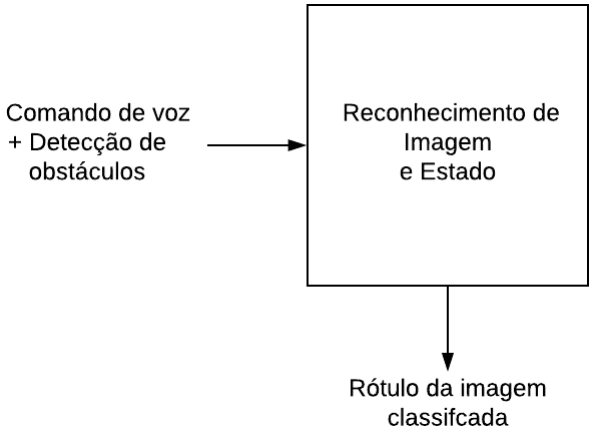


Figura 4.7: Módulo de reconhecimento de imagem e estado.

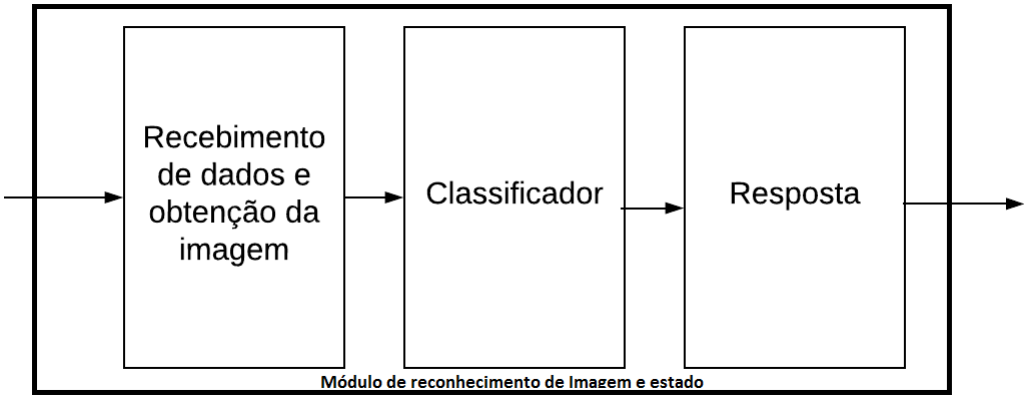


Figura 4.8: Arquitetura interna do módulo de reconhecimento de imagens.

Os códigos do novo módulo foram adicionados ao sistema e realizado a passagem de parâmetros descritas no funcionamento das partes que compõe a classificação de imagem e estados. Por auxiliar a detecção de obstáculos, a classificação se localiza logicamente na arquitetura do sistema embutido após a Detecção de Obstáculos e em um mesmo nível próximo ao *Feedback* Sonoro.

4.3 Considerações Finais

Neste capítulo foi apresentada a descrição do protótipo, sendo mostrados os módulos que compõe sua arquitetura antes e depois da adição do reconhecimento de imagem e estado. Também foram detalhadas as funcionalidades e partes desenvolvidas neste trabalho, sendo abordado como o novo módulo é divido e qual a função de cada parte.

Capítulo 5

RESULTADOS EXPERIMENTAIS

Este capítulo apresenta os testes realizados e os resultados alcançados com eles. Assim, abrangendo desde os treinamentos das redes até a sua aplicação em ambiente laboratorial e prático.

5.1 Planejamento dos Experimentos

Para o início dos experimentos foi definido quais seriam os ambientes em que eles iriam ocorrer. Para os treinamentos iniciais, além da comparação entre o desempenho dos algoritmos, foi planejado verificar o impacto da influência da utilização de GPU nos resultados quando comparado com a execução somente em CPU. Em ambos os ambientes foram utilizados o SO Ubuntu 16.04 com o *framwork* Caffe para o treinamento das redes com a interface gráfica e recursos do DIGITS (NVIDIA, 2016). O ambiente de teste em que utilizou somente CPU contou com um processador Intel Core i7 e 8 Gb de memória RAM, enquanto que o outro possuía um processador AMD e uma GPU NVIDIA gtx 960 com 4 Gb de memória decicada para a execução do treinamento.

A escolha do *framwork* Caffe se deve pela combinação de módulos e funções específicas que esse *framwork* apresenta, como a convolução e *pooling*, e determina as operações do sistema que auxiliam na comunicação entre os módulos (ICHINOSE et al., 2016). Assim, o Caffe apresenta uma arquitetura maleável com um desempenho interessante na variação do tamanho do *batch size*, que representa a quantidade de imagens utilizadas de uma vez pela rede. Diferenciando do *framwork* Tensorflow que tem uma queda de desempenho na utilização de um *batch size* reduzido. Essa característica possibilita a utilização do Caffe em dispositivos que apresentem desde um hardware mais limitado até um com mais recursos.

Para as diferentes bases foi adotada a divisão de 25% como teste e 75% como base de treinamento da rede. Esta divisão é feita de forma padrão pelo Caffe e pode ser configurado para valores diferentes caso desejado. Outro parâmetro que foi estabelecido e mantido para todos os testes que serão descritos foi o valor do *batch size*, em que no caso da rede AlexNet foi definido o valor de *batch size* de 60 e para a GoogLeNet de 30. Também foi definido o valar da taxa de aprendizagem em 0.001, pois apresentou um impacto melhor nos resultados do que os valores 0.1 e 0.0001.

Nos treinamentos foram alterados os parâmetros da quantidade de épocas, que é a quantidade de vezes que a base de treinamento passa pela rede, e também foram executadas bases com diferentes quantidades de imagens por classe. Todos estes parâmetros foram definidos para avaliar o impacto de diferentes fatores no treinamento e no resultado obtido, verificando o quanto a variação de épocas e de tamanho da base pode aumentar a acurácia da rede e melhorar a taxa de reconhecimento. Assim, podendo escolher o melhor resultado obtido para ser aplicado não só em experimentos de laboratório, mas também para ser embutido no protótipo (DOURADO; PEDRINO, 2018).

Após ser definido quais treinamentos seriam feitos foram iniciados os testes que comparam os treinamentos das redes AlexNet e GoogLeNet com a execução somente em CPU e GPU. O objetivo deste teste foi de verificar o impacto da utilização de GPU no tempo de treinamento e na acurácia obtida em comparação com o treinamento em CPU com a utilização de uma base de imagens. Também foi realizada a variação do parâmetro épocas para medir como esse valor influenciava no treinamento.

Definido quais os recursos que seriam utilizados para o treinamento houve o aumento da base de dados em relação a quantidade de classes, ampliando para sete, e número de amostra por objetos classificados. Nesta etapa foram utilizados três valores de amostras por classe e a mesma variação de épocas do teste de CPU e GPU. Após a finalização do treinamento com sete classes, a base de dados foi aumentada a fim de classificar dez objetos, porém mantendo a mesma variação de épocas. Com a finalidade de obter valores comparativos com os resultados obtidos com o treinamento das redes utilizando a base de dez classes foi treinado um classificador utilizando o algoritmo SVM.

Finalizado os treinamentos das redes e o comparativo com SVM foi dado inicio aos testes laboratoriais e práticos em que foram utilizadas as redes convolucionais treinadas com dez classes. Nesta etapa foram obtidas imagens do ambiente de teste e classificadas em laboratório com a finalidade de medir sua acurácia. Então foram elaborados os testes práticos utilizando as CNNs treinadas com dez classes embutidas no protótipo.

Desde o inicio do planejamento do trabalho não foi previsto a realização dos testes com deficientes visuais. Não houve esta etapa por não estar entre os objetivos primários uma vez que o protótipo, anterior a implementação da classificação de objetos e estados, foi testado com deficientes visuais. Desta forma, o objetivo definido foi a adição desta nova função na arquitetura.

Na Figura 5.1 é mostrado um diagrama com a sequência em que os testes foram realizados.

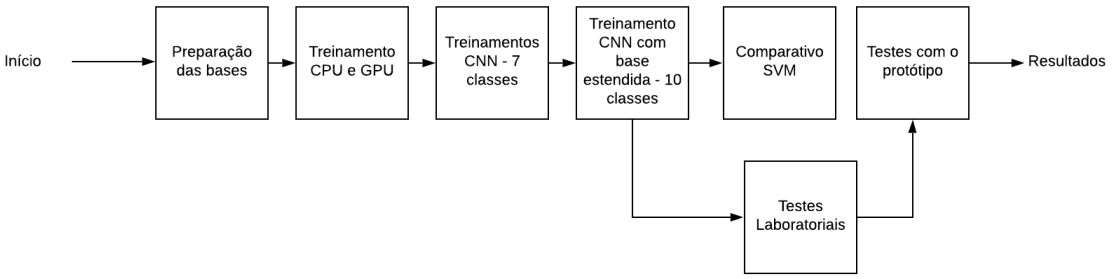


Figura 5.1: Diagrama do planejamento de testes realizado.

