GISELE FRANCO DA SILVA

TECNOLOGIAS ASSISTIVA E A DEEP LEARNING: Aplicativo Com Reconhecimento De Imagem No Auxílio A Deficientes Visuais

Ji-Paraná, RO

2019

GISELE FRANCO DA SILVA

TECNOLOGIAS ASSISTIVA E A DEEP LEARNING: Aplicativo Com Reconhecimento De Imagem No Auxílio A Deficientes Visuais

Monografia apresentada à Banca Examinadora do Centro Universitário São Lucas Ji-Paraná, como requisito de aprovação para obtenção do Título de Bacharel em Sistemas de Informação.

Orientador: Prof. Me. Thyago Bohrer Borges.

Ji-Paraná, RO

2019

Dados Internacionais de Catalogação na Publicação
Gerada automaticamente mediante informações fornecidas pelo(a) autor(a)

S586t Silva, Gisele Franco

Tecnologia assistiva e a Deep Learning: aplicativo com reconhecimento de imagem no auxílio a deficientes visuais / Gisele Franco da Silva . -- Ji-Paraná, RO, 2019.

62 p.

Orientador(a): Prof. Me Thyago Bohrer Borges

Trabalho de Conclusão de Curso (Graduação em Sistemas de Informação) - Centro Universitário São Lucas

1. Tecnologia Assistiva. 2. Deficientes visuais. 3. Inteligência Artificial. I. Borges, Thyago Bohrer. II. Título.

CDU 004.4

Bibliotecário(a) Alex Almeida CRB 11.853

SÃO LUCAS
EDUCACIONAL

ATA Nº 05/2019 DE TRABALHO DE CONCLUSÃO DE CURSO EM SISTEMAS DE INFORMAÇÃO

No segundo dia do mês de dezembro de 2019, das 17h as 22h reuniram-se na sala de Inovação Tecnológica 7 o(a) professor(a) orientador(a) Thyago Bohrer Borges e os(as) professores(as) José Rodolfo Milzzotto Olivas e Willian Alves de Oliveira Fachetti para comporem Banca Examinadora de Trabalho de Conclusão de Curso em Sistemas de Informação sob presidência do(a) primeiro(a), para analisarem a apresentação do trabalho "Tecnologias Assistivas e Deep Learning: Aplicativo com reconhecimento de imagem no auxílio a deficientes visuais". Após as arguições e apreciação sore o trabalho exposto foi atribuída à menção como nota do Trabalho e Concluso do curso do Acadêmico(a) Gisele Franco da Silva.

OBS: Trabalho de Conclusão de Curso (X) Aprovado ou (__) Reprovado com nota total de 9,4, atribuídos o valor de 9,4 (nove virgula quatro pontos) para o trabalho escrito e o valor de 9,5 (nove virgula cinco pontos) para a apresentação oral.

Gisele Franco da Silva

Prof. Esp. Willian A. de Olveira Fachetti

Prof. Esp. José Rodolfo M. Olivas

Prof. Prof. Me. Thyago Bohrer Borges
Orientador

Prof. Me. Thyago Bohrer Borges
Coord. Sistemas de Informação

Á minha família, em especial a minha querida prima Islanda Carine por sua garra em superar as suas limitações.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por ser o melhor amigo, protetor, conselheiro e fonte inesgotável de força e alegria.

Aos meus pais, Maria Nilza e Elio Alves, que nunca mediram esforços para me proporcionar o melhor e por todo o amor e carinho que sempre recebi.

Ao meu irmão, Dyego Franco, e ao meu namorado Valdeir Cesconetto por sempre acreditar, incentivar e nunca me deixar desistir.

Aos meus amigos, todos os que contribuíram com o apoio e conhecimento que foram fundamentais para que esse momento se concretizasse, em especial, ao grupo #ETGG e ao Hiago Luan.

Ao meu orientador, Thyago Borges, por me motivar, orientar e disponibilizar o seu tempo para me guiar no desenvolvimento deste trabalho e todos os professores que compartilharam seus conhecimentos comigo ao longo dessa caminhada.

"Inteligência é a capacidade de se adaptar à mudança."

(Stephen Hawking)

RESUMO

A tecnologia caminha no sentido de tornar a vida mais simples e prática, e com o advento da inteligência artificial, o mais inteligente e humana possível. O processo de ensinar uma máquina o modelo de raciocínio humano é chamado de *Machine Learning* ou aprendizado de máquina, que apesar da crescente expansão e diversas aplicações, ainda é subutilizado como uma importante técnica de tecnologia assistiva, principalmente para pessoas com deficiência visual, cujo um dos maiores desafios é a percepção de imagens estáticas. Neste contexto, o objetivo geral deste trabalho é comprovar a aplicabilidade prática das técnicas de *Machine Learning*, mais precisamente *Deep Learning*, no desenvolvimento de tecnologias assistivas e de baixo custo. A metodologia utilizada para levantamento dos conceitos e ferramentas disponíveis atualmente foi à pesquisa bibliográfica, além da modelagem e prototipação de um aplicativo móvel com reconhecimento de imagem chamado Vedere que utiliza bibliotecas e base de dados *open source*, desenvolvimento ágil, bem como redes neurais artificiais para detecção de objetos.

Palavras-chave: Tecnologia Assistiva. Reconhecimento de imagem. *Machine Learning. Deep Larning*.

ABSTRACT

Technology moves towards making life simpler and more practical, and with the advent of artificial intelligence, the smartest and most humane possible. The process of teaching a machine the human reasoning model is called Machine Learning, that despite the growing expansion and various applications, it is still underutilized as an important technique of assistive technology, especially for visually impaired people, whose one of the biggest challenges is the perception of static images. In this context, the general objective of this work is to prove the practical applicability of Machine Learning techniques, more precisely Deep Learning, in the development of assistive and low-cost technologies. The methodology used to survey the concepts and tools currently available was the bibliographic research, In addition to modeling and prototyping an image-aware mobile application called Vedere that uses libraries and open source database, agile development as well as artificial neural networks for object detection.

Keywords: Assistive technology. Image recognition. Machine Learning. Deep Learning.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

IBGE	Instituto Brasileiro de Geografia e Estatística
TA	Tecnologias Assistivas
ELA	Esclerose Lateral Amiotrófica
ML	Machine Learning
CGEE	Centro de Gestão e Estudos Estratégicos
CAT	Comité de Ajuda Técnicas
ISO	International Organization for Standardization
DV	Deficiência Visual
MEC	Ministério da Educação
AM	Aprendizado de Máquina
IA	Inteligência Artificial
SBC	Sociedade Brasileira de Computação
AP	Aprendizado Profundo
STF	Supremo Tribunal Federal
RGB	Red, Green, Blue
HSV	Hue, Saturation, Value
RNA	Redes Neurais Artificiais
RNC	Redes Neurais Convolucionais
OpenCV	Open Source Computer Vision
API	Application Programming Interface
CNTK	Microsoft Cognitive Toolkit
Yolo	You Only Look Once
GPU	Graphics Processing Unit
CPU	Central Process Unit
JSON	JavaScript Object Notation
IDE	Integrated Development Environment
UML	Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

O avanço da tecnologia segue no caminho de tornar a vida menos complexa, seja no simples *smartphone* com funções que pertenciam somente a grandes computadores, em relógios que não mostram apenas horas ou ainda nas chamadas casas inteligentes que tornam o dia a dia das pessoas mais descomplicado e autônomo. Conforme relatório anual divulgado pelo Banco Mundial ainda no ano de 2016 o avanço tecnológico já detinha o poder de possibilitar a participação social e econômica além da inclusão de milhões de pessoas, principalmente as com alguma espécie de deficiência.

Segundo o último censo realizado pelo Instituto Brasileiro de Geografia e Estatística IBGE (2010) em torno de 23,9% da população no Brasil (45,6 milhões) possui algum tipo de deficiência, deformação ou insuficiência de uma função física ou mental, sendo aproximadamente 35,8 milhões com alguma dificuldade para enxergar e também para se integrar ao moderno mundo digital e até mesmo realizar atividades rotineiras.

Se para pessoas sem nenhum tipo de deficiência a tecnologia simplifica, para aqueles que a possuem, essa é talvez a única forma de realizar atividades funcionais e participar da crescente era da informação. É nesse contexto que surge as Tecnologias Assistivas (TA) classificação de todo o arsenal de serviços e ferramentas que promovem habilidades funcionais, independência e inclusão.

1.1 PROBLEMATIZAÇÃO

Para as pessoas com baixa ou nenhuma visão o maior empecilho é o reconhecimento de objetos estáticos, tendo a dependência de outros para realização de diversas atividades, que vão desde uma simples compra no supermercado até se locomover livremente em ambientes desconhecidos. Com o advento do sistema Braille, sistema de escrita e leitura de pontos em relevo, o acesso à alfabetização, conhecimento, cultura e lazer passou a ser possível, mas extremamente limitada à adoção desse mecanismo por parte dos fabricantes dos produtos e serviços. Uma vez que mesmo sendo amplamente conhecido, ainda não há nenhuma

regulamentação que padronize a forma com que os impressos em Braille deverão ser inseridos nos objetos, e dependendo da estrutura ou textura esse escrito fica incompreensível ou passível de ser danificado com o tempo.

Diante dessa necessidade de maior precisão no reconhecimento de imagem, é que a inteligência artificial, tecnologia capaz de simular o raciocínio humano, e mais precisamente do *Machine Learning,* em português aprendizado de máquina, surge como uma poderosa arma nesse processo de inclusão. No Catálogo Nacional de Produtos de Tecnologia Assistiva nos mais de 1.200 itens distribuídos, menos de um terço é voltado à deficiência visual, sendo menor ainda a porcentagem dos que utilizam *Deep Learning* para visão computacional.

1.2 OBJETIVOS

1.2.1 Objetivo geral

Observando o contexto acima descrito, o objetivo geral do presente trabalho é comprovar a eficácia da aplicação de técnicas de aprendizado de máquina, no caso o reconhecimento de imagem, como uma poderosa tecnologia assistiva na inclusão de pessoas com deficiência visual, não somente na sociedade como também no mundo tecnológico.

1.2.2 Objetivos Específicos

- Apresentar a importância do investimento em tecnologias assistiva na vida de pessoas com deficiência visual.
- Apresentar as principais características e aplicações da técnica de reconhecimento de imagem (Visão Computacional).
- Mostrar a aplicabilidade prática da *Deep Learning* por meio da modelagem e prototipação de um aplicativo móvel para reconhecimento de objetos.
- Utilizar um sistema de detecção de objetos em tempo real e público para treinar o algoritmo de reconhecimento de objetos.

1.3 DELIMITAÇÃO DO ESTUDO

O foco desta pesquisa está na implementação da *Deep Learning* nas tecnologias assistivas. A deficiência visual foi escolhida devido ao extremo impacto

nas condições de vida de quem a possui, e ainda assim possuir poucas ferramentas de auxílio, que podem ser de fato efetiva nas atividades do dia a dia. Com intuito de promover maior autonomia e locomoção foi adotado um aplicativo para dispositivos móveis, já presente na vida cotidiana das pessoas com deficiência visual graças aos recursos de leitor de tela. No sistema operacional Android o leitor de tela é chamado de *TalkBack* e no IOS é conhecido como *VoiceOver,* ambos com finalidade de identificar todos os itens presentes no local pressionado da tela. Neste estudo o aplicativo será voltado à detecção e identificação de os objetos estáticos, a fim de situar e dar certa independência para seus usuários. A detecção dos objetos é delimitada a base de dados do sistema *TensorFlow*, escolhido por ser público e de simples adaptação.

1.4 RELEVÂNCIA DO ESTUDO

Stephen William Hawking, renomado astrofísico inglês e um dos maiores cosmólogo da história, foi um grande exemplo de superação ao conquistar esse status mesmo convivendo com esclerose lateral amiotrófica (ELA). A utilização de elevada tecnologia, como um sintetizador de voz, junto a sua força de vontade permitiu com que continuasse transmitindo ao mundo todo o seu conhecimento. Exemplo que mostra a importância da inclusão de todas as pessoas nos mais variados ambientes educacionais, esportivos, lazer ou sociais, assim como o essencial investimento em tecnologias capazes de devolver, ao menos parcialmente, aquilo que a deficiência impossibilita ou dificulta (Britânica Escola, 2019).

Desse modo, o presente estudo se torna extremamente relevante ao mostrar a relação entre, uma tecnologia de grande expansão no mercado, a *Machine Learning* (ML) com a automação de habilidades funcionais, de suma importância para pessoas com deficiência visual.

O tempo de treinamento do algoritmo está diretamente ligado com a porcentagem de acertos que o mesmo pode obter, por ser semelhante ao processo de aprendizagem humana, a máquina inicialmente pode equivocar-se devido à semelhança entre as imagens, contudo, é passível de ser superada a cada nova interação.

Com o resultado do estudo é possível reproduzir o desenvolvimento de um aplicativo de reconhecimento de objetos, funcional e que pode ser aplicado e

treinado para tornar-se cada vez mais efetivo no auxílio a pessoas com deficiência visual.

2 REFERENCIAL TEÓRICO

O referencial teórico da presente pesquisa foi estruturado em cinco capítulos principais, sendo: A Tecnologia Assistiva e a Deficiência visual, Inteligência Artificial e o Aprendizado de Máquina, Reconhecimento de Imagem, Redes neurais Convolucionais Profundas e Processamento de Imagem na Nuvem.

2.1 TECNOLOGIA ASSISTIVA E A DEFICIÊNCIA VISUAL

A palavra assistiva significa algo que é acompanhado ao longo de um processo, de forma que tecnologia assistiva é o nome dado a todo o arsenal de equipamentos ou serviços que proporcionam ou ampliam as habilidades funcionais de pessoas com deficiência ao longo da vida e, por conseguinte promovem a inclusão e autonomia (CGEE, 2012). De modo que a principal finalidade dessa tecnologia é suprir as lacunas entre o que a pessoa precisa fazer e o que a deficiência lhe impede ou dificulta.

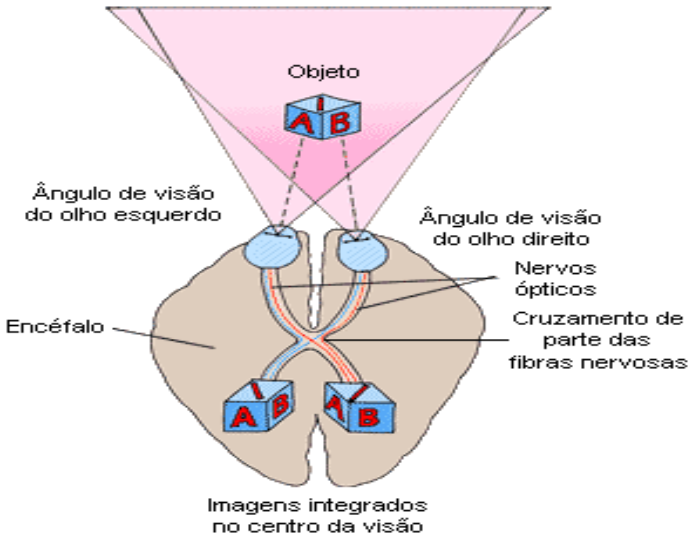
No Brasil foi instituído um Comité de Ajuda Técnicas (CAT) por meio da portaria n°142 em 16 de novembro de 2016 para dar transparência e legalidade no desenvolvimento de tecnologia assistivas, abrangendo tanto tarefas básicas do dia a dia quanto às atividades profissionais, promovendo parcerias entre órgãos públicos e sociedade civil, estímulos a cursos, pesquisas e propostas de soluções governamentais (BRASIL, 2009).

A categorização dos tipos de tecnologia assistiva, segundo CGEE (2012), ocorre em função da sua finalidade conforme ISO 9999:2007, sendo as mais conhecidas: Comunicação aumentativa e alternativa – destinadas as pessoas com dificuldades na fala ou na escrita, sistemas de controle de ambiente – voltada a limitação motora utilizando controles remotos, auxílios de mobilidade – qualquer equipamento utilizado para melhoria da mobilidade individual, órtese e prótese – peças artificias que substituem partes do corpo ausentes, recursos de acessibilidade ao computador – permite dar acessibilidade a pessoas com privações sensórias sejam visuais, motoras, auditivas ou intelectuais, auxílios para vida diária e prática e ainda para cegos ou para pessoas com visão subnormal, sendo esse último, o objeto de estudo do presente artigo.

Conforme o Ministério da Saúde (2008) a deficiência visual (DV) é oriunda de uma limitação sensorial no órgão responsável pela visão, sendo dividida entre cegueira total ou baixa visão. Cegueira é o nome dado ao processo de perda total da visão e percepção de luz, possuindo uma acuidade menor ou igual a 0,3 no olho com melhor visão, classificada como congênita quando já vem desde o nascimento, e a adquirida quando se perde com o tempo. Ainda segundo o Ministério da Educação - MEC em GIL (2000) a cegueira adquirida pode possibilitar memórias visuais das cores e luzes de imagens vistas antes de se perder a visão, sendo extremamente útil no processo de readaptação. Já a chamada baixa visão é decorrente de inúmeros fatores isolados ou não, mas que de alguma forma, alteram a capacidade funcional da visão, causando redução no campo visual, alterações corticais ou ainda sensibilidades. De acordo com Fernandes (2012) nesse meio termo existem ainda conhecidas patologias tais como o astigmatismo, miopia e estrabismo entre outras, que apesar de não serem classificadas como deficiência visual, atrapalham o processo normal de visão principalmente se não tratadas adequadamente.

No processo de visão do olho humano existem as estruturas responsáveis pela captação da luz com função ótica e os transformadores de impulso luminoso em elétrico por meio de reação química (RAMOS, 2019). Cada um dos olhos forma uma imagem separada, sendo o cérebro o responsável por reuni-las formando o objeto final, conforme Figura 01.

Figura 1 – Processo de visão humana



Fonte: EducaBras, 2019

A perda de visão seja total ou parcial interfere no desenvolvimento individual, psicológico e emocional além de limitar rotinas básicas diárias, profissionais e até mesmo de locomoção ao depender constantemente de acompanhamento de demais pessoas, já que a visão constitui um dos canais mais importante ao se relacionar com o mundo externo (GIL, 2000).

Um estudo composto por Dias e Pereira (2008) mostra que a audição das pessoas com deficiência visual tende a ser mais aguçadas quando comparado com pessoas que pode enxergar, devido à atenção direcionada a audição. No estudo também foi observado o fenômeno denominado plasticidade de modalidade cruzada, onde pode ocorrer uma reorganização de propriedades morfológicas e funcionais em níveis de processamento de informação no modulo sensorial auditivo. Portanto, dispositivos com saídas auditivas se tornam instrumentos viáveis e práticos.

Entre as ferramentas disponíveis para o auxílio ao DV a mais conhecida e fundamental é o sistema Braille, um mecanismo de escrita e leitura em alto relevo com alfabeto caracterizado por seis pontos, totalizando 63 combinações que representam desde músicas, pontuações, letras simples entre outras, sendo o criador francês, Louis Braille (JÚNIOR, 2009). Com avanço da tecnologia veio às impressoras em Braille, leitores de telas, óculos estimuladores, teclados em Braille, relógio inteligente entre outros.

De acordo com o Ministério da Educação (GIL, 2000), diferente do que muitos acreditam a função de enxergar não é ingênita, é desenvolvida, por meio de um processo inconsciente, à medida que é ensinado ao recém-nascido o que é uma mamadeira, um carro ou qualquer outro objeto e até mesmo uma pessoa. Cada ser humano tem a capacidade de desenvolver seu próprio guia para se orientar no ambiente em geral. A orientação se dá marcando-se os pontos referenciais como, por exemplo: ao aprender um endereço ou ao diferenciar objetos por suas características físicas, como cor, textura e forma. Com advento da Inteligência Artificial, mais especificamente o Aprendizado de Máquina (Seção 2.2) estas características podem ser replicados e ensinados a uma máquina na finalidade de reconstruir o sistema que foi danificado pela deficiência visual.

2.2 INTELIGÊNCIA ARTIFICIAL E O APRENDIZADO DE MÁQUINA

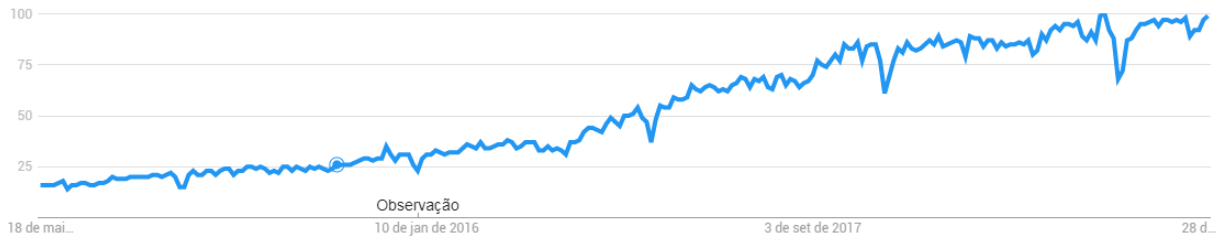
A palavra inteligência significa ato de aprender, compreender e adaptar-se, mas quando aplicado ao campo artificial, não existe um conceito único e universal sendo normalmente classificado em quatro ramos principais: sistemas que pensam como humanos; sistemas que atuam como seres humanos; sistemas que pensam racionalmente e sistemas que atuam racionalmente.

Ao longo dos anos a IA foi evoluindo a ponto de ser aplicada nas mais diversas categorias, desde mecânica automotiva, medicina, processamento de linguagem, automação e reconhecimento de imagem. De modo a ser desenvolvidas técnicas como *Machile Learning, Deep Learning* ou analise preditiva.

Aprendizado de Máquina (AM) ou *Machine Learning* é uma área da ciência da computação, que compõem a esfera da Inteligência Artificial (IA), voltada ao estudo de métodos de análise de dados e construção de modelos analíticos, entre outras palavras, baseia-se na ideia que sistemas podem não apenas consultar como também aprender com os dados, formando padrões e tomando decisões com quantidade mínima de intervenção do ser humano (IBM, 2018).

Essa área de pesquisa tem crescido exponencialmente tanto em avanço tecnológico quanto em interesse por parte da população, conforme gráfico de interesse do Google *Trends* na Figura 02.

Figura 2 - Frequência pesquisa do termo “*Machine Learning*” nos últimos cinco anos.



Fonte: Google Trends, 2019.

Grandes empresas investiram pesado na IA, expondo hoje tecnologias como carro autônomo do Google, o Tesla, *Smartphones* com reconhecimento facial, ou ainda os sistemas de recomendações da Amazon e do Netflix, entre outros exemplos nas mais variadas atividades.

O processo de aprendizado é semelhante ao humano, onde uma base de dados é passada e posteriormente ao se deparar com uma nova informação, a mesma é classificada e relacionada ao que já é conhecido. Os algoritmos de ML utilizam de processos matemáticos e computacionais para o mapeamento das probabilidades, o que permite aprendizagem de elementos com abstração cada vez mais elevada (SBC, 2019).

Segundo Carvalho (2014) existe diferentes métodos de aprendizado, como o aprendizado supervisionado, utilizado nesse projeto, onde existe a entrada de uma informação com sua saída correspondente para que o algoritmo consiga compreender a relação e a utilize nas próximas entradas informadas. Já no aprendizado não supervisionado, o algoritmo não possui dados iniciais com os quais se basear, deve aprender a relacionar as informações passadas de modo autônomo. Existe ainda outro processo de aprendizado, chamado de aprendizado por reforço, que mescla aspectos desses dois métodos, sendo utilizado quando a rotulação dos dados é custosa e ensina qual ação priorizar dada determinada situação.

Na 39° edição da revista Sociedade Brasileira de Computação - SBC (2019) é evidenciado ainda um subgrupo específico de ML que necessitam de alta quantidade de dados e treinamento bem como algoritmos complexos, chamado de *Deep Learning* ou em português Aprendizado Profundo (AP). Essa técnica permite a construção de redes neurais artificiais com aproximadamente 10 milhões de neurônios, reconhecimento de imagens, traços e demais aspectos detalhistas, no entanto está extremamente relacionada à capacidade de processamento.

A revista destaca ainda um grande exemplo de aplicação das técnicas de aprendizado de máquina no mundo do direito, o projeto Victor, que foi resultado de uma parceria entre o Supremo Tribunal Federal (STF) e a Universidade de Brasília, mais uma prova de que o *Machine Learning* é ideal para a automação desde processos simples, como reconhecimento de padrões, até complexos como as tomadas de decisões feitas normalmente por especialistas.

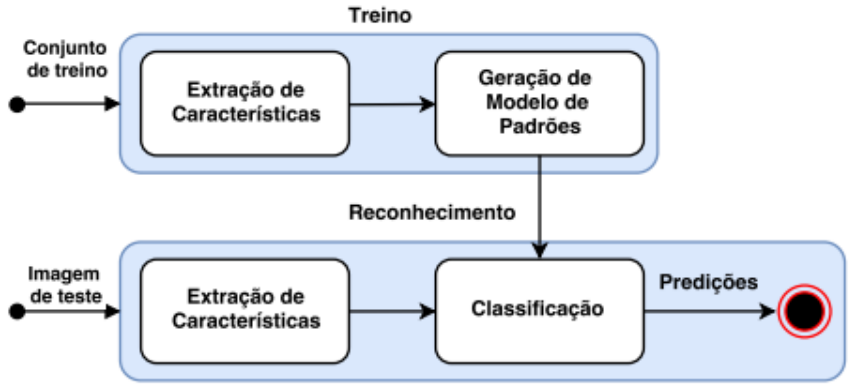
2.3 RECONHECIMENTO DE IMAGEM

A *Deep Learning* é aplicada em diversos campos, mas é na visão computacional que possui maior destaque, sendo essa área responsável por programar a capacidade “ver” a máquinas (JOKELA, 2018). Nesse aspecto, para o

tratamento e classificação de uma imagem no meio digital é preciso extrair e identificar todos os atributos nela contidos, como cor, traços, iluminação, fundos, sombras e tamanhos, além da separação de todos os demais itens presente na mesma. Essa categorização apesar de simples para o ser humano é extremamente complexa para a máquina, uma vez que diferentes objetos podem pertencer a uma mesma categoria ou ainda podem estar em diferentes ângulos, cores e fundos e qualquer outro tipo de mudança ou filtro que hoje é possível aplicar a uma imagem (PRASAD, 2010).