5.2 Treinamentos CPU e GPU

Para a realização do treinamento das CNNs foi construída inicialmente uma base composta por três categorias de imagens que são portas gavetas e fogões. A base foi feita com imagens retiradas do ImageNet, que é uma base com imagens de uso gratuitos para fins científicos, e conta com 300 imagens em cada categoria, totalizando 900 imagens nesta base inicial. Em relação a sua separação entre treinamento do algoritmo e base de testes, ela é feita de forma automática pelo framework que separa uma porcentagem da base definida pelo usuário. No caso, foram utilizados 25% do total da base para serem utilizadas como testes e 75% para o treinamento. No experimento foi feita a variação de épocas nos valores de 30, 60, 90, 120 e 150. Os parâmetros analisados para a obtenção dos resultados nos experimentos parciais foram o tempo de treinamento e a acurácia da rede. Na Figura 5.2 é mostrado um exemplo das imagens que fazem parte da base de treinamento.

Nas Tabelas 5.1 e 5.2 são mostrados os resultados obtidos, com a execução da base descrita acima, com utilização somente do poder computacional da CPU.

Nas Tabelas 5.3 e 5.4 são mostrados os resultados obtidos com a execução em GPU das redes AlexNet e GoogLeNet com os mesmos parâmetros definidos anteriormente.

A partir dos resultados obtidos com esta primeira base é possível observar que ambos os valores de acurácia são próximos quando comparadas as redes com suas execuções em CPU e

Figura 5.2: Exemplo da base utilizada na comparação cpu x gpu.

Tabela 5.1: AlexNet CPU

Época	Acurácia	Tempo de treinamento
30	59.16%	2h 35min
60	65.41%	5h 09min
90	66%	7h 36min
120	67.6%	9h 45min
150	70%	13h 34min

Tabela 5.2: GoogLeNet CPU

Época	Acurácia	Tempo de treinamento
30	67.5%	4h 57min
60	72%	10h 13min
90	70.22%	16h 58min
120	75.5%	20h 49min
150	75.6%	24h

Tabela 5.3: AlexNet GPU

Época	Acurácia	Tempo de treinamento
30	51.25%	2min 17s
60	67.9%	4min 32s
90	70.83%	6min 34s
120	69.16%	9min 48s
150	70.4%	12min 19s

Tabela 5.4: GoogLeNet GPU

Época	Acurácia	Tempo de treinamento
30	65.41%	4min 10s
60	70.41%	8min 13s
90	68.75%	12min 15s
120	77.08%	16min 18s
150	74.16%	20min 26s

GPU em uma mesma quantidade de épocas. O valor da acurácia apresenta pequenas alterações devido a forma que é calculado na rede, pois diferente dos demais métodos que utilizam uma forma estatística para a realização do cálculo, voltado para a exatidão, o *deep learning* busca a aproximação. O valor que mais difere é o tempo de treinamento em que ocorreu uma redução considerável quando comparado os ambientes de treinamento, pois a utilização de GPU consegue aperfeiçoar as operações que ocorrem com as matrizes e proporcionar esta drástica redução no tempo. Assim, o uso de GPU se mostra vantajoso para a aceleração do treinamento, possibilitando alterações de base e de parâmetros de forma mais eficiente. Porém, o seu uso não influência na acurácia da rede.

Os dados referentes a comparação da acurácia pode ser observado graficamente na Figura 5.3 e os que mostram a vantagem de tempo de processamento nas Figuras 5.4 5.5.

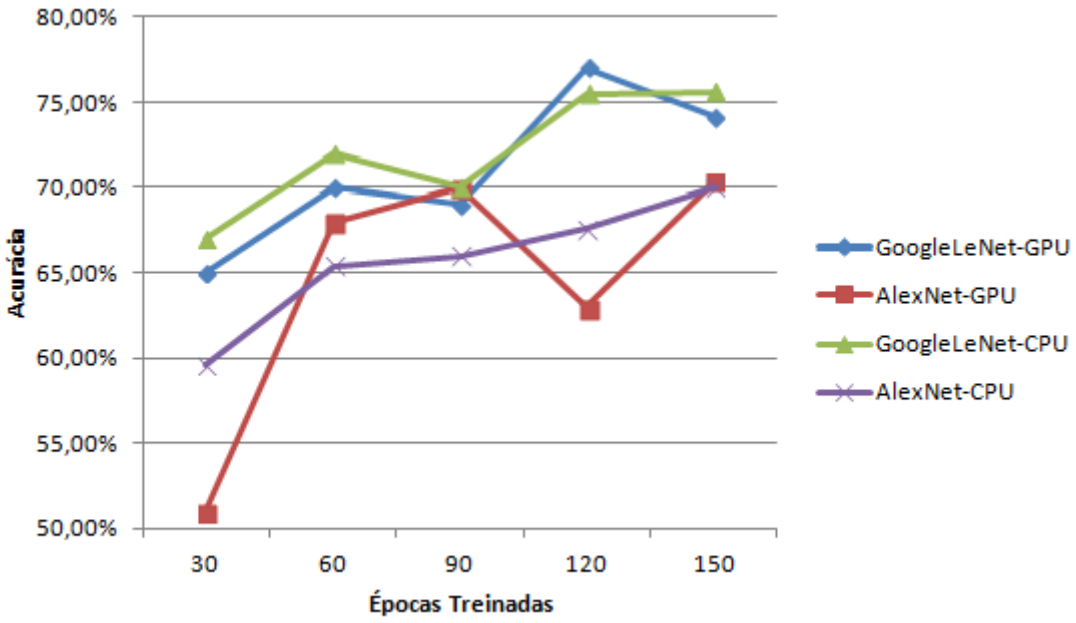


Figura 5.3: Comparação gráfica da acurácia obtida com o uso de CPU x GPU.

5.3 Treinamentos das Redes Convolucionais

Após os testes iniciais em que foi comparado o desempenho do treinamento com a execução em CPU e GPU, a base de imagens foi ampliada e estabelecida novas quantidades de dados para verificar qual o impacto da base de treinamento na acurácia resultante da rede.

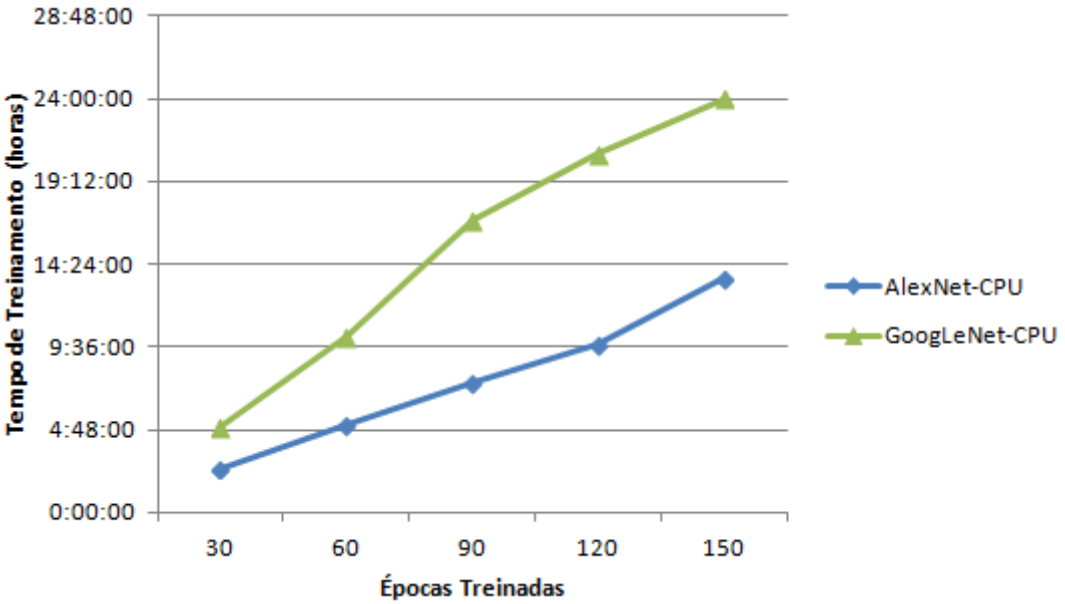


Figura 5.4: Comparação gráfica do tempo de treinaemnto utilizando CPU.