Segundo Lucena, Veloso e Lopes (2016) esse processo ocorre em duas etapas, sendo primeiro a obtenção das características e catalogalização com treinamento e reconhecimento. Sendo o treinamento focado na busca de um padrão nas coleções de imagens testes, e o reconhecimento na correlação entre os padrões já estabelecidos com a nova imagem informada, como exemplificada na Figura 03.

Figura 3 - Algoritmo de reconhecimento de objetos



Fonte: Lucena, Veloso e Lopes, 2016.

O processo de captura realizado por uma câmera digital é estruturada bidimensionalmente, sendo constituída por vários sensores sensíveis a luz e capazes de captar energia luminosa, agrupando as imagens em duas categorias, a vetorial ou *bitmap*. Sendo *bitmap* uma matriz bidimensional constituída de *pixels*, valores numéricos que representam a cor, que geralmente adquire o aspecto distorcido quando ampliado. Já nas imagens do tipo vetoriais a qualidade se mantém mesmo quando ampliada, devido a sua composição com formas geométricas e equações matemáticas. Com relação às cores também ocorre variação, sendo as mais utilizadas o RGB (*Red, Green, Blue*) composto pelo vermelho, verde e azul e

formando por meio de adição de intensidades a alguma dessas três cores, e o HSV (*Hue, Saturation, Value*) que recebe um valor para determinar a matiz da cor, outro valor para saturação e também um para nível de brilho. (AZEVEDO, 2016).

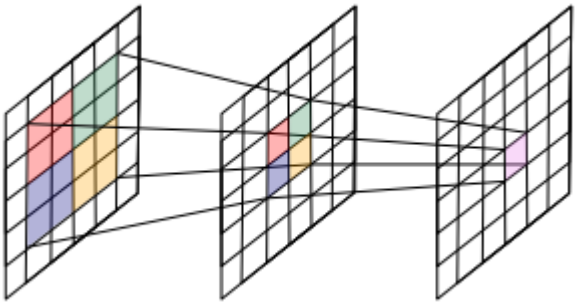
As etapas básicas do processamento de imagem conforme Bento (2016) são a aquisição de imagem, onde a imagem é capturada e convertida ao modelo de cor ou tipo especificado, no pré-processamento ocorre à correção e suavização de brilhos, contrastes ou ruídos. Segmentação, momento de identificação e separação das partes importantes da imagem, seguido da extração das características e enumeração descritiva. Nessa etapa o reconhecimento vai rotular os objetos da imagem com base nessa descrição e na interpretação atribuir significado para assim ser armazenado na base de conhecimento.

2.4 REDES NEURAIS CONVOLUCIONAIS PROFUNDAS

Segundo Miyazaki (2017) Redes Neurais Artificiais (RNA) são estruturas computacionais de aprendizado de máquina baseadas no sistema biológico, cujos elementos básicos são os neurônios, sendo amplamente utilizadas para reconhecimento de objetos em imagens. Um neurônio artificial é comumente expresso por uma função, sendo o modelo mais básico conhecido de uma rede, o *Perceptron*, capaz de classificar padrões agrupados, mesmo que, de modo linear. Com o aprendizado profundo, *Deep Learning,* houve um considerável crescimento de parâmetros a serem utilizados nos algoritmos de aprendizados, dando origem as chamadas Redes Neurais Convolucionais (RNC).

Esse modelo destaca-se na área de processamentos de imagens e detecção de objetos, uma vez que lida com dados em estruturas locais, assim como os *pixels*, e possui uma arquitetura semelhante ao córtex visual, onde cada neurônio é responsável por um processo de visão e no final se juntam e toda a imagem é tratada. A profundidade dessa saída é diretamente proporcional à quantidade de filtros aplicados, sendo esse formado por pesos definidos previamente e atualizados a cada nova entrada durante o processo de treinamento. Os filtros possuem parâmetros, que podem vir já determinados, como preenchimento da imagem e a forma como a mesma deve ser percorrida.

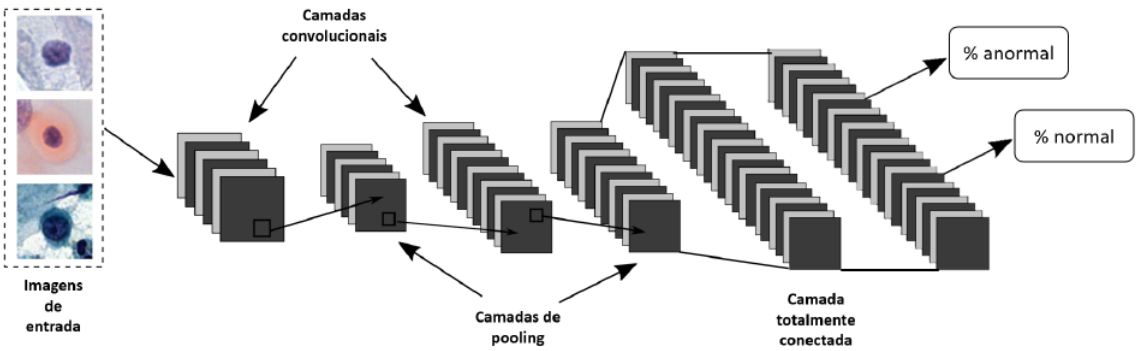
Figura 4 - Campo de visão do neurônio ao longo das camadas.



Fonte: Kovaleski, 2018.

Conforme exemplificado na Figura 04, o nome convolucional é devido aos filtros aplicados em pequenos grupos de dados vizinhos em buscas das mais relevantes características, extraindo e repassando somente essas as demais camadas, permitindo a identificação de padrões mais complexos quando combinados com os já identificados padrões, aumentando a visão do neurônio de acordo com a profundidade na rede (KOVALESKI, 2018). Os neurônios não comunicam com todos os da camada posterior, mas utilizam da estrutura tridimensional para analisar uma região, filtrando as próximas conexões, as chamadas *pooling* proporcionam menos dimensionalidade e a rede totalmente conectada deixam o processo de treinamento mais efetivo, como mostra a Figura 5.

Figura 5 – Arquitetura da rede neural convolucional (RNC).



Fonte: Araújo, Carneiro e Silva (2017).

Os resultados são geralmente otimizados de acordo com as amostras previamente classificadas, sendo que quanto maior a quantidade de filtros melhor a profundidade, mas também aumenta a complexidade, consumo de memoria e tempo.

2.5 PROCESSAMENTO DE IMAGEM NA NUVEM

É chamado de computação em nuvem todo o fornecimento de tecnologias como servidores, armazenamento, banco de dados, rede, analise e inteligência, software entre outros pela internet (AZURE, 2019). Esse modelo de serviço tem ganhado espaço devido a reduzir os custos operacionais de se construir um próprio ambiente com toda a infraestrutura necessária, proporcionar amparo computacional escalonar sempre e onde for necessária, a segurança e o controle de acesso a informação também é mais robusta, os serviços são comumente executados em *datacenters* seguros e atualizados. Com serviço sob demanda uma grande quantidade de recursos pode ser disponibilizada em minutos e regulada conforme as necessidades do contratante.

Há três diferentes tipos de implantação de nuvem: privada, hibrida e publica. O primeiro é ocorre quando a infraestrutura da nuvem é exclusivamente utilizada por uma empresa, sendo administrado por ela ou por terceiros. É característica marcante desse modelo à implantação de controles de acessos, seja no gerenciamento de redes, autenticações, provedores de serviços e autorizações (SOUSA, MOREIRA, MACHADO, 2010).

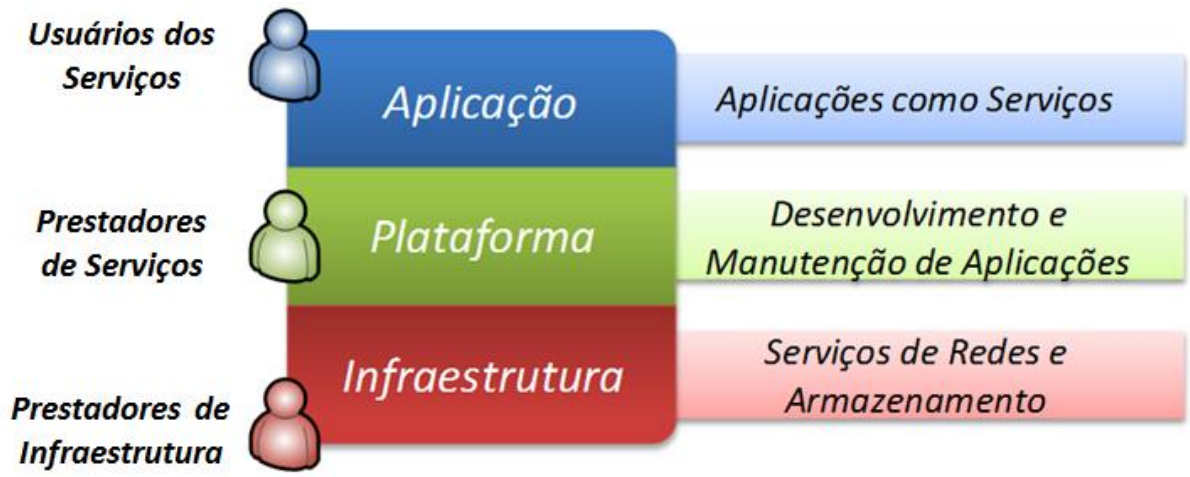
O modelo de nuvem publica, como o nome sugere, é aberto ao publico em geral, podendo ser acessado por qualquer que tenha conhecimento da sua localização, ao contrario do modelo privado não ocorre politicas de controle de acesso e restrição no gerenciamento de redes, sendo esse modelo o adotado neste trabalho por meio da tecnologia Google *Colab*. O modelo hibrido por sua vez, mescla as características dos serviços anteriores, permitindo que os dados e aplicações se movam entre os dois modelos, dando maior flexibilidade e otimização (AZURE, 2019).

Os serviços de computação na nuvem são comumente divididos em quatro categorias: infraestrutura como serviço, onde se paga pelo uso de servidores,

armazenamentos, redes, sistemas entre outros processos de infraestrutura; plataforma como serviço, utilizado para facilitar o desenvolvimento de aplicações web ou moveis, fornecendo ambiente escalável, gerenciamento e fornecimento de aplicativo de software; sem servidor, modelo com estrutura escalável e gerenciadas por eventos; e software como serviço, que é baseado em assinatura sob demanda, gerenciando a distribuição de aplicativos de software (SOUSA, MOREIRA, MACHADO, 2010).

A arquitetura de computação em nuvem é baseada em camadas, sendo que cada camada pode ser gerenciada ou monitorada separadamente. A primeira, de mais baixo nível, contendo os serviços de rede e armazenamento na nuvem, como centros de dados, clusters e roteadores. Na próxima camada de *middleware* é gerenciado o núcleo logico oferecendo serviços como gerenciamento de virtualização, cobranças, verificação de requisições entre outras. Seguindo encontra-se a camada com ambientes de desenvolvimento responsáveis por dar suporte para as aplicações, geralmente, com programação concorrida, bibliotecas e suportes a diversas linguagens utilizadas pelos usuários desenvolvedores (AZURE, 2019). Ao final, esta a camada de aplicação em nível de usuário, por onde o mesmo consegue utilizar os aplicativos enquanto as demais garantem a estabilidade e disponibilidade do mesmo, conforme Figura 06 e 07.

Figura 6 – Abstração das Camadas



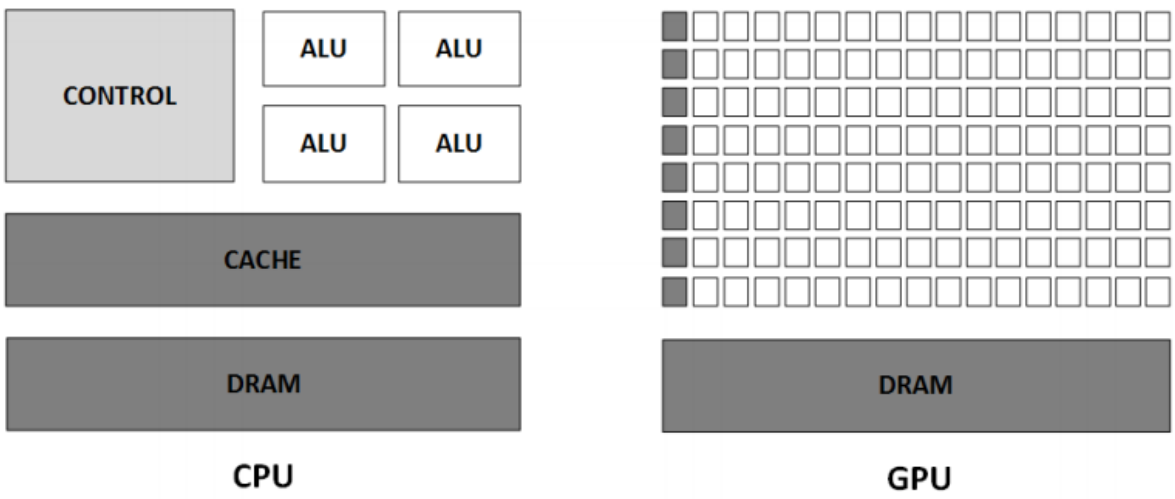
Fonte: Chirigati, 2009.

Figura 7 - Arquitetura computação na nuvem

Fonte: SOUSA, MOREIRA, MACHADO (2010).

Nesse sentido o processamento de imagem na nuvem também ganhou força devido à praticidade e economia que proporciona. Por isso, nesse projeto será utilizada a ferramenta Google *Colab*, um ambiente disponível e acessível por meio da conta do Google drive, com GPU Tesla K80 e PyTorch já instalado além de permitir o uso do *Tensorflow* e Keras.

Figura 8 - CPU X GPU



Fonte: PAILLARD, 2015.

Proporcionando maior poder de processamento que uma CPU, a GPU (Unidade de Processamento Gráfico) utiliza milhares de ALUs em um único processador conforme Figura 08, o possibilita que a execução de múltiplos cálculos simultâneos, e por isso é grandemente utilizada no processo de aprendizado profundo (THEO, 2018).

3 MÉTODOS E MATERIAIS

Neste capítulo serão descritas as ferramentas disponíveis atualmente e os métodos utilizados para a solução do problema apresentado, bem como a modelagem e prototipação do aplicativo Vedere.

3.1 MATERIAIS E BIBLIOTECAS DISPONÍVEIS

Há uma gama de bibliotecas e empresas distintas que oferecem atualmente recursos de reconhecimento de imagem de forma gratuita ou não. Ferramentas que podem se complementar visando proporcionar maior eficácia à aplicação final. Nesta seção serão apresentados os materiais gratuitos utilizados na elaboração do presente estudo.

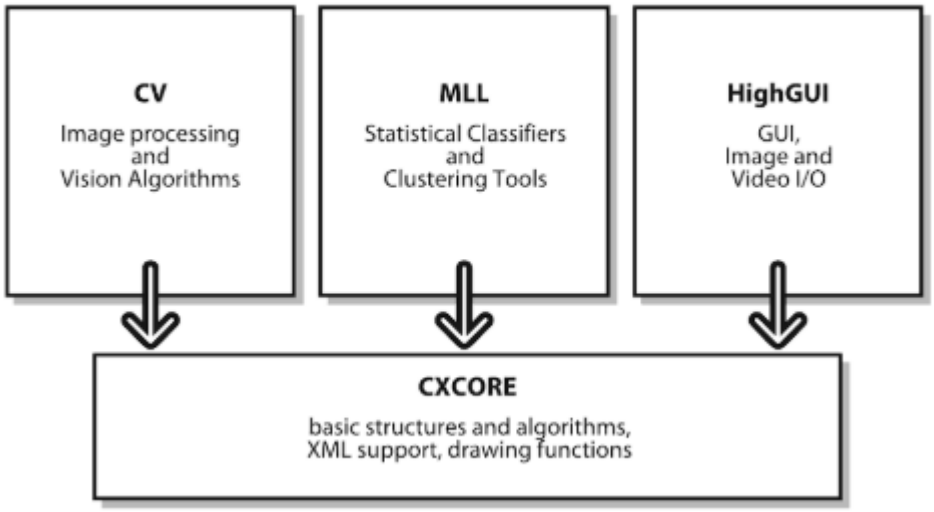
- **OpenCV**

Open Source Computer Vision ou apenas OpenCV é o nome dado a uma biblioteca voltada ao aprendizado de máquina e visão computacional. Desenvolvida originalmente pela Intel, é totalmente livre ao uso comercial ou estudo sob a licença BSD, com diversos módulos de processamento de vídeos e imagens, estrutura de dados, detecção de objetos, ajuste de câmera entre outros, tudo com processamento em tempo real e suporte a diversas linguagens de programação, como Python, C++ e Ruby (OPENCV, 2019).

Em sua biblioteca há cinco módulos disponíveis: processamento de imagem, reconhecimento de padrões, analise estrutural, rastreamento de objetos e reconstrução em 3D. No processamento de imagem, o primeiro passo é converter a imagem para determinado formato ou tamanho seguido de filtros para remoção de ruídos, efeitos de iluminação, intensidade ou qualquer outro elemento que possa prejudicar a identificação do objeto. Um dos principais processos é a segmentação por detecção de borda, sendo classificado como borda o local onde há uma mudança grande nas cores dos *pixels*, de maneira que ao serem detectadas é possível traçar uma caixa entorno da onde se estaria o objeto na imagem.

O OpenCV possui uma estrutura de quatro componentes principais, conforme Figura 09:

Figura 9 - Estrutura básica do OpenCV



Fonte: Bradski e Kaehler, 2008.

O componente CV contém o processamento de imagem básico, além de algoritmos de reconhecimento de imagem de alto nível, Já no ML ficam as bibliotecas de *Machine Learning* com classificadores estáticos e agrupamento. No componente HighGUI ficam as funções e rotinas de I/O para carregar e armazenar vídeos e imagens, e por último no CXCORE estão as estruturas e os conteúdos básicos dos dados (BRADSKI; KAEHLER, 2008).

- **TensorFlow**

Biblioteca também de código aberto relacionado ao aprendizado de máquina e redes neurais profundas, desenvolvido por engenheiros e pesquisadores da área de inteligência de máquina do Google. Sua primeira versão foi disponibilizada em 2015, e pode ser rodado em tanto em gigantes computadores até em pequenos *smartphones*, tornando-se assim uma inovadora biblioteca com uma comunidade grande e ativa (TENSORFLOW,2019).

A versão voltada para dispositivos móveis é chamada de *TensorFlow Lite*, e contém dois componentes principais sendo eles, o interpretador que executa em diferentes tipos de hardware modelos otimizados indo de celulares até micro controladores, e o conversor que transforma esse modelos, otimiza o tamanho e desempenho dos binários. Por ser projetado para simplificar a execução de técnicas

de aprendizado de máquina não há envio de dados de um servidor ao outro, garantindo privacidade, mas consumindo uma fonte considerável de energia.

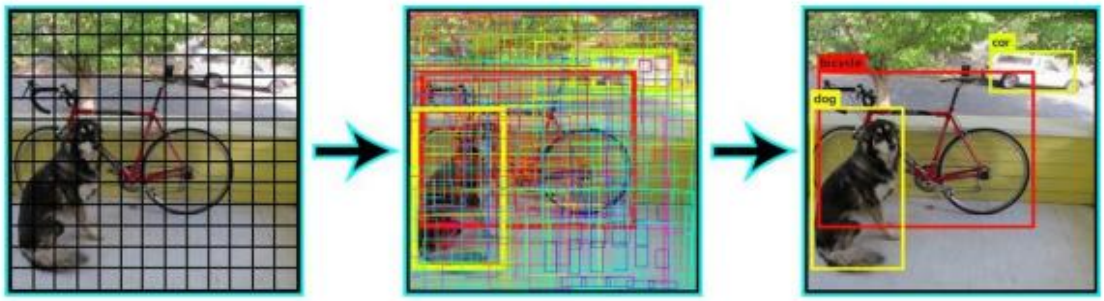
Resumidamente, o *TensorFlow Lite* é um complemento do *TensorFlow* voltado a dispositivos com baixa latência, utilizando um modelo de conversão de representação binária reduzida, *FlatBuffer*, e um núcleo de interprete reduzido e temporariamente sem suporte a treinamento no próprio dispositivo. Reúne diversos modelos e algoritmos de aprendizado e os torna relevantes usando a linguag*em* Python para estruturar a *API (Application Programming Interface)* usada na integração de sistemas, enquanto em executa na linguagem C++ os aplicativos. Há outras bibliotecas similares ao *TensorFlow* sendo as três principais: CNTK ou *Microsoft Cognitive Toolkit* voltada mais a criação das redes neurais profundas e por isso possui maior conjunto de dados e *APIs*, no entanto possui uma elevada curva de aprendizado. O *PyTorch* que também utiliza Python, componentes e modelos interativos se tornando uma viável opção para projetos rápidos e ainda o Apache MXNet, desenvolvido pela Amazon, que oferece suporte, variedades de *APIs* e bom dimensionamento.

- **Yolo**

Existem três grandes detectores de objetos relacionados à aprendizagem profunda, sendo o R-CNN e distribuições, SSDs detector de disparo único e o Yolo. Desses a Yolo vem crescendo exponencialmente tanto em usuários quanto em capacidade, a abreviação vem do inglês "*You Only Look Once*" significando em português "Você só olha uma vez", sendo em um sistema moderno de detecção de objeto em tempo real extremamente rápido, lançado originalmente também em 2015, de código aberto e de livre utilização, utiliza redes neurais convolucionais cuja arquitetura principal é denominada *Darknet*, rede neural que funciona em GPU ou CPU.

Segundo Jokela (2018) nessa metodologia de detecção cada imagem é dividida em grade NxN e cada grade por sua vez contém K caixas delimitadoras e sua respectiva confidencialidade, sendo essa a precisão e a informação de que a caixa contém integralmente um objeto, como mostra a Figura 10:

Figura 10 - Processo de detecção de imagem Yolo



Fonte: Jokela, 2018

- **Dataset Coco**

A palavra *Dataset* significa conjunto de dados, mais precisamente dados tabulares formatados em planilhas onde cada coluna é a descrição ou características dos registros na linha. O conjunto de dados escolhido para esse projeto foi o Coco, uma vez que o mesmo é voltado à segmentação e legenda de objetos em grandes escalas, com 80 categorias e 1,5 milhão de instância de objetos (COCO, 2018). Na metodologia de detecção de objetos primeiramente é desenhada uma linha de contorno no objeto, como exemplificado na Figura 11, e montado no formato JSON uma lista de categoria e anotações. Originalmente há 80 categorias, sendo possível criar ainda uma lista com categorias novas, mas respeitando um identificador único para cada.

Figura 11- Detecção de objetos em segmentação

Fonte: COCO, 2018.

- **Android Studio**

A plataforma de desenvolvimento *Android Studio* é uma IDE disponibilizada pela Google. Baseado na *Intellij*, o *Android Studio* é uma ferramenta poderosa de desenvolvimento mobile, possui um editor de layout com suporte ao gesto gráfico de arrastar e soltar, além de uma compilação baseada em *Gradle* com ferramentas que avaliam o desempenho e a compatibilidade da aplicação que está sendo desenvolvida, foi lançado no evento Google I/O 2013, mas a versão 1.0 do ambiente só saiu oficialmente no final de 2014. (DEVELOPERS, 2019).

- **Google *Colab***

Também chamado de colaborativo é um serviço gratuito hospedado na nuvem pelo Google, como incentivo aos estudos de aprendizado de maquina e inteligência artificial, dada a grande barreira de alto poder computacional exigente para esses tipos de aplicações (GOOGLE, 2019). Possui suporte para aceleração de GPU (Unidade de processamento gráfico), comandos *bash*, principais bibliotecas do Python pré-instaladas como *TensorFlow*, *Scikit-learn* entre outras. Python 2.7 e 3.6, construído com base no *Jupyter Notebook* e permite compartilhamento do mesmo entre seus usuários, dispõe de um Tesla K80 com 12 GB de memória por 12 horas, sendo desconectado após esse período.

Integrado ao *Tensorflow* e acessível por meio do *Colab*, esta a ferramenta *TensorBoard*, ferramenta de visualização de gráficos e informações de desempenho do treinamento. A mesma lê os eventos do *Tensorflow* e já vem previamente instalado, podendo ser inicializada pelo comando abaixo:

tensorboard --logdir=[dir]

Onde em [dir] deve ser passada a localização dos eventos, sendo extremamente útil para visualização de gráficos de perdas, acurácia e precisão.

- ***Jupter Notebook***

Ambiente web computacional interativo para criação de documentos no formato estilo JSON com uma lista ordenada de células que podem conter desde formulas matemáticas, plotagens, imagens ou texto. Podem ser executadas sequencialmente ou selecionadas manualmente, sua extensão é.jpynb e pode ser convertido em outros formatos como slides, PDF, Python, HTML entre outros, seja interface ou linha de comando (JUPYTER, 2019). Suas bibliotecas são *open-source* e pode conectar núcleos de diferentes linguagens de programação como R, Ruby, Scala, Python e mais 40 linguagens de programação.

- **Anaconda**

A Anaconda ou simplesmente conda possui código aberto e simples manuseio permite trabalhar com ciência de dados Python/R além de aprendizado de maquina (ANACONDA ENTERPRISE, 2019). Gerencia ambientes, dependências e bibliotecas com o conda, permite *download* de mais de 1500 pacotes de ciência de dados, podendo ser usada em Windows, Linux e MacOS, visualização de resultados com *Matplotlib, Bokeh, Datashader e Holoviews* entre outros, com escalabilidade e desempenho com *Dask, NumPy, pandas e Numba*.

- **Trello**

Trello é uma ferramenta de gerenciamento de projetos no modelo kanban, onde é possível criar quadros para gerenciar o que está sendo feito, a fazer ou já esta pronto. Permite o compartilhamento com times ou pode ser utilizada para planejamento individual, além de anexar imagens, documentos, links, lembretes e personalização de cor e fundos (TRELLO, 2015).

- **Astah**

Ferramenta para modelagem de diagramas UML (*Unified Modeling Language*), desenvolvida em Java, utilizada para modelar projetos de *software* orientados a objetos, disponível em versão gratuita para estudante e também paga na versão profissional (ASTAH, 2019).