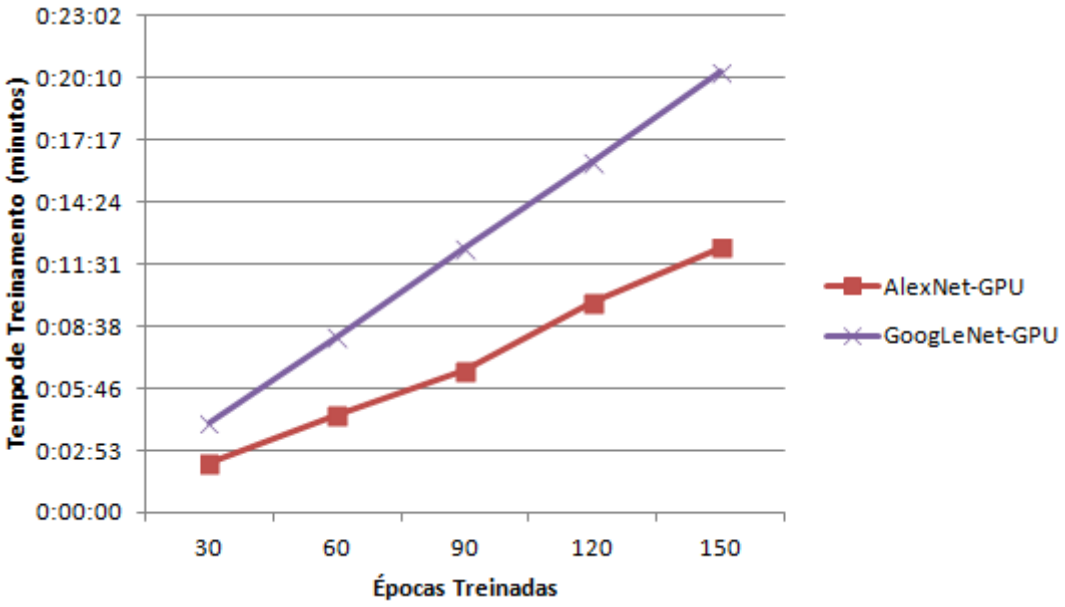


Figura 5.5: Comparação gráfica do tempo de treinaemnto utilizando GPU.

5.3.1 Treinamento das Redes com 7 Classes

Depois de decidido que o treinamento seria realizado com a utilização de GPU, foram construídas três novas bases, com o mesmo método da primeira, utilizando imagens de uso gratuito do ImageNet e mesclando com fotos tiradas especificamente para a base. Nesta nova etapa foi testado o impacto na acurácia da rede do aumento da quantidade de imagens por classe e a utilização de maior quantidade de classes. As Três novas bases contaram com sete classes de objetos, sendo elas fogões, geladeiras, portas, gavetas, escadas, cadeiras e armários, como

mostrado da Figura 5.6, variando a quantidade de imagens por classe em 300, 500 e 1000.

Figura 5.6: Exemplo da base com 7 classes de objetos.

Nas Tabelas abaixo são mostrados os resultados obtidos com as execuções das bases com 7 classes de objetos e 300, 500 e 1000 imagens em cada uma com as rede AlexNet e GoogLeNet. Como feito no primeiro experimento, foi analisada a acurácia e o tempo de treinamento variando a quantidade de épocas em 30, 60, 90, 120 e 150.

Tabela 5.5: AlexNet com uso de GPU, 300 imagens por classe e 7 objetos.

Época	Acurácia	Tempo de treinamento
30	41.11%	4min 11s
60	57.03%	8min 16s
90	65.92%	12min 11s
120	69.62%	16min 30s
150	82.77%	20min 37s

Tabela 5.6: GoogLeNet com uso de GPU, 300 imagens por classe e 7 objetos.

Época	Acurácia	Tempo de treinamento
30	58.88%	9min 23s
60	72.22%	18min 31s
90	77.59%	29min 35s
120	82.77%	40min 23s
150	79.44%	46min 48s

Analisando os resultados obtidos mostrados nas tabelas e na Figura 5.7 e Figura 5.8, é possível observar que a diferença entre a acurácia das redes se mantiveram com uma variação semelhante das obtidas no primeiro experimento, com a base de 3 categorias e 300 imagens

Tabela 5.7: AlexNet com uso de GPU, 500 imagens por classe e 7 objetos.

Época	Precisão	Tempo de treinamento
30	48.33%	5min 56s
60	65.77%	11min 48s
90	72.22%	18min 02s
120	77.55%	24min 27s
150	79.77%	30min 34s

Tabela 5.8: GoogLeNet com uso de GPU, 500 imagens por classe e 7 objetos.

Época	Acurácia	Tempo de treinamento
30	67.11%	16min 14s
60	78.88%	32min 33s
90	82%	48min 51s
120	82.77%	1h 02min
150	82.22%	1h 18min

Tabela 5.9: AlexNet com uso de GPU, 1000 imagens por classe e 7 objetos.

Época	Acurácia	Tempo de treinamento
30	65.44%	11min 24s
60	76.22%	21min 37s
90	80.83%	34min 33s
120	83.11%	45min 32s
150	83.27%	55min 58s

Tabela 5.10: GoogLeNet com uso de GPU, 1000 imagens por classe e 7 objetos.

Época	Acurácia	Tempo de treinamento
30	75.31%	30min 43s
60	81.52%	1h 1min
90	84.63%	1h351min
120	85.53%	2h
150	86.1%	2h 36min

em cada, em uma mesma época. Também pode ser observado que o aumento da quantidade de épocas promove o aumento da acurácia, porém a tendência é que a diferença entre as acurácias diminuam, alcançando assim um valor mais estável, tal que o aumento da quantidade de épocas promova uma variação pequena nos resultados obtidos a partir de um certo valor.

5.3.2 Treinamentos das Redes com 10 Classes

Para esta etapa foi utilizada a base de 500 imagens por categoria devido a uma variação pequena com a base de 1000 imagens por categoria, aumentando a quantidade de classes treinadas

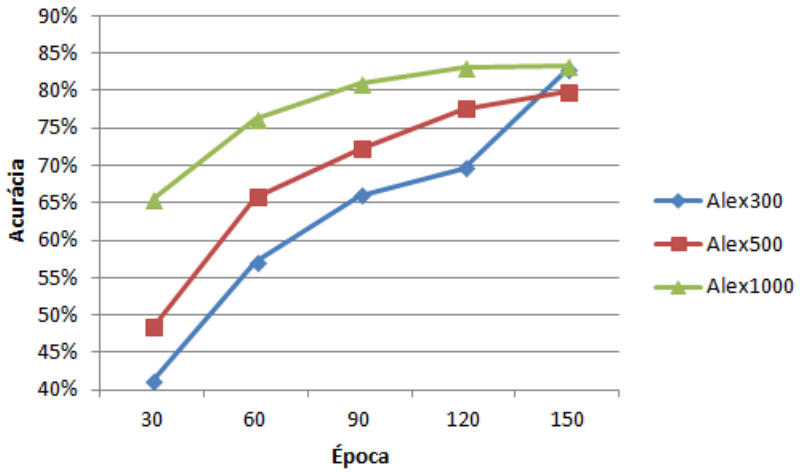


Figura 5.7: Comparativo entre as acurácias obtidas com AlexNet.

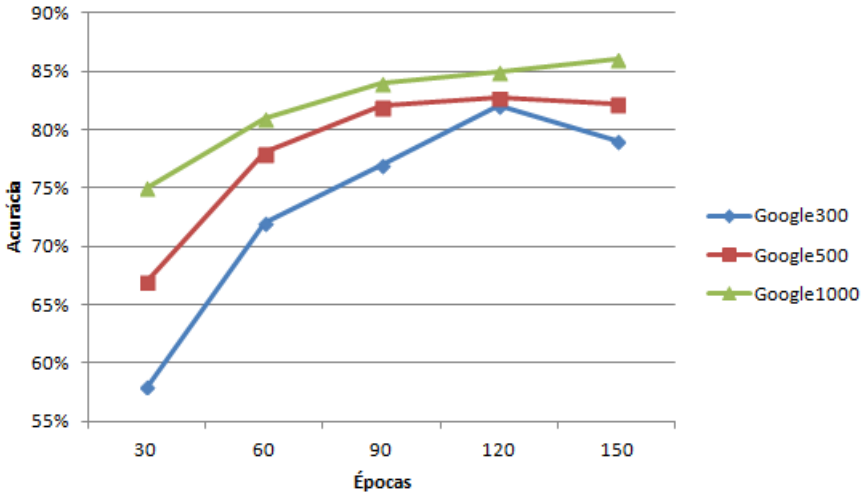


Figura 5.8: Comparativo entre as acurácias obtidas com GoogLeNet.

para dez. Dentre estas classes foram divididas as portas e gavetas em portas livres, bloqueadas, gavetas abertas e fechadas com a finalidade de proporcionar a identificação de estados destas duas classes de objetos. Assim, a base utilizada para o experimento principal conta com as classes de geladeiras, fogões, armários, cadeiras, mesas, escadas, portas livres e bloqueadas, gavetas abertas e fechadas e mesas. Um exemplo da base é mostrado na Figura 5.9.

Os resultados obtidos são mostrados nas Tabelas 5.11 e 5.12.

Analisando os valores das acurácias obtidas com o treinamento das redes AlexNet e GoogLeNet com a base estendida para dez classes e comparando com a de sete classes, ambas as bases com 500 imagens por classe, é possível ver que em ambos os casos os resultados são bem próximos. Assim, possuindo apenas pequenas alterações de porcentagem.

Quando é analisado o tempo de treinamento é contatado a mesma tendência apresentada

Figura 5.9: Exemplo de objetos da base com 10 classes de objetos.