- **Marvel**

Uma plataforma de design online para criar *wireframes*, *mockups*, protótipos de sites ou aplicativos para diferentes dispositivos, a partir do navegador, permitindo ainda sincronização com *Photoshop* e importação de ícones e imagens (MARVEL, 2019).

3.2 MÉTODOS

3.2.1 Requisitos Do Sistema

O termo requisito é usado para descrever todas as funções que o sistema abordado deve realizar e quais as restrições presentes no mesmo. A seguir serão listados todos os requisitos levantados na elaboração do aplicativo Vedere.

3.2.1.1 Requisitos Funcionais

Os Requisitos funcionais determinam as funcionalidades e como o sistema vai reagir mediante determinadas situações, para o aplicativo foram levantados cinco requisitos de software essenciais sendo:

- RF001 – Os usuários devem poder obter a identificação do objeto capturado pela câmera em tempo real.
- RF002 – O aplicativo deverá capturar a imagem em tempo real.
- RF003 – O aplicativo deverá processar algoritmo de reconhecimento.
- RF004 – O aplicativo deverá devolver o nome do objeto identificado por meio de áudio.
- RF005 – O aplicativo deverá solicitar permissão de acesso à câmera.

3.2.1.2 Requisitos não funcionais

Já os requisitos classificados como não funcionais são, normalmente, as restrições de funções ou serviços oferecidos pelo sistema. Sendo:

- RFN001 – O aplicativo será, inicialmente, voltado somente ao sistema Android.
- RFN002 – A aplicação requer conexão com a internet.
- RFN003 – Só serão identificados objetos pertencentes ao *Dataset* Coco.
- RFN004 - Disponível apenas para versão 8.1 do Android ou posterior.

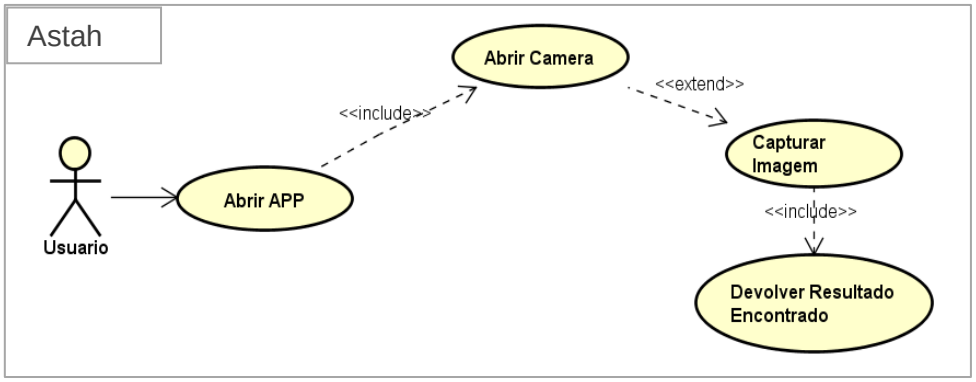
3.2.2 Diagramas

Diagrama pode ser definido como um esquema gráfico simplificado, usado para esclarecer os requisitos e funções do sistema, nesse estudo foi adotado o padrão de modelagem UML, pois além de ser extremamente utilizado, possui diferentes modelos para construção de diagramas que contempla quase todos os aspectos da modelagem de software.

3.2.2.1 Diagrama de Caso de Uso

O diagrama de caso de uso é voltado às características e funcionalidades, bem como a maneira com que as mesmas se relacionam com os usuários ou entidades terceiras no sistema. A Figura 12 representa o diagrama de caso de uso do aplicativo Vedere, contendo apenas um usuário como ator responsável pela ação “abrir aplicativo” que desencadeia todas as demais ações de reconhecimento até obter o retorno com nome do objeto ou mensagem de erro quando não for possível a identificação.

Figura 12 - Diagrama de caso de uso - Reconhecer imagem



Fonte: Próprio autor.

A ação de reconhecimento depende de uma pré-condição e pode apresentar dois diferentes resultados, conforme especificação do caso de uso abaixo:

Quadro 1 – Reconhecer imagem

Caso de Uso:	Reconhecer imagem
Descrição:	Detecta e apresenta o nome do objeto capturado pela câmera.
Atores:	Usuário
Pré-condição:	O usuário deve abrir o aplicativo.
Pós-condição:	-
Fluxo principal:	1. O usuário abre o aplicativo, que inicia a câmera. 2. A câmera procura e captura objeto. 3. O algoritmo de reconhecimento compara objeto. 4. O aplicativo devolve nome do objeto em áudio.
Fluxo alternativo:	FA01 - No passo 3 do fluxo principal o algoritmo não consegue encontrar o nome do objeto capturado. 1. O aplicativo devolve mensagem de erro “Objeto não reconhecido!”. 2. O caso de uso se encerra
Fluxo de exceção:	-

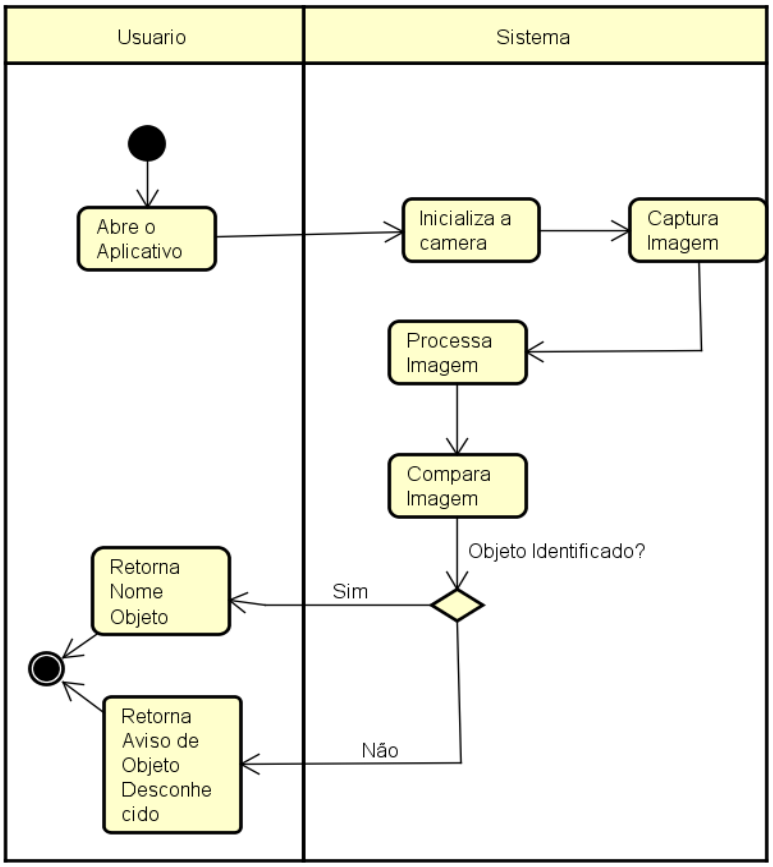
Fonte: Próprio autor.

3.2.2.2 Diagrama de Atividades

O Diagrama de Atividades é utilizado para descrever as tarefas envolvidas na execução de uma determinada atividade e como as mesmas se relacionam, pode conter regiões chamadas de *swimlanes* que são associadas a um objeto maior e atividades conectadas por transições.

A Figura 13 representa graficamente a sequência de ações a serem executadas na utilização do aplicativo Vedere, não são muitos os passos para execução da ação de reconhecer, sendo que alguns processos que ocorrem somente na primeira vez em que o aplicativo é utilizado, como a solicitação de permissão para acesso a câmera, galeria e áudio, não foram representados no diagrama de atividades principal.

Figura 13 - Diagrama de Atividades reconhecimento de imagem

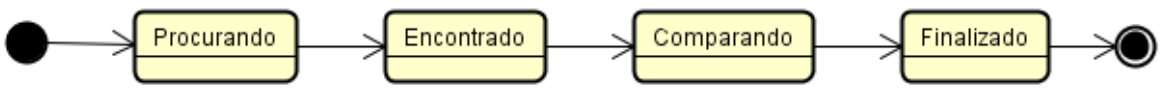


Fonte: Próprio Autor

3.2.2.3 Diagrama de Estados

O diagrama de estado representa todas as transições e *status* que a aplicação pode assumir sequencialmente. Os estados que podem ser assumidos pela aplicação são sequencialmente representados pelo diagrama de estados na Figura 14:

Figura 14 - Diagrama de Atividades reconhecimento de imagem



Fonte: Próprio Autor

3.2.3 Protótipo

A prototipação dentro do desenvolvimento de software tem como objetivo avaliar as ideias e modelos propostos antes de construir o projeto final. Sendo, portanto a maneira mais econômica e eficaz de avaliar um projeto, não requer softwares de edição de gráfico robustos e possibilita mostrar ao cliente o que está sendo desenvolvido e receber retorno do mesmo.

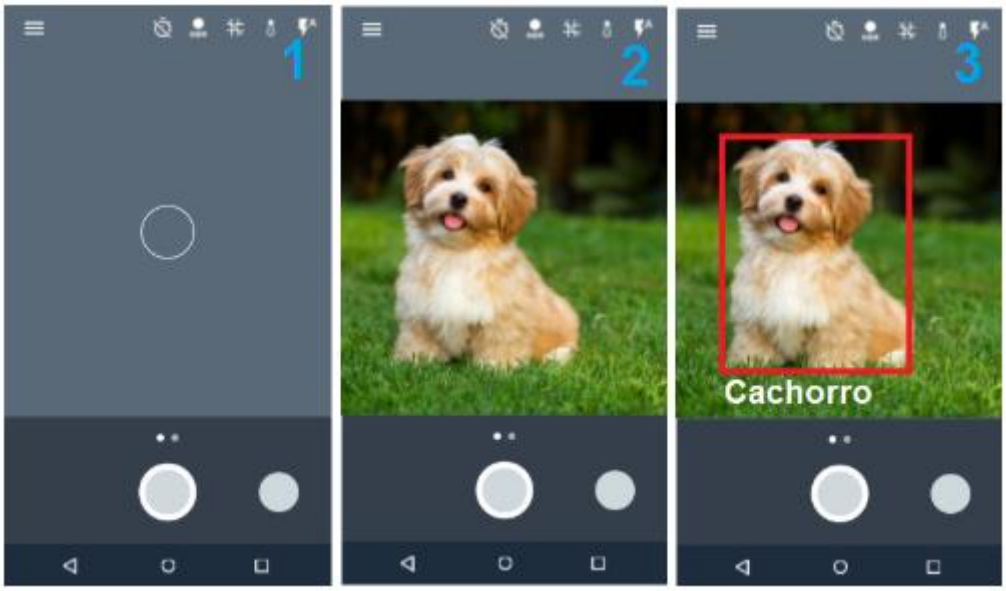
Existem protótipos de diferentes fidelidades ao layout final, sendo classificado em baixo, médio e alto nível. Chamado de rascunho os protótipos de baixo nível são caracterizados por não incluírem detalhes visuais ou validações das telas, mas fornecerem um panorama geral do projeto. Realizados na fase inicial do projeto e normalmente desenhados a mão em papel, já os chamados *wireframes*, são mais próximos da etapa final, desenvolvidos na estruturação da arquitetura da informação, é comum nesse modelo a utilização de um software de edição. O protótipo de alta fidelidade, *mockupes*, é funcional e formam a mais próxima aparência e funcionalidade do projeto final, a criação ocorre na fase final do projeto, por meio de poderosas ferramentas de edição (FABRÍCIO GUILHERMO, 2019).

Com os recursos de acessibilidade disponíveis tanto no sistema operacional Android como no IOS, é possível configurar o modo leitura de tela e a cada toque o sistema devolver em áudio o nome do item ou aplicativo selecionado, permitindo que pessoas com DV realizem operações por meio de smartphone de modo autônomo.

Como o público alvo desse aplicativo são pessoas com deficiência visual, o *design* foi pensado para possuir uma quantidade mínima de interação, para que os recursos de leitura de tela possa permitir a utilização. Sendo composto apenas por uma tela principal, figura 15, que abrirá a câmera para captura da imagem, sem botões e com inicio automático da fala quando o objeto for reconhecido.

A Figura 15 apresenta o seguinte fluxo: Assim que inicializado o aplicativo é aberta a câmera (1), que quando direcionada a um objeto captura (2) e identifica, retornando o nome do mesmo em áudio e em texto na tela bem como a marcação da caixa onde foi encontrada (3).

Figura 15 - *Wireframe* da tela principal do aplicativo Vedere.



Fonte: Próprio autor utilizando a ferramenta Marvel.

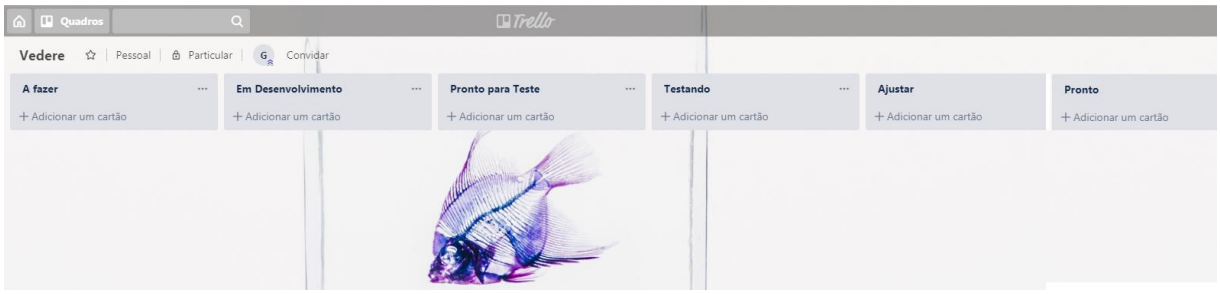
3.2.4 Métricas e Cronograma

3.2.4.1 Metodologia de Desenvolvimento Kanban

Como em todo processo de construção de software foi adotado uma metodologia de desenvolvimento, o Kanban foi escolhido por sua simplicidade e adaptação ao desenvolvimento solo. Desenvolvido originalmente pela Toyota, permite acompanhar e controlar o progresso do projeto de maneira visual, utilizando pequenos cartões coloridos, que representam as tarefas, dispostos em um fundo com divisões em colunas que representam o status das tarefas compondo a visão do status geral do projeto no Kaban.

Esse método possui quatro metodologias principais sendo: Começar com que se faz agora, buscar mudanças para evoluir, respeitar papéis e responsabilidades atuais e incentivar a autonomia, é receptivo a mudanças e permite eliminação de gargalos (KLEBER BERNARDO, 2014). Para aplicação dessa metodologia foi utilizada a ferramenta Trello que atua como modelo digital melhorado do quadro branco, conforme exemplificado na Figura 16:

Figura 16 – Estrutura de Quadros Trello no desenvolvimento do aplicativo Vedere.



Fonte: Próprio autor utilizando a ferramenta Trello.

3.2.5 Análise E Design

O aplicativo proposto possui três componentes principais: a entrada de dados por meio da câmera, o processamento interno e a saída em forma de alerta ao usuário.

3.2.5.1 Arquitetura do Sistema

O *TensorFlow Lite* possui um modelo com quatro componentes principais sendo: o formato de arquivo com as mínimas dependência, leve e com cálculos de gráficos de flutuador de 32 bits. O interpretador com mínima sobrecarga e menores números de operadores no *kernel*, e por isso não suporta todos os modelos, mas contém um conjunto interno otimizado para CPU que funciona tanto em *float* como em *quantized*. Dependendo da capacidade do dispositivo móvel o fluxo do tensor é direcionado a *API* do NN (Rede Neural), caso contrário é direcionado na CPU. A *API* do NN é compatível com a versão 8.1 ou superior do *Android Oreo*, com a arquitetura conforme Figura 17.

No centro está o aplicativo Android, que não necessariamente precisa acessar diretamente a *API* de rede neural, mas sim conectar com a interface de *Machine Learning*, representada pelo interpretador e o tempo de execução NN. Sendo esse último responsável por comunicar com a camada de abstração do *hardware* (BRIJESH, 2017).

Figura 17 - Arquitetura da API de Rede Neural Android

Fonte: Brijesh, 2017.

3.2.5.2 Compatibilidade Dos Modelos *Tensorflow* e a Versão *Lite*

O *site* do *Tensorflow* possui diversos exemplos e guias de como aplica-lo em aplicativos Android e IOS, além de explicar detalhadamente cada conceito utilizado, é possível aplicar um modelo já existente ou construir um totalmente novo. É chamado de modelo toda composição de dados, conhecimento, logica e o aprendizado de maquina, treinada para resolver um determinado problema. Não é possível treinar ou até mesmo criar um modelo usando o *TensorFlow Lite*, é preciso antes utilizar um modelo do *TensorFlow* e posteriormente converte-lo para mobile, atualmente há limitado o conjunto de modelos que são suportados na versão *Lite*.

Na grande maioria das operações suportadas pela versão *Lite* é utilizada a inferência de ponto flutuante e quantizado, sendo que a diferença entre os dois esta na forma como são convertidos, onde a quantificação utiliza técnicas que representam precisões reduzidas de pesos e ainda possui suporte em plataformas de CPU existentes, reduzindo custo de acesso a memoria. O treinamento com esse reconhecimento permite a classificação mais rápida de imagem e com menor tamanho, mas perde na precisão das respostas. Já os modelos com ponto flutuante ganham na precisão e aumentam o tamanho e reduz o desempenho, mas para conseguir aceleração de GPU é preciso aplicar esse modelo.

Criadas para obter um alto rendimento de carga de trabalho paralelizável, os GPUs são os mais indicados para redes neurais profundas, ao contrato das CPUs elas calculam com números flutuantes de 32 ou 16 bits e não exigem a quantização para um desempenho integral, além de consumir menos energia e gerar menor calor.

O processo chamado de inferência é a previsão baseada nos dados de entrada e suas respectivas saídas, o intérprete do *Tensorflow Lite* é responsável por essa inferência usando a ordenação estatística dos gráficos e um alocador de memoria para garantir latência, inicialização e carga mínima. Após o carregamento do modelo ocorre à transformação dos dados e a inferência em execução com uso da *API* e por fim a interpretação da saída. No Android esse processo pode ser realizado usando *APIs* Java ou C++ diretamente nas classes de atividades do Android.

4 TESTES

O teste é uma etapa no processo de desenvolvimento de software fundamental para a manutenção da qualidade, identificando possíveis falhas, melhorias e adequação aos requisitos levantados. As técnicas tradicionais de testes trabalham com uma entrada X que resulta em uma saída Y, no entanto quando se trabalha com aprendizado de maquina a saída varia de acordo com o tempo e treinamento do algoritmo e pode ainda ser apresentada em probabilidades. Nesse sentido, esta seção apresenta o plano de teste adotado para o aplicativo Vedere bem como as métricas utilizadas para medir a qualidade do produto final.

4.1 PLANO DE TESTE

Para medir a confiabilidade e eficácia do aplicativo Vedere será adotada matriz de confusão. Esse modelo é formado por conjuntos dispostos em linhas e colunas com números de unidades de uma determinada caraterística e sua relação real. No geral a matriz calcula a quantidade acertos (verdadeiro positivo) e erros (falso positivo) e com base neles quantifica a sensibilidade, acurácia e especificidade. Sendo a sensibilidade à proporção entre a quantidade de verdadeiro positivo e o total geral da amostra. Já a acurácia equivalente à relação de casos de acertos, seja em reconhecimento do objeto ou em mensagem de erro, pela quantidade geral da amostra. E por fim a especificidade relacionada à quantidade de erros verdadeiros, quando o objeto de entrada não faz parte do banco de dados conhecido do algoritmo, pelo conjunto total.

Com base nesse processo, foi elaborado três cenários de possíveis resultados da matriz de confusão no momento de execução dos testes no aplicativo já em produção.

Quadro 2 – Possíveis cenários de testes

Otimista	90% de precisão
Realista	50% de precisão
Pessimista	30% de precisão

Fonte: Próprio Autor.

Na previsão de cenário otimista os testes realizados pela matriz de confusão deverão atingir 90% de acertos na classificação dos objetos, já no cenário mais provável ou também chamado de realista espera que ao menos 50% das imagens testadas tenham resultados assertivos e por ultimo levando em consideração fatores externos e falhas que possam ocorrer espera-se que no menino 30% das classificações estejam corretas.

5 RESULTADOS E DISCUSSÃO

Nos capítulos anteriores foram apresentadas as principais ferramentas, metodologia e praticas para o desenvolvimento do aplicativo Vedere. Nesta parte serão apresentados os resultados obtidos bem como a importância que cada etapa e recurso possuem sobre o resultado final.

5.1 TREINAMENTO DO ALGORITMO

O treinamento foi realizado em Python3 na *IDE Jupyter Nootebook* no Google *Colab*, utilizando o modelo SSD *MobileNet* (TENSORFLOW MODELS,2019) quantificado e treinado no conjunto de dados COCO (COCO,2019) O numero de etapas de treinamento foi 60000, as etapas de avaliação 20 e o tempo de execução do *notebook* em GPU bem como Keras para criação das camadas dos modelos. O conjunto de imagens utilizado contem os objetos marcados em inglês, totalizando 80 categorias conforme tabela abaixo.

Quadro 3 - Conjunto de objetos identificáveis pelo aplicativo Vedere (traduzidos em português)

Pessoa	Cavalo	Taco de beisebol	Cenoura	Microondas
Bicicleta	Ovelha	Luva de Beisebol	Cachorro quente	Forno
Carro	Vaca	Skate	Pizza	Torradeira
Motocicleta	Elefante	Prancha de surfe	Rosquinha	Pia
Avião	Urso	Raquete de tênis	Bolo	Geladeira
Ônibus	Zebra	Garrafa	Cadeira	Livro
Trem	Girafa	Taça de Vinho	Sofá	Relógio
Caminhão	Mochila	Copo	Vaso de planta	Vaso
Barco	Guarda-chuva	Garfo	Cama	Tesouras
Semáforo	Bolsa	Faca	Mesa de jantar	Urso teddy
Hidrante	Gravata	Colher	Banheiro	Secador de cabelo
Sinal de parada	Mala de viagem	Tigela	Televisão	Escova de dente
Parquímetro	Avelã	Banana	Computador portátil	Cão
Banco	Esquis	Maçã	Rato	Pipa
Pássaro	Figo	Sanduíche	Controlo remoto	Brócolis
Gato	Bola de esportes	Laranja	Teclado	Celular

Fonte: Próprio Autor.

É de extrema importância que as imagens utilizadas para o treinamento seja o mais semelhante possível aos objetos reais, sendo que para cada imagem é gerado um arquivo no formato XML contendo as anotações da mesma, utilizando a ferramenta *LabelImg*. Após o treinamento é extraído o gráfico de inferência utilizado para executar a detecção de objetos, o tamanho das imagens de saída, em polegadas, é de (12,8) e as matrizes no formato *numply float32*.

Posteriormente foi realizado a exportação para o *tflite*, e obtido o modelo otimizado por meio do *TensorFlow Lite* Otimizando *Conveter* (TOCO), onde o gráfico congelado do *tflite_graph.pb* é convertido para *flatbuffer TensorFlow Lite* (*detect.tflite*) arquivo contendo todos os parâmetros a ser interpretado no dispositivo *Android*. A API responsável por essa conversão é a *tf.lite.TFLiteConverter* sendo utilizada neste projeto a classe *from_keras_model()*. Nesse processo a imagem da câmera é redimensionada para 300x300 pixels, resultando em uma saída de quatro matrizes sendo elas: classe de detecção, caixas de detecção, números de detecções e pontuações.

5.2 Saída em áudio com *TextToSpeech*

O reconhecimento de texto em imagens naturais é ainda um maior desafio, devido às distorções, ruídos ou formas de escritas diferentes com contornos e ondulações altamente estilizadas. Uma vez que identificado e classificado o objeto a saída é feita marcando o quadrado que o contém, nomeando e também realizando a leitura em áudio do mesmo por meio do mecanismo *TextToSpeech*, que sintetiza em reprodução imediata ou criando um novo arquivo de som. Primeiramente foi criada a instancia da classe informando o ‘*Context’* e o *OnlnitListener* responsável por verificar se o objeto foi criado com sucesso ou não, já o método *setLanguage* é o responsável por definir o idioma em que o áudio será executado, no entanto, nem todas as linguagens são atualmente suportadas, sendo que para o aplicativo Vedere foi definido o idioma Inglês.

A grande dificuldade encontrada na leitura, é a pronuncia de números que variam de acordo com o contexto que esta inserida, como por exemplo, “1234” que pode ser um valor único ou ainda códigos distintos, nos testes realizados pelo aplicativo notou-se que cada número é lido individualmente, mesmo se tratando de conjuntos monetários. Os textos estilizados com sombreamentos e fontes

diferenciadas também não foram identificáveis pelo aplicativo, somente os textos impressos e no ambiente com boa luminosidade.

5.3 Avaliação e testes de desempenho

Para avaliar o funcionamento do aplicativo foram utilizadas instancias diferentes das utilizadas durante o treinamento. Assim o conjunto de dados foi dividido entre treinamento e teste, de modo que o desempenho seja com base no percentual de inconsistências entre esses conjuntos. O algoritmo de aprendizado reúne um modelo, com base em todos os exemplos, que tente minimizar a perda. Sendo que é chamado de perda o valor atribuído a uma previsão incorreta, comumente possuindo o valor zero quando a previsão estiver correta e algum valor positivo quando errada, variando de quão distante da verdade a classificação ocorreu. Durante o treinamento são salvos os pesos e desvios periodicamente, sendo utilizada para montar o gráfico de inferência congelada como mostra a Figura 19.

Figura 18 - Épocas de treinamento

```
Epoch 5/10
1563/1563 [==============================] - val_acc: 0.6610 - loss: 1.8456 - acc: 0.6376 - val_loss: 1.75
Epoch 6/10
1563/1563 [==============================] - val_acc: 0.6616 - loss: 1.7572 - acc: 0.6624 - val_loss: 1.74
Epoch 7/10
1563/1563 [==============================] - val_acc: 0.6938 - loss: 1.6818 - acc: 0.6818 - val_loss: 1.65
Epoch 8/10
1563/1563 [==============================] - val_acc: 0.6838 - loss: 1.6117 - acc: 0.7008 - val_loss: 1.64
```

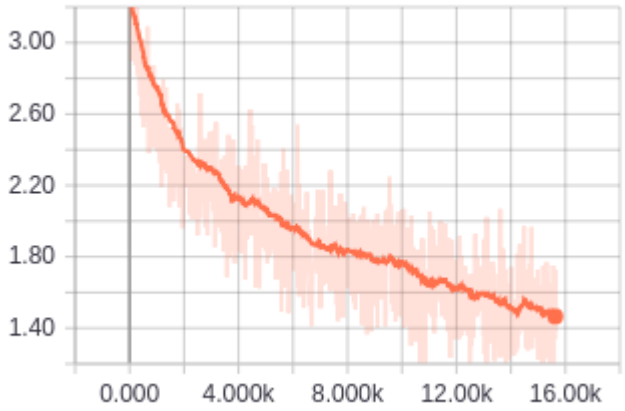
Fonte: Próprio Autor.