Tabela 5.11: AlexNet com uso de GPU, 500 imagens por classe e 10 objetos.

Época	Acurácia	Tempo de treinamento
30	51.59%	8m 37s
60	64.62%	17m 18s
90	72.95%	25m 281s
120	77.34%	33m 14s
150	79.09%	40m 30s

Tabela 5.12: GoogLeNet com uso de GPU, 500 imagens por classe e 10 objetos.

Época	Acurácia	Tempo de treinamento
30	60.23%	22m 08s
60	70.31%	43m 54s
90	74.72%	1h 05m
120	78.6%	1h 28m
150	81%	1h 49m

nos demais testes em que o treinamento da GoogLeNet é mais lento, porém alcançando valores de acurácia um pouco maiores do que os da AlexNet. O tempo de treinamento apresenta um aumento em ambas as redes com a utilização de mais classes. Isso se deve a maior quantidade de imagens utilizadas.

5.4 Comparativo com SVM

Para comparar a acurácia obtida com o treinamento das redes AlexNet e GoogLeNet com outro algoritmo foi utilizado o software WEKA para a realização da simulação do algoritmo

SVM. O WEKA simula diferentes algoritmos e pode tratar dados para serem utilizados aplicando filtros e técnicas antes de serem treinados. Assim, é possível realizar a extração de características das imagens para serem utilizadas na execução dos testes. A mesma base de imagens que foi colocada para o treinamento das redes de *deep learning*, com dez classes e cada uma com 500 imagens, foi aplicada nos testes comparativos com o SVM.

5.4.1 Configuração do teste comparativo

Com a utilização do WEKA foi realizado o treinamento, sendo extraídas as características em uma etapa anterior. O WEKA permite que a extração de características ocorra com a utilização de filtros que são importados para ele e podem ser aplicados diretamente em um conjunto de imagens. Para a realização desta etapa é feito um arquivo com a extensão arff que contém um cabeçalho que discrimina quais classes serão rotuladas e os dados que serão submetidos ao processo. No corpo do arquivo é adicionado o nome da imagem (incluindo seu formato) e a classe que a rotula. Um exemplo de cabeçalho e de corpo do arquivo arff é mostrado na Figura 5.10.

```
@relation house_itens

@attribute filename string

@attribute class {CADEIRA,ESCADA,GELADEIRA,MESA}

@data

cadeira1.png,CADEIRA
cadeira10.png,CADEIRA
cadeira100.png,CADEIRA
cadeira101.png,CADEIRA
cadeira102.png,CADEIRA
cadeira103.png,CADEIRA
cadeira104.png,CADEIRA
cadeira105.png,CADEIRA
cadeira106.png,CADEIRA
cadeira107.png,CADEIRA
cadeira108.png,CADEIRA
cadeira109.png,CADEIRA
cadeira11.png,CADEIRA
```

Figura 5.10: Exemplo de arquivo arff.

Este arquivo deve ser colocado na mesma pasta das imagens e será aberto pelo WEKA na aba Processo. Também é utilizado o caminho do arquivo para a configuração do filtro que será responsável por realizar a extração de características das imagens cujos nomes estão no arquivo

arff.

Na mesma aba Processos ocorre a seleção do filtro que será utilizado. Para a extração de características dos testes realizados com o SVM foi utilizado o Filtro de Histograma de Borda (*Edge Histogram Filter*) que extrai características baseadas nos contornos dos objetos. Em seu processo de extração de são categorizadas cinco tipos de bordas. As bordas são verticais, horizontais, com angulação de 45graus, angulação de 135 graus e bordas que não possuam direcionamento (PRAJAPATI; NANDANWAR; PRAJAPATI, 2016). A Figura 5.11 mostra as características extraídas com o filtro.

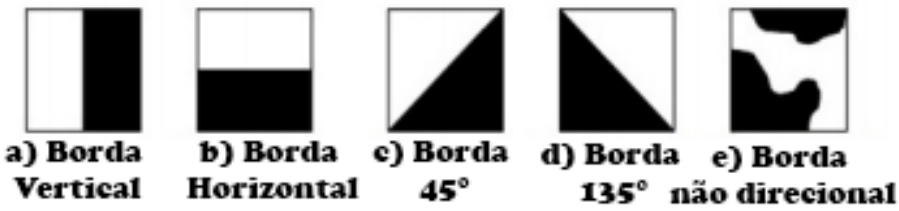


Figura 5.11: Características extraídas com o filtro, adaptado de (PRAJAPATI; NANDANWAR; PRAJAPATI, 2016).

Após a etapa descrita anteriormente, as características extraídas foram utilizadas na aba Classificação. Nesta aba ocorre a seleção do algoritmo que será utilizado para treinar as características anteriormente extraídas. Para o treinamento foi escolhido o algoritmo SVM devido a sua ampla utilização na literatura. Para a execução foi usado a implementação SMO (Sequential Minimal Optimization) (KEERTHI et al., 2001) do algoritmo SVM que é importado pelo próprio WEKA.

Por padrão o SVM é um classificador binário, porém existem técnicas que possibilitam a classificação multiclasse. A implementação utilizada faz uso da classificação em pares para a resolução de problemas multiclasse (1-vs-1) (WEKA, 2016). A resposta do problema é obtida através do treinamento individual de SVMs, assim, são criados instâncias distintas que caracterizam cada classe treinada. As instâncias treinadas são utilizadas de forma comparativas, assim, possibilitando que multiplas classes sejam utilizadas.

5.4.2 Resultados obtidos

A Tabela 5.13 mostra os valores obtidos com a variação da de lasses, porém mantendo a mesma quantidade de imagens por classe já utilizada no teste anterior.

Como mostra os resultados obtidos na Tabela 5.13, a variação da quantidade de classes em cada base impacta diretamente na acurácia dos resultados obtidos com o treinamento. Assim,

Tabela 5.13: Resultado do algoritmo SVM com variação da quantidade de classes.

Quantidade	Acurácia
2	83.5%
4	70.2%
6	55.66%
8	48.77%
10	47.68%

quanto menor a quantidade de classes melhor se mostra o desempenho do SVM. Tal impacto chega ao ponto de mostrar essa diferença de desempenho com mais clareza na base com dez classes de objetos que apresentou um resultado de 47.68%.

5.4.3 Conclusão

Comparando com os resultados obtidos na execução da mesma base com dez categorias e quinhentas imagens em cada uma, as CNN AlexNet e GoogLeNet apresentaram 79.09% e 81% no valor de sua acurácia, resultados mostrados nas Tabelas 5.11 e 5.12 . Estes são bem superior ao desempenho do SVM para a mesma base. A diferença nos resultados entre os métodos de CNN e SVM se dá pela forma que os dados são utilizados por cada um, pois as CNN's utilizam as imagens de forma a extrair características de uma forma mais direta do que o SVM que necessita de um algoritmo de extração antes de ser utilizado, como já explicado no Capítulo 2.

Mesmo observando os melhores resultados das CNNs, quando comparados com os valores obtido com o uso do SVM, existem outros algoritmos que seriam interessantes de serem aplicados na comparação levando em conta o modo dos seus funcionamentos. Entre as técnicas que podem apresentar resultados comparativos interessantes estão o KNN e a árvore de decisão, pois ambos já possibilitam a classificação multiclasse e poderiam explorar parâmetros mais próximos das técnicas utilizadas no trabalho. Diferentemente do SVM que é um classificador binário em que são utilizadas técnicas de comparação para problemas multiclasse.

5.5 Resultados Laboratoriais

A partir das redes treinadas mostradas, nas Tabelas 5.11 e 5.12, foram realizados testes com novas imagens em ambiente de laboratório com a finalidade de realizar uma simulação. Estas imagens foram obtidas de ambientes do Departamento de Computação (DC) da Universidade Federal de São Carlos (Ufscar), utilizando o dispositivo microssoft Kinect para a captura, que é

a mesma câmera que é utilizada no protótipo. Para cada categoria foram utilizadas dez imagens para serem classificadas com as redes, totalizando cem imagens na base de validação para os testes em laboratório. Neste teste não é levado em consideração o tempo de reconhecimento da imagem, pois foi buscado a taxa de reconhecimento e não o tempo de identificação por não possuir hardware dedicado, como é o caso do protótipo.

Utilizando a rede AlexNet para executar a base de validação em laboratório foi obtido um acerto de 86% da totalidade da base. Tratando individualmente cada categoria testada, as imagens de porta aberta, escada e cadeira obtiveram 100% de acerto, enquanto que as categorias fogão e gaveta aberta obtiveram 90%, geladeira, porta fechada e armário 80%, armário, mesa e gaveta fechada 70%. Já com rede GoogLeNet foi obtido 91% de acerto na totalidade da base. Os valores obtidos de forma individual, por categoria, foram 100% em porta fechada, fogão e cadeira, 90% em escada, gaveta fechada, porta aberta e armário e 80% e 70%, respectivamente, nas categorias geladeira e mesa. Na Tabela 5.14 é mostrada a média da porcentagem com que os objetos foram identificados corretamente com a utilização da AlexNet e GoogLeNet nos testes de laboratório.

Tabela 5.14: Média da porcentagem de certeza para cada classe identificada.

Classe	AlexNet	GoogLeNet
Armário	72%	79.2%
Cadeira	95.6%	87.2%
Escada	98%	96.23%
Fogão	89.41%	97.5%
Gaveta Aberta	75.55%	90.6%
Gaveta Fechada	65.48%	78.95%
Geladeira	77.5%	75%
Mesa	60%	65.9%
Porta Aberta	70.6%	82.22%
Porta Fechada	95.2%	80.2%

Com a análise dos dados obtidos com a execução da base de validação em laboratório foi possivel concluir que foram obtidos resultados positivos para ambas as redes, porém a GoogLeNet apresentou melhor desempenho com relação a quantidades de acertos e médias mais equilibradas na classificação individual das classes abordadas. Isso ocorreu principalmente nas categorias escada, porta aberta porta fechada, gaveta aberta e fechada que são consideradas mais críticas na classificação por possuírem estado e no caso da escada ser um potencial risco ao usuário.

5.6 Testes com uso do protótipo

Nesta seção será descrito o planejamento de testes utilizando o protótipo do colete para auxilio a deficientes visuais. Também será descrito o planejamento de testes com o protótipo e os resultados obtidos com a utilização do equipamento com o uso de ambas as CNN's estudadas. Por fim, uma análise dos valores obtidos no teste.

5.6.1 Planejamento dos testes para o protótipo

Além de verificar o resultado das redes convolucionais e compara-los com os valores obtidos no treinamento do SVM com a simulação utilizando o WEKA, também foram planejados testes para embutir as redes treinadas no protótipo. O objetivo do teste foi o de verificar no protótipo o desempenho da classificação realizada até então somente em laboratório. Outro parâmetro importante obtido, além da classificação correta, é o tempo que o sistema demora a retornar a resposta ao usuário quando solicitado. Então, foram definidos os parâmetros de tempo de classificação e a acurácia individual da qual foi requisitada a identificação. Para obter estes parâmetros foi implementada uma função que utiliza a imagem RGB do frame capturado pelo Kinect e armazena em um arquivo os valores do tempo de classificação e as primeiras cinco classificações que foram obtidas para o objeto requisitado com suas respectivas porcentagens de certeza.

O local escolhido para o teste com a rede embutida no protótipo foi o Departamento de Computação (DC) da Universidade Federal de São Carlos (Ufscar) por contemplar todos os objetos definidos para a classificação. Além disso, o DC apresenta ambientes internos que se assemelham a uma casa, como cozinhas e salas, e possibilitam os testes em ambientes mais amplos que podem representar internamente diferentes locais desconhecidos.

Desta maneira, a execução dos testes buscou contemplar todos os objetos treinados que foram planejados para serem classificados. Assim, possibilitando que mesas, armário, cadeira e porta sejam classificadas no ambiente do laboratório GAPIS no andar superior, a escada entre os andares e a verificação de itens como geladeira e fogão nas áreas de cozinha presente em ambos os pisos do deparrtamento.

Os testes realizados consistiram na utilização do colete nos ambientes dos pisos inferior e superior do DC que são mostrados nas Figuras 5.12 e 5.13. Nos ambientes do departamento foram utilizadas ambas as redes treinadas, AlexNet e GoogLeNet, para que pudesse ser estabelecida uma comparação entre as CNNs e qual proporcionaria melhor desempenho para ser

utilizada no equipamento.

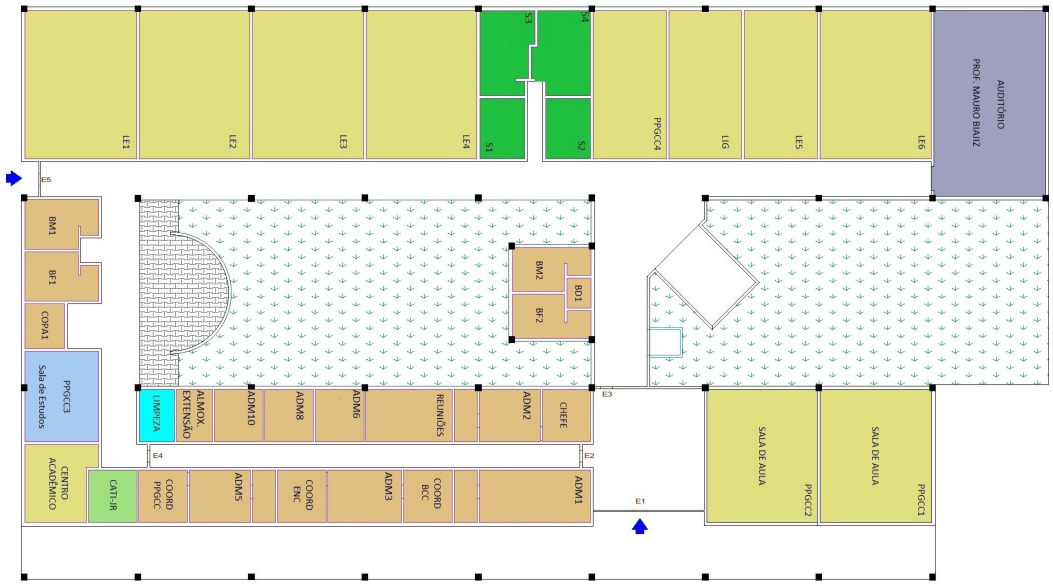
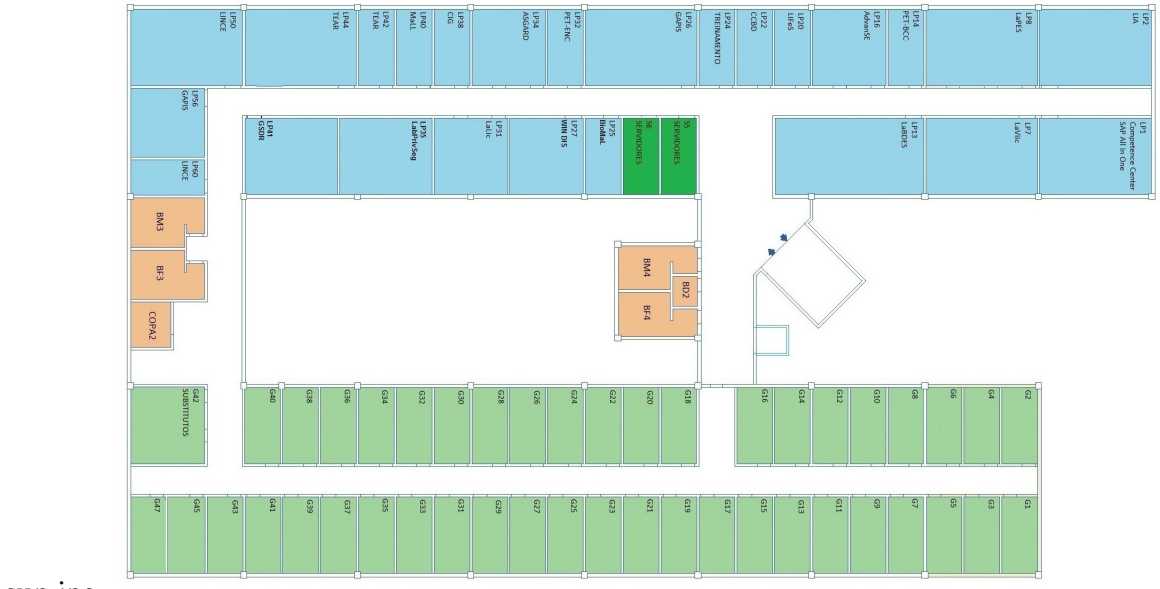


Figura 5.12: Mapa do Departamento de Computação do piso térreo.



sup.jpg

Figura 5.13: Mapa do Departamento de Computação do piso superior.

5.6.2 Resultados Protótipo

Nesta seção serão abordados os resultados dos testes com a redes AlexNet e GoogLeNet aplicado ao protótipo para deficientes visuais no ambiente de teste já descrito na seção anterior. É importante ressaltar que dentre os objetos treinados para ser classificado pela rede será dado destaque para as portas e gavetas, por ter a intenção de mostrar a classificação de estados, e

a classificação de escadas pelo risco potencial que uma queda pode causar ao usuário. Os parâmetros utilizados para analisar os resultados obtidos serão a acurácia e o tempo para o resultado ser obtido. Na Figura 5.14 são mostradas imagens obtidas com o uso do dispositivo Kinect no teste com o protótipo.

Figura 5.14: Imagens obtidas com o uso do dispositivo kinect.

Na Tabela 5.15 é mostrados o resultado de uma amostra de cada classe de imagens obtidas pelo colete e utilizada na classificação pela rede AlexNet e GoogLeNet como exemplo das certezas obtidas. Em ambos os testes o tempo de identificação da imagem levou entre 2 e 3 segundos.