Todo o treinamento durou cerca de 3 horas, sendo que na ultima época a perda se manteve em aproximadamente 1,56 e a em precisão 73%. O Google *Colab* permite interação com o *TensorBoard*, responsável por fornecer gráficos que sinalizam o desempenho do treinamento e demais informações sobre o algoritmo, o gráfico (Figura 18) exibe a perda diminuindo a cada interação até chegar ao ponto de equilíbrio e solução.

É chamado de perda à medida que a rede neural se ajusta e comporta os dados, sendo que à medida que se aproxima de zero, melhor o desempenho. Inicialmente os pesos atribuídos pela rede neural foram aleatórios, assim no

momento inicial a perda foi mais expressiva e as previsões mais incorretas, conforme gráfico abaixo.

Figura 19 – Gráfico de perdas



Fonte: Tensorboard, 2019.

Um grande desafio na identificação de objetos é quando o mesmo está posicionado em um ângulo diferente do qual foi usado no treinamento do algoritmo, ou ainda esta contido em outro objeto que também treinado para detectar. Essa sobreposição de objetos ou ainda figuras semelhantes levam a classificações incorretas, mas com grau de precisão elevado, como mostra a Figura 21.

Figura 20 - Identificação *Airplane* (Avião) e *Bird* (Pássaro).

Fonte: Próprio Autor.

Na primeira instancia o aplicativo classificou corretamente a imagem como sendo um avião com 82,42% de precisão, assim como na terceira instancia classificada como 73,83% Pássaro, porém na segunda imagem o pássaro estava em um ângulo diferente que ocasionou certa semelhança as assas de um avião levando a classificação incorreta do mesmo com 69,92% de precisão. Sendo que em *Machine Learning* a precisão corresponde ao percentual de certeza ou de identificação de todos os itens, nesse caso, capturados pela câmera, divididos por todos os objetos nela contidos.

Outro caso de previsão falsa é o da Figura 22 onde em uma imagem há dois objetos como Cão e Gato, mas no momento de classificar o destaque foi para o maior objeto no caso o cão, mesmo que algoritmo consiga diferenciar cada objeto como mostra a segunda e terceira instancia na Figura.

Figura 21 – Classificação de *Dog* (Cão) e *Cat* (Gato).

Fonte: Próprio Autor.

Quando ambos os objetos identificáveis possui o mesmo tamanho na imagem, o retorno foi às delimitações sobrepostas, conforme Figura 23.

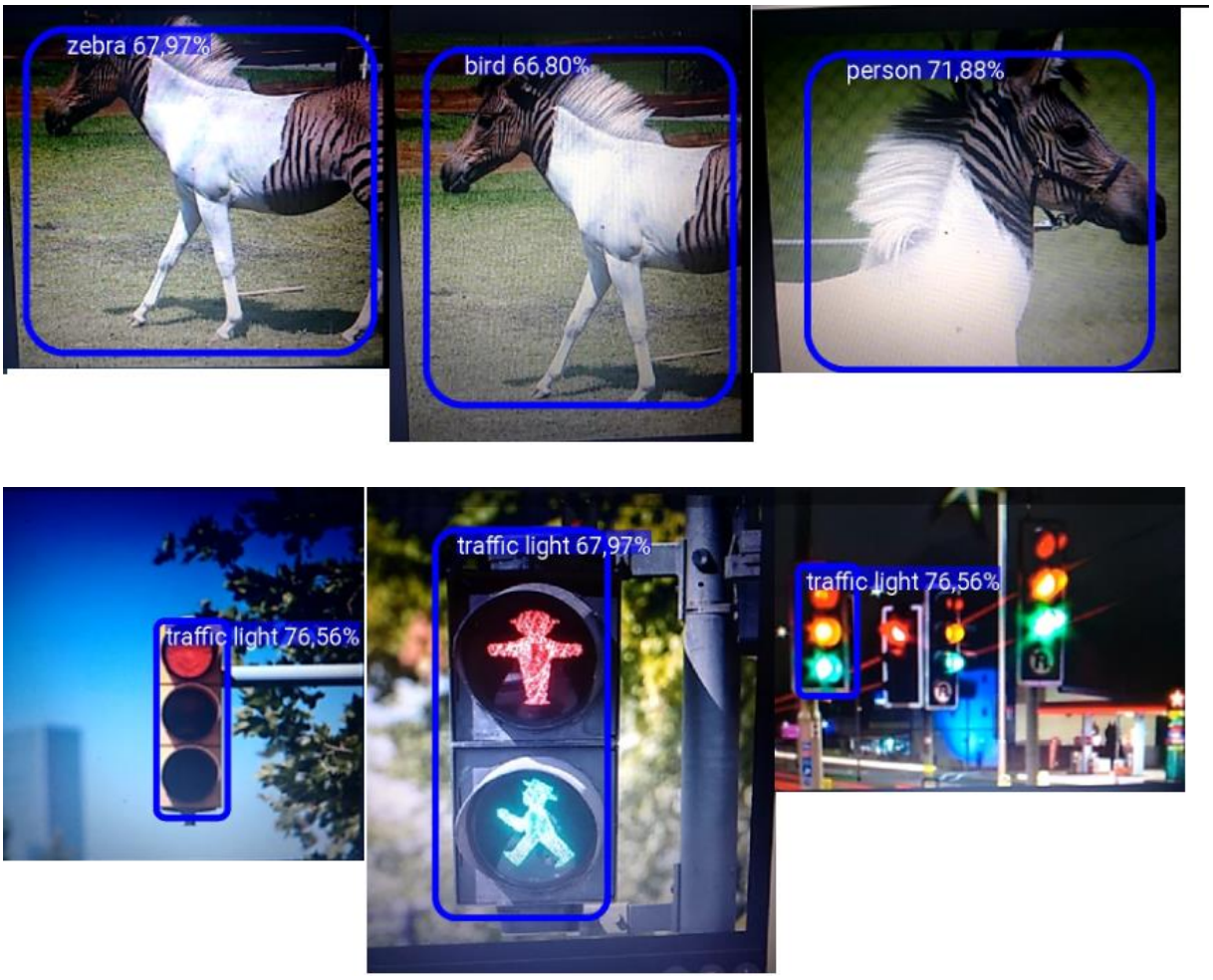
Figura 22 - Ciclistas

Fonte: Próprio autor.

Em ambos os casos para impedir a falsa previsão é preciso treinar o modelo com grande quantidade de amostra e de vários ângulos, de forma a permitir uma base solida de comparação e classificação assertiva. O ruído e a qualidade da câmera bem como a iluminação também afeta diretamente esse processo, foi identificado nos testes, que em locais um pouca luminosidade há um maior grau de previsões falsas ou até mesmo dificuldade para identificar algum objeto.

Na Figura 24 há uma imagem de um cavalo-zebra que devido possuir as características de ambas as espécies fora classificado diferentemente nas três angulações, o mesmo não ocorreu com a classificação do semáforo onde os três modelos distintos foram classificados corretamente:

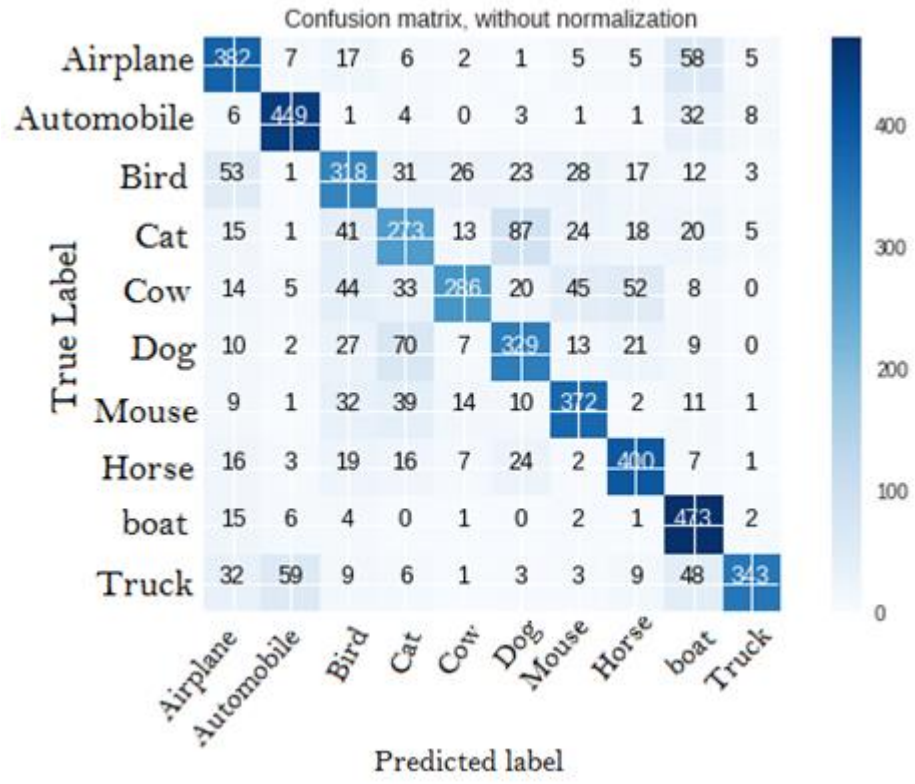
Figura 23 – Classificação Cavalo-Zebra e Semáforo



Fonte: Próprio autor.

Assim como ocorre nas comparações *booleanas*, o classificador retorna um valor positivo ou negativo de acordo com a condição estabelecida, nesse sentido, a métrica mais utilizada para compreensão da classificação de objetos é a matriz de confusão ou tabela de contingencia, Figura 25, onde as classes previstas são resultados do algoritmo e a verdadeira etiqueta o que de fato são os objetos.

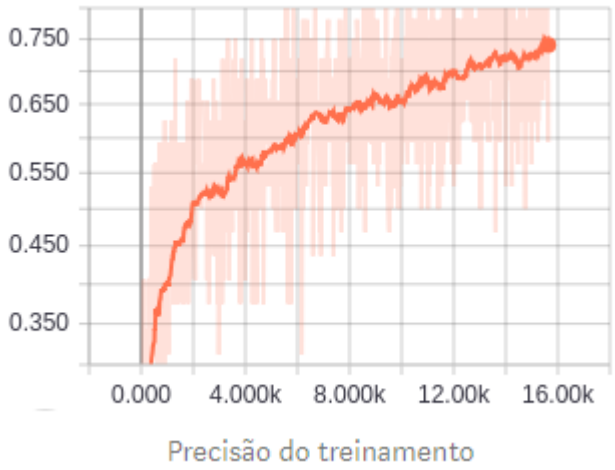
Figura 24 – Matriz de confusão



Fonte: Próprio autor.

A matriz de confusão é bidimensional, as linhas representam os rótulos reais e as colunas às previsões, com esse gráfico foi possível ver que a grande maioria das previsões foi assertiva, mas casualmente as semelhanças entre algumas imagens como cão e cato ou avião e pássaro, faz com que a previsão seja falsa. Juntamente com o gráfico de precisão (Figura 26), pode comprovar que o aplicativo ficou dentro do cenário “Otimista” previsto no plano de teste, tendo mais de 50% de assertividades nas classificações.

Figura 25 - Precisão do treinamento



Fonte: Tensorboard, 2019.

5 CONCLUSÃO

Assim como destacado no Estatuto da Pessoa com Deficiência, o direito a igualdade e liberdades fundamentais são essenciais e devem ser promovidos visando uma sociedade justa e igualitária. Este projeto, portanto teve como objetivo a aplicação dos conceitos de aprendizado de maquina a tecnologias assistivas, tendo em vista fomentar o uso dos avanços tecnológicos na melhoria de vida e inclusão de pessoas com deficiência visual. O campo da visão computacional junto às técnicas de *Deep Learning* possibilita que um objeto seja reconhecido em uma imagem capturada por dispositivo móvel, dando origem a uma poderosa ferramenta de auxílio e de baixo custo, com grande potencial de aperfeiçoamento e utilização.

No entanto, se para o ser humano adulto é fácil distinguir diferentes objetos em uma cena ou até mesmo diferenciar uma laranja de uma maça ou uma mexerica, para crianças não é tão fácil, assim como para o aprendizado de maquina. A grande variedade de cores, tamanhos e formatos que cada objeto natural pode possuir contribui para complexidade da distinção, mas não impossibilita o aprendizado, de modo que, a visão computacional cresce exponencialmente, não somente na detecção de objetos, mas pessoas, agricultura, robótica e até na identificação de doenças.