Tabela 5.15: Resultado obtido com uma amostras de cada classe na execução das redes AlexNet e GoogLeNet.

Classe	AlexNet	GoogLeNet
Armário	71.91%	66.9%
Cadeira	62.62%	81%
Escada	52.5%	99.36%
Fogão	66%	66%
Gaveta Aberta	62.1%	76%
Gaveta Fechada	39.9%	54.76%
Geladeira	69.63%	82.82%
Mesa	86.97%	70.31%
Porta Aberta	76.51%	80.86%
Porta Fechada	68.69%	63.87%

Na Tabela 5.16 é mostrada uma comparação com a média das acurácias obtidas com as imagens classificadas por cada rede. Para a realização do calculo da média foram feitas classificações com três imagens de cada categoria.

Os valores obtidos nos testes laboratoriais e embutidos, mostrados nas Tabelas 5.14 e 5.16, são separados e comparados individualmente segundo a sua rede na Figura 5.15 e Figura 5.16.

Em relação ao tempo de classificação, os valores obtido tanto com o uso da AlexNet como

Tabela 5.16: Média de certeza por classe com com utilização do protótipo.

Classe	AlexNet	GoogLeNet
Armário	66.9%	74.61%
Cadeira	81%	66.25%
Escada	99.36%	99.96%
Fogão	66%	65.6%
Gaveta Aberta	76%	65.93%
Gaveta Fechada	54.76%	61.4%
Geladeira	69.63%	82.82%
Mesa	86.97%	70.31%
Porta Aberta	76.51%	80.86%
Porta Fechada	68.69%	63.87%

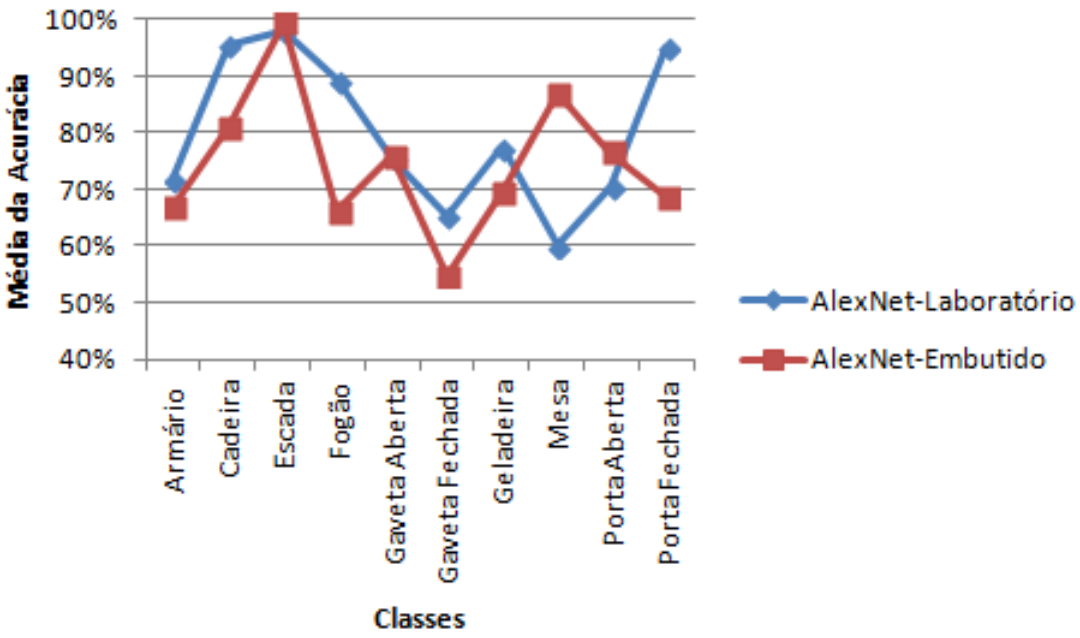


Figura 5.15: Comparação entre os resultados laboratoriais e embutido com a rede AlexNet.

pela GoogLeNet se mostraram bem próximos, sendo obtido valores entre 2 e 3 segundos após a requisição da classificação pelo comando de voz. Mostrando que mesmo com estruturas e processos diferentes para a realização da extração de características e classificação, existe uma dependencia do hardware que é utilizado em relação ao tempo.

Com os valores de acurácia utilizados, como mostrados na Tabela 5.15, é possível observar que ambas as redes conseguiram classificar corretamente os objetos que foram treinadas para serem reconhecidos. Mesmo quando o objeto requerido para classificação não obtém porcentagens majoritárias, o sistema retorna a resposta como "parece" antes de dizer qual a classificação obtida. Outra semelhança que pode ser observada é a porcentagem majoritária na classificação

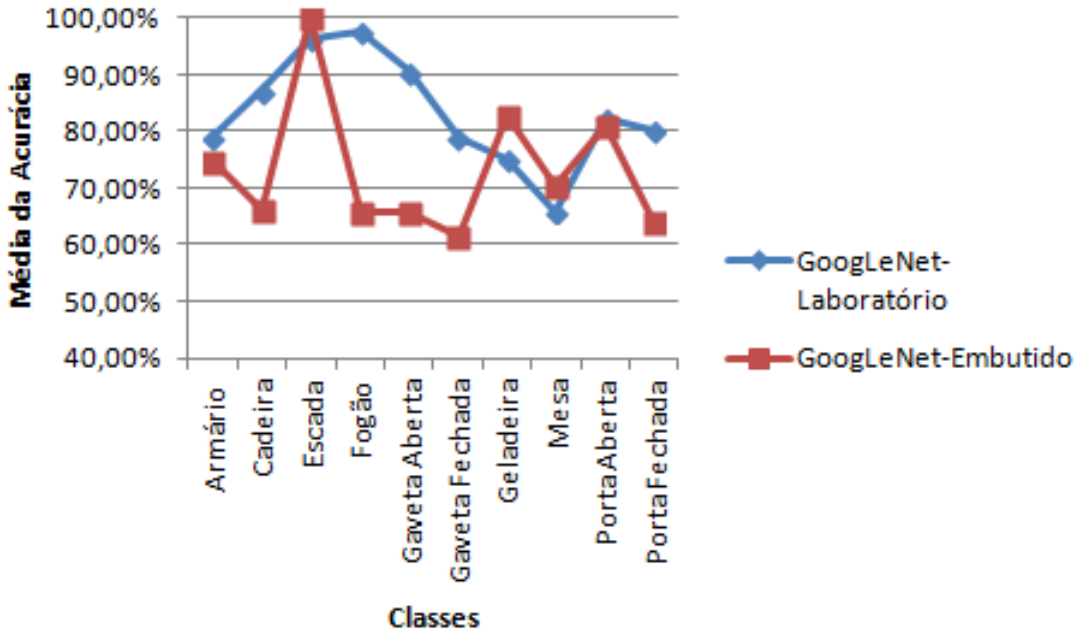


Figura 5.16: Comparação entre os resultados laboratoriais e embutido com a rede GoogLeNet.

de pontos importantes para o reconhecimento, como os objetos que possuem estados e escadas.

Em relação as porcentagens obtidas nas amostras, com a AlexNet a porcentagem de Gaveta aberta, Gaveta fechada, Porta aberta, porta fechada e escadas foram, respectivamente, de 72.9%, 38.9%, 65.6%, 88% e 98.1%. Já para a GoogLeNet, para as mesmas classes foram obtidas os valores de 62.1%, 39.9%, 84.9%, 62.6% e 99.9%. Para as demais classes também há discrepâncias, como na amostra de cadeira e Geladeira que foi obtido 99.3% e 54.7% com a AlexNet e 69.9% e 78.6% com a GoogLeNet.

Na execução de ambos os testes a classe Gaveta Fechada, apesar de aparecer em primeiro lugar quando classificado, ocorreu com uma porcentagem baixa de certeza que se enquadrou em retornar ao usuário como "parece". Isso ocorre por frequentemente este objeto estar relacionado com mesas, o que durante o treinamento mesmo com o usou de rótulos distintos para estes objetos acaba afetando a classificação. Quando comparado os resultados obtidos nas Tabelas 5.14 e 5.16, que correspondem respectivamente aos resultados em laboratório e os obtidos com o teste no colete, é possível observar valores maiores nos resultados obtidos em laboratório. Assim, mostrando que os resultados de laboratório podem mostrar o desempenho que é possível alcançar quando embutida a rede no protótipo para a sua utilização prática.

Mesmo com resultados satisfatórios na aplicação prática há uma porcentagem um pouco mais baixa de certeza na classificação de gavetas fechadas com ambas as redes. Isto ocorre devido a proximidade da classe mesa com a de gaveta fechada, assim, fazendo com que em alguns casos esta última seja classifica como mesa ou até mesmo armário, sendo obtido na

classificação do estado de Gavetas com uso de da AlexNet os valores de 76% para aberta e 54.7% para fechada. Já com o uso da GoogLeNet foi obtido 65.9% e 61.4%.

Os resultados obtidos mostram que a classificação com o uso do protótipo se mostrou eficiente, sendo possível classificar Escadas com mais de 95% de certeza tanto com a AlexNet como com GoogLeNet e avaliar a questão do estado dos objetos, Portas e Gaveta Aberta, de uma forma correta apesar das porcentagens mais baixas da classe de Gaveta Fechada.

5.7 Considerações Finais

A partir dos resultados obtidos que foram mostrados no capítulo, pode ser concluído que a utilização de GPU para a realização do treinamento das CNNs torna o treinamento mais rápidos, apesar de não exercer influencia sobre a acurácia obtida. Com a realização dos treinamentos das redes AlexNet e GoogLeNet foram obtidos resultados bons, sendo considerada para a utilização nos testes laboratoriais e embutidos a base de 500 imagens por classe treinadas por 150 épocas em ambas as redes. Nas redes treinadas para serem embutidas foi alcançada a acurácia de 79,09% para AlexNet e 81% com a GoogLeNet.

Com a realização dos testes executados em laboratório foi mostrado que as redes treinadas conseguiram classificar corretamente os objetos treinados com valores de certezas confiáveis. Os resultados obtidos com o uso do protótipo apresentaram uma certeza menor do que os valores obtidos nos testes em laboratório, porém foram resultados satisfatórios e que apresentaram uma correta classificação dos objetos treinados. Tal diferença de valores pode ser atribuida a utilização da rede juntamente com o sistema do protótipo, o que não ocorreu com os testes em laboratório. Para aumentar os valores obtidos para a rede embutida podem ser realizadas melhorias como o aumento da base de imagens para o treinamento ou a parametrização do sistema dentro do prórprio protótipo.

Capítulo 6

CONCLUSÃO

Este capítulo apresenta as conclusões obtidas com o desenvolvimento do trabalho e os possíveis trabalhos futuros.

6.1 Conclusões do Trabalho

A aplicação de tecnologias nas mais variadas áreas podem proporcionar desenvolvimentos em diversos âmbitos. Entre as áreas que podem ser beneficiadas está o auxílio a deficientes visuais. Essa deficiência causa inúmeras dificuldades e limitações no dia a dia e prejudica a percepção do ambiente. O desenvolvimento da tecnologia pode ser uma forma de melhorar a qualidade de vida destas pessoas e proporcionar a elas recursos que permitam perceber melhor o ambiente que as cerca, além de ferramentas manuais de auxílio e adaptações feitas no ambiente. Um exemplo de área que pode ser explorada para este propósito é o reconhecimento de imagens. Mesmo não sendo uma área nova na computação, os avanços tecnológicos permitiram melhores abordagens nos problemas da área e o desenvolvimento de novas aplicações.

O *deep learning* é uma tecnologia que vem ganhando destaque na área do reconhecimento de imagens. O seu funcionamento consiste em uma rede que analisa e classifica imagens a partir de uma base de conhecimento do classificador que pode ser previamente treinada. Assim, fazendo uso de técnicas de *deep learning*, o trabalho desenvolvido teve como objetivo o estudo de métodos de reconhecimento de objetos para o desenvolvimento de um classificador para ambientes internos, classificando uma série de objetos e os seus estados. Também, desenvolvendo um módulo para o protótipo de auxílio a deficientes visuais (DOURADO; PEDRINO, 2018) com a realização de testes laboratoriais e embarcados.

Com a comparação do treinamento das redes utilizando somente o poder computacional da

CPU e GPU, em ambientes que foram dadas condições semelhantes de sistemas a ambos os testes, variando apenas a fonte de processamento e mantendo a mesma variação do parâmetro época foi constatado que não havia influência da fonte escolhida na acurácia obtida. Ambos os treinamentos apresentaram resultados próximos e o impacto maior que pode ser observado é no tempo de treinamento. A duração do treinamento mostrou que o uso de GPU é mais interessante por executar em menor tempo para todas as redes treinadas em suas variações de parâmetros.

A realização dos treinamentos utilizando a base de sete classes com a variação de épocas em 30, 60, 90, 120 e 150, e do tamanho da base em 300, 500 e 1000 imagens por classe, mostraram que a alteração destes parâmetros impacta no resultado da acurácia obtida. Com o aumento de ambos os valores o tempo de treinamento aumenta, isso também ocorre no valor da acurácia obtida. Porém, com o aumento da quantidade de épocas pode ser observado que em uma mesma base a diferença das acurácias obtidas tende a diminuir cada vez mais e os valores resultantes diminuindo a sua variação.

Com o aumento da quantidade de classes na base de imagens de sete para dez os valores de acurácias obtidas no treinamento foram altos e a quantidade de 500 imagens por classe mostrou-se um tamanho bom para futuras adições de objetos a serem identificados. Assim, a rede treinada com 500 imagens por classe por 150 épocas foi escolhida para ser utilizada nos testes laboratoriais e embutida no protótipo.

Em relação aos testes comparativos executados com o algoritmo SVM, o uso das CNNs mostrou-se mais vantajoso para treinamentos com maiores números de classes de objetos. Assim, para treinamentos com poucas classes a utilização do SVM é vantajosa. Porém, quando utilizado com mais classes de objetos, o SVM apresenta uma queda no valor da acurácia obtida. Como o objetivo para a utilização da rede no módulo era realizar o reconhecimento de uma gama maior de objetos, o SVM não se apresentou como uma forma favorável e corroborou com a utilização das CNNs para a aplicação no módulo.

Na execução dos testes práticos as redes escolhidas tiveram um bom desempenho. Os resultados mostraram que os valores obtidos com a execução em laboratório foram maiores dos que os alcançados com a utilização das redes embutidas no hardware, apesar de classificar corretamente os objetos treinados. Os valores resultante para ambas as redes quando embutidas é inferior ao que foi alcançado nos testes laboratoriais devido à rede estar submetida a todo um sistema, não somente a inserção da imagem para classificação.

O tempo de classificação obtido com ambas as redes utilizadas foi satisfatório, pois antes da classificação o sistema já deve ter detectado o obstáculo. Assim, a tarefa de classificação complementa a detecção e a atuação conjunta dos dois módulos consegue não só evitar um

impacto, mas também permite a identificação do que está obstruindo o caminho.

A partir dos resultados obtidos e do que foi comparado, é possível observar que a utilização das técnicas de *deep learning*, com a utilização das redes convolucionais AlexNet e GoogLeNet, apresentaram um bom desempenho na classificação de imagens. Com ambas as redes foram obtidos resultados satisfatórios que resultaram na classificação correta dos objetos e seus estados, tanto nos testes em laboratório como os que ocorreram com uso do protótipo.

Em relação ao desempenho das redes, a GoogLeNet apresentou melhor desempenho do que a AlexNet. Isso ocorre devido à diferença de arquitetura das redes. Essa diferença ocorre pela maior quantidade de convoluções proporcionada pelas camadas *Inceptions*, assim, reduzindo mais os parâmetros e obtendo mais detalhes do que esta sendo classificado. Isso pode ser observado na classificação de mesas e gavetas fechadas que podem ser distinguidas mais precisamente com a utilização da GoogLeNet.

O módulo de reconhecimento de imagem e estados mostrou-se muito útil e com valores de acurácias confiáveis para ser embutido no sistema do protótipo, assim, além da detecção de obstáculos é possível classifica-los. Desta forma, os resultados obtidos com os testes práticos, após a adição do novo módulo, mostraram que pode ser feito o uso do equipamento como uma forma de auxílio a deficientes visuais, complementando os recursos já utilizados para locomoção e acessibilidade nos ambientes. O seu uso permite que o ambiente seja mais bem percebido e possíveis riscos a saúde das pessoas com essas limitações sejam previstos e evitados.

Comparando o trabalho desenvolvido com os estudados na literatura, o que mais se assemelha é o estudo feito por (BASHIRI et al., 2018). Em ambos os trabalhos é proposto a ideia de desenvolver tecnologias para o propósito de auxiliar deficientes visuais e fazendo a utilização de *deep learning*. Outra semelhança ocorre por parte dos objetos que são identificados, BASHIRI busca classificar apenas três classes (portas, placas e escadas), duas das categorias classificadas também são identificadas no trabalho aqui apresentado (portas e escadas). Porém, no módulo de reconhecimento de imagem e estado a quantidade de categorias reconhecidas é ampliada para dez.

Em relação às diferenças, apesar do propósito semelhante, o trabalho desenvolvido nesta pesquisa visa a classificação para ambientes internos desconhecidos de uma forma mais ampla. Já em (BASHIRI et al., 2018) a classificação ocorre para ambientes específicos. O trabalho da literatura também difere em relação a técnica de classificação. Bashir usa para a classificação a extração de características uma parte da rede convolucional AlexNet e a classificação é realizada com a utilização do SVM, enquanto que na dissertação apresentada é utilizada a extração e classificação através da própria rede convolucional.

6.2 Trabalhos Futuros

Como trabalhos futuros pode ser citado a expansão da base imagens para abranger maior quantidade de classes para a classificação com e sem estados. Além disso, pode ser estudado o impacto de outros modelos de CNN para fins de comparação com os utilizados no trabalho.

Outra questão que pode ser abordada é portabilidade do sistema, pensando em formas de se utilizar o que já foi desenvolvido e quais seriam as alterações necessárias, avaliando o que seria preciso para ser utilizado em outros dispositivos. Essas alterações visando o uso do dispositivo com o sistema em ambientes internos e possíveis expansões para locais externos desconhecidos ao usuário.

REFERÊNCIAS

ACADEMY, D. S. *Deep Learning Book*. 2019. Http://www.deeplearningbook.com.br/. Access date: 15 July. 2019.

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. In: IEEE. *Engineering and Technology (ICET), 2017 International Conference on*. [S.l.], 2017. p. 1–6.

ALOM, M. Z. et al. The history began from alexnet: a comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

BAI, J. et al. Smart guiding glasses for visually impaired people in indoor environment. *IEEE Transactions on Consumer Electronics*, IEEE, v. 63, n. 3, p. 258–266, 2017.

BASHIRI, F. S. et al. Object detection to assist visually impaired people: A deep neural network adventure. In: SPRINGER. *International Symposium on Visual Computing*. [S.l.], 2018. p. 500–510.

BOER, P.-T. D. et al. A tutorial on the cross-entropy method. *Annals of operations research*, Springer, v. 134, n. 1, p. 19–67, 2005.

CENGIL, E.; ÇINAR, A.; ÖZBAY, E. Image classification with caffe deep learning framework. In: IEEE. *Computer Science and Engineering (UBMK), 2017 International Conference on*. [S.l.], 2017. p. 440–444.

CHUANG, T.-K. et al. Deep trail-following robotic guide dog in pedestrian environments for people who are blind and visually impaired-learning from virtual and real worlds. In: IEEE. *2018 IEEE International Conference on Robotics and Automation (ICRA)*. [S.l.], 2018. p. 1–7.

CONTRERAS, S.; ROSA, F. D. L. Using deep learning for exploration and recognition of objects based on images. In: IEEE. *Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR), 2016 XIII Latin American*. [S.l.], 2016. p. 1–6.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine learning*, Springer, v. 20, n. 3, p. 273–297, 1995.

DENG, F.; ZHU, X.; REN, J. Object detection on panoramic images based on deep learning. In: IEEE. *Control, Automation and Robotics (ICCAR), 2017 3rd International Conference on*. [S.l.], 2017. p. 375–380.

DING, X. et al. Indoor object recognition using pre-trained convolutional neural network. In: IEEE. *Automation and Computing (ICAC), 2017 23rd International Conference on*. [S.l.], 2017. p. 1–6.

DOURADO, A. M. B.; PEDRINO, E. C. Embedded navigation and classification system for assisting visually impaired people. In: *VISIGRAPP*. [S.l.: s.n.], 2018. p. 516–523.

EITEL, A. et al. Multimodal deep learning for robust rgb-d object recognition. In: IEEE. *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. [S.l.], 2015. p. 681–687.

FILHO, O. M.; NETO, H. V. *Processamento digital de imagens*. [S.l.]: Brasport, 1999.

FILIPE, V. et al. Assisted guidance for the blind using the kinect device. In: ACM. *Proceedings of the 7th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion*. [S.l.], 2016. p. 13–19.

FONNEGRA, R. D.; BLAIR, B.; DÍAZ, G. M. Performance comparison of deep learning frameworks in image classification problems using convolutional and recurrent networks. In: IEEE. *Communications and Computing (COLCOM), 2017 IEEE Colombian Conference on*. [S.l.], 2017. p. 1–6.

GONZALEZ, R.; WOODS, R. Processamento de imagens digitais, tradução do original digital image processing. *Edgard Blucher, São Paulo*, 2000.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. Www.deeplearningbook.org.

HU, Z. et al. Facial age estimation with age difference. *IEEE Transactions on Image Processing*, IEEE, v. 26, n. 7, p. 3087–3097, 2017.

HUI-BIN, L. et al. Recognition of individual object in focus people group based on deep learning. In: IEEE. *Audio, Language and Image Processing (ICALIP), 2016 International Conference on*. [S.l.], 2016. p. 615–619.

ICHINOSE, A. et al. Evaluation of distributed processing of caffe framework using poor performance device. In: IEEE. *Big Data (Big Data), 2016 IEEE International Conference on*. [S.l.], 2016. p. 3980–3982.

JABEEN, F.; MUHAMMAD, A.; ENRÍQUEZ, A. M. M. Feed forward neural network training based interactive shopping for blind. In: *CCE*. [S.l.: s.n.], 2015. p. 1–6.

KEERTHI, S. et al. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, v. 13, n. 3, p. 637–649, 2001.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2012. p. 1097–1105.

KUNENE, D.; VADAPALLI, H.; CRONJE, J. Indoor sign recognition for the blind. In: ACM. *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*. [S.l.], 2016. p. 19.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LI, Y.; SONG, Y.; LUO, J. Improving pairwise ranking for multi-label image classification. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2017.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1, n. 14, p. 281–297.

MONTEIRO, J. et al. Virtual guide dog: An application to support visually-impaired people through deep convolutional neural networks. In: IEEE. *Neural Networks (IJCNN), 2017 International Joint Conference on*. [S.l.], 2017. p. 2267–2274.

NVIDIA. *NVIDIA DIGITS*. 2016. Https://developer.nvidia.com/digits. Access date: 9 april. 2019.

NVIDIA. *Embedded Systems*. 2017. Https://www.nvidia.com.br/object/embedded-systems-br.html. Access date: 9 april. 2019.

PRAJAPATI, N.; NANDANWAR, A. K.; PRAJAPATI, G. Edge histogram descriptor, geometric moment and sobel edge detector combined features based object recognition and retrieval system. *International Journal of Computer Science and Information Technologies 0975–9646*, 2016.

QUINLAN, J. R. Induction of decision trees. *Machine learning*, Springer, v. 1, n. 1, p. 81–106, 1986.

SHAILAJA, K.; ANURADHA, B. Effective face recognition using deep learning based linear discriminant classification. In: IEEE. *Computational Intelligence and Computing Research (ICCIC), 2016 IEEE International Conference on*. [S.l.], 2016. p. 1–6.

SHIM, V. A.; YUAN, M.; TAN, B. H. Automatic object searching by a mobile robot with single rgb-d camera. In: IEEE. *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2017*. [S.l.], 2017. p. 056–062.

SUGOMORI, Y. et al. *Deep Learning: Practical Neural Networks with Java*. [S.l.]: Packt Publishing Ltd, 2017.

SZEGEDY, C. et al. Going deeper with convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 1–9.

UÇAR, A.; DEMIR, Y.; GÜZELIŞ, C. Moving towards in object recognition with deep learning for autonomous driving applications. In: IEEE. *INnovations in Intelligent SysTems and Applications (INISTA), 2016 International Symposium on*. [S.l.], 2016. p. 1–5.

VELLA, F. et al. Classification of indoor actions through deep neural networks. In: IEEE. *Signal-Image Technology & Internet-Based Systems (SITIS), 2016 12th International Conference on*. [S.l.], 2016. p. 82–87.

WEKA. *WEKA SMO*. 2016. Http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/SMO.html. Access date: 9 july. 2019.

YANG, X. et al. Context-based indoor object detection as an aid to blind persons accessing unfamiliar environments. In: ACM. *Proceedings of the 18th ACM international conference on Multimedia*. [S.l.], 2010. p. 1087–1090.

YANG, X.; YUAN, S.; TIAN, Y. Recognizing clothes patterns for blind people by confidence margin based feature combination. In: ACM. *Proceedings of the 19th ACM international conference on Multimedia*. [S.l.], 2011. p. 1097–1100.

ZHANG, H. et al. Pedestrian detection method based on faster r-cnn. In: IEEE. *Computational Intelligence and Security (CIS), 2017 13th International Conference on*. [S.l.], 2017. p. 427–430.

ZHANG, L. et al. Robust visual tracking using oblique random forests. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2017. p. 5589–5598.

ZHANG, Y. et al. Physically-based rendering for indoor scene understanding using convolutional neural networks. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. [S.l.], 2017. p. 5057–5065.

ZHU, P. et al. Deep learning feature extraction for target recognition and classification in underwater sonar images. In: IEEE. *Decision and Control (CDC), 2017 IEEE 56th Annual Conference on*. [S.l.], 2017. p. 2724–2731.

ZOCCA, V. et al. *Python Deep Learning*. [S.l.]: Packt Publishing Ltd, 2017.

GLOSSÁRIO

CNN – *Convolutional Neural Network*

DBM – *Deep Boltzman Machine*

LSTM – *Long Short Time Memory*

RBM – *Restricted Boltzmann Machine*

RNN – *Recurrent Neural Network*

SVM – *Support Vector Machine*