O avanço da tecnologia e a distribuição de pesquisas permitem que grandes quantidades de imagens sejam processadas e utilizadas como base para a Deep Learning, o *dataset* COCO utilizado nesse projeto se mostrou altamente eficiente e de fácil manuseio, os modelos pré-treinados disponíveis no chamado zoológico de detecção de objetos do *Tensorflow* possibilitou a transferência de aprendizado de maquina, promovendo a eficiência e significativa redução no tempo de treinamento, bem como a utilização do Google *Colab*.

A classificação de objetos em tempo real é complexa devido à variedade de instancias semelhante, e aos ruídos e efeitos de luminosidades que o ambiente pode possuir. Imagens com mais de uma instancia, são as que possuem o maior índice de previsões falsas, as falhas encontradas no percurso mostra que ainda há um grande caminho para se percorrer em busca do aperfeiçoamento da visão computacional,

mas que hoje é possível a utilização de tecnologias gratuitas e eficientes para construir um aplicativo de detecção de objetos estáticos de baixo custo em constante melhoria, mas que a curva de aprendizado do *Tensorflow* no Google *Colab* é grande, apesar de possuir vasta documentação.

Visando dar continuidade no projeto se faz necessários algumas melhorias futuras, como a utilização do *dataset* ou construção do próprio, com imagens rotuladas em português, utilização de outro leitor de texto que possui a compatibilidade com o idioma, bem como um treinamento com maior quantidade de imagens para atingir maior precisão e a implantação do reconhecimento de voz, para iniciar, ler ou encerrar a aplicação.

REFERÊNCIAS

ANACONDA ENTERPRISE. **Anaconda**. Disponível em: <https://www.anaconda.com/distribution/>. Acesso em: 20 set. 2019.

ARAÚJO, F. H.; CARNEIRO, A. C.: SILVA, **R. R. Redes neurias convolucionais com tensorflow**: Teoria e prática.2017. III Escola Regional de Informática do Piauí. Livro Anais - Artigos e Minicursos, v. 1, n. 1, p.382-406. Disponível em: <https://docplayer.com.br/57119969-Redes-neurais-convolucionais-com-tensorflow-teoria-e-pratica.html >

ASTAH. **Guia do usuário.** Disponível em: <http://astah.net/manual>. Acesso em: 18 mar. 2019.

AZEVEDO, Lucas Pereira de. **Aplicação de redes neurais artificiais no processo de classificação de orquídeas do gênero cattleya.** 2016. 37 f. TCC (Graduação) - Curso de Sistemas de Informação, Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais, Sabará, 2016. Disponível em: <https://www.ifmg.edu.br/sabara/biblioteca/trabalhos-de-conclusao-de-curso/tcc-documentos/TCCLucasAzevedo.pdf>. Acesso em: 20 abr. 2019.

AZURE. Microsoft. **O que é computação em nuvem?.** 2019. Disponível em: <https://azure.microsoft.com/pt-br/overview/what-is-cloud-computing/>. Acesso em: 01 set. 2019.

BENTO, Damaris da Silva. **Técnicas de processamento de imagens para reconhecimento e análise da propagação de trincas em chapas de aço.** 2016. 48 f. TCC (Graduação) - Curso de Ciência da Computação, Campus Rio Pomba, Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais, Rio Pomba, 2016. Disponível em: <https://sistemas.riopomba.ifsudestemg.edu.br/.../TCC--Dmaris-Bento.compressed.pdf>. Acesso em: 26 abr. 2019.

BRADSKI, Gary; KAEHLER, Adrian. Opencv Structure and Content. In: BRADSKI, Gary; KAEHLER, Adrian. **Learning OpenCV:** Computer Vision with the OpenCV Library. O'reilly, 2008. Cap. 1. p. 1-15. Disponível em: <https://books.google.com.br/books?hl=pt-BR&lr=&id=seAgiOfu2EIC&oi=fnd&pg=PR3&dq=OpenCV&ots=hUI8dnhGOf&sig=feSAP02nbaR3BGtELFDitJAzDK4#v=onepage&q=OpenCV&f=false>. Acesso em: 26 abr. 2019.

BRASIL. Secretaria de Direitos Humanos da Presidência da República - SEDH. **Tecnologia Assistiva.** Brasília: Comitê de Ajudas Técnicas, 2009. 138 p. CORDE. Disponível em: <https://www.pessoacomdeficiencia.gov.br/app/publicacoes/tecnologia-assistiva>. Acesso em: 21 mar. 2019.

BRIJESH. **TensorFlow Lite.** 2017. Disponível em: <http://androidkt.com/tenserflow-lite/>. Acesso em: 23 abr. 2019.

BRITÂNICA ESCOLA. **Stephen Hawking**. Disponível em: <https://escola.britannica.com.br/artigo/Stephen-Hawking/631067>. Acesso em: 20 set. 2019.

CARVALHO, Hialo Muniz. **Aprendizado de Máquina voltado para Mineração de Dados: Árvores de Decisão.** 2014. 68 f. Monografia (Especialização) - Curso de Engenharia de Software, Faculdade Unb Gama - Fga, Universidade de Brasília - Unb, Brasília, 2014. Disponível em: <https://fga.unb.br/articles/0000/5556/TCC_Hialo_Muniz.pdf>. Acesso em: 23 abr. 2019.

CGEE. Centro de Gestão e Estudos Estratégicos. **Mapeamento de Competências em Tecnologia Assistiva.**2012. ed. Brasília: Cgee, 2012. 449 p. (Relatório Final). Disponível em: <https://edisciplinas.usp.br/pluginfile.php/3378862/mod_resource/content/1/tec%20assistiva.pdf>. Acesso em: 20 mar. 2019.

CHIRIGATI, Fernando Seabra. **Computação em Nuvem**. 2009. Disponível em: <https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2009_2/seabra/arquitetura.html>. Acesso em: 02 set. 2019.

COCO. Common Objects In Context. **What is COCO?** Disponível em: <http://cocodataset.org/#home>. Acesso em: 09 abr. 2019.

COCO. **COCO-Text**. 2019. Disponível em: <https://bgshih.github.io/cocotext/>. Acesso em: 20 set. 2019.

DIAS, Thaiana Lice Lopes; PEREIRA, Liliane Desgualdo. Habilidade de localização e lateralização sonora em deficientes visuais. **Revista da Sociedade Brasileira de Fonoaudiologia**, [s.l.], v. 13, n. 4, p.352-356, 2008. FapUNIFESP (SciELO). Disponível em: <http://www.scielo.br/pdf/rsbf/v13n4/a09v13n4.pdf>. Acesso em: 02 out. 2019.

DEVELOPERS. Estúdio Android. **Conheça o Android Studio.** Disponível em: <https://developer.android.com/studio/intro>. Acesso em: 30 mar. 2019.

DLOLOGY. **Tensorflow object detection:** training colab. 2019. Disponível em: <https://colab.research.google.com/github/Tony607/object_detection_demo/blob/master/tensorflow_object_detection_training_colab.ipynb#scrollTo=GStNeHWPkTcN>. Acesso em: 20 set. 2019.

EDUCABRAS. **VISÃO.** 2019. Disponível em: <https://www.educabras.com/enem/materia/biologia/anatomia_e_fisiologia_2/aulas/visao>. Acesso em: 02 nov. 2019.

FABRÍCIO GUILHERMO. Medium. **A importância dos protótipos para seu aplicativo**. 2019. Disponível em: <https://medium.com/mackmobile/a-

import%C3%A2ncia-da-prototipagem-do-seu-aplicativo-b2ae95f40262>. Acesso em: 20 set. 2019.

FERNANDES, Mariana Abreu. **As implicações de problemas visuais no processo de aprendizagem escolar das crianças.** 2012. 32 f. Dissertação (Mestrado) - Curso de Optometria Ciências da Visão, Ciências da Saúde, Universidade da Beira Interior, Covilhã, 2012. Disponível em: <https://ubibliorum.ubi.pt/bitstream/10400.6/1216/1/Disserta%C3%A7%C3%A3o%20Mariana%20Fernandes.pdf>. Acesso em: 24 abr. 2019.

GIL, Marta (Org.). **Deficiência visual**. Brasília: MEC – Secretaria de Educação a Distância, 2000. 80 p. (Cadernos da TV Escola).Disponível em: <http://portal.mec.gov.br/seed/arquivos/pdf/deficienciavisual.pdf>. Acesso em: 29 Abr. 2019.

GOOGLE. **Colab**. Disponível em: <https://colab.research.google.com/notebooks/welcome.ipynb#recent=true>. Acesso em: 20 set. 2019.

GOOGLE TRENDS. **Machine Learning.** Disponível em: <https://trends.google.com.br/trends/explore?date=today%205-y&q=Machine%20Learning>. Acesso em: 16 abr. 2019.

IBGE. **Censo demográfico 2010:** Características gerais da população, religião e pessoas com deficiência. 2012. Instituto Brasileiro de Geografia e Estatística. Disponível em: <https://www.ibge.gov.br/estatisticas/sociais/populacao/9662-censo-demografico-2010.html?edicao=9749&t=destaques>. Acesso em: 02 abr. 2019.

IBM. **Machine Learning For Dummies.** John Wiley & Sons, Inc., 2018. 68 p. Limited Edition. Disponível em: <https://www.ibm.com/downloads/cas/GB8ZMQZ3>. Acesso em: 8 mar. 2019.

JUPYTER. **Notebook**. 2019. Disponível em: <https://jupyter.org/>. Acesso em: 20 set. 2019.

JOKELA, Jussi. **Person Counter Using Real-time Object Detection And A Small Neural Network**. 2018. 38 f. Tese (Doutorado) - Curso de Information And Communications Technology, Turku University Of Applied Sciences, Turku, 2018. Disponível em: <https://www.theseus.fi/bitstream/handle/10024/153489/Jokela_Jussi.pdf?sequence=1>. Acesso em: 7 mar. 2019.

KLEBER BERNARDO. Cultura Ágil. **Kanban**: Do início ao fim. 2014. Disponível em: <culturaagil.com.br/kanban-do-inicio-ao-fim/>. Acesso em: 20 set. 2019.

KOVALESKI, Patrícia de Andrade. **Implementação De Redes Neurais Profundas Para Reconhecimento De Ações Em Vídeo.** 2018. 48 f. TCC (Graduação) - Curso de Engenharia de Computação e Informação, Escola Politécnica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018. Disponível em:

<http://monografias.poli.ufrj.br/monografias/monopoli10023237.pdf>. Acesso em: 11 abr. 2019.

LUCENA, Oeslle A. S.; VELOSO, Luciana R.; LOPES, Waslon T. A.. Reconhecimento de Objetos em Imagens. In: Encontro Anual Do Iecom Em Comunicações, Redes E Criptografia, 2016, Fortaleza. **Artigo.** Fortaleza: Encom, 2016. p. 1 - 2. Disponível em: <https://iecom.org.br/encom2016/ENCOM_files/Encom2016_Artigos/17.pdf>. Acesso em: 24 mar. 2019.

MARVEL. **Design.** Disponível em: <https://marvelapp.com/features/design>. Acesso em: 22 abr. 2019.

MINISTÉRIO DA SAÚDE. Portaria nº 3.128 de 24 de dezembro de 2008. Define que as Redes Estaduais de Atenção à Pessoa com Deficiência Visual sejam compostas por ações na atenção básica e Serviços de Reabilitação Visual. Brasil, Gabinete do Ministro. Disponível em: <http://bvsms.saude.gov.br/bvs/saudelegis/gm/2008/prt3128_24_12_2008.html>. Acesso em: 25 mar. 2019.

MIYAZAKI, Caio Kioshi. **Redes neurais convolucionais para aprendizagem e reconhecimento de objetos 3D.** 2017. 56 f. Monografia (Especialização) - Curso de Engenharia Elétrica, Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2017. Disponível em: <http://www.tcc.sc.usp.br/tce/disponiveis/18/180500/tce-22022018-121624/publico/Miyazaki_caio_tcc.pdf>. Acesso em: 25 abr. 2019.

OLIVEIRA, Walber Rodrigues de; MELLO, Carlos Alexandre Barros de; OLIVEIRA, Bruno Medeiros de. **Sistema de Visão Computacional para Reconhecimento de Modelos CAD Complexos.** 2018. 45 f. TCC (Graduação) - Curso de Ciência da Computação, Centro de Informática, Universidade Federal de Pernambuco - Ufpe, Recife, 2018. Disponível em: <https://www.cin.ufpe.br/~tg/2018-1/wro-tg.pdf>. Acesso em: 17 abr. 2019
OPENCV. **Introdução**. 2014. Disponível em: <https://docs.opencv.org/master/d1/dfb/intro.html>. Acesso em: 03 abr. 2019.

PAILLARD, T Nóe F(2015).**Techniques and Methods for Testing the Postural Function inHealthy and Pathological Subjects**. BioMed Res Int 2015:1-15. Disponível em: <https://www.researchgate.net/publication/283892682_Techniques_and_Methods_for_Testing_the_Postural_Function_in_Healthy_and_Pathological_Subjects >. Acesso em: 25 abr. 2019.

PRASAD, Dilip Kumar. **Object Detection in Real Images.** 2010. 113 f. Pesquisa (PhD) - Curso de Engenharia de Computação, Nanyang Technological University, Singapura, 2010. Disponível em: <https://arxiv.org/ftp/arxiv/papers/1302/1302.5189.pdf>. Acesso em: 9 abr. 2019.

RAMOS André. **Fisiologia da Visão:** Um estudo sobre o “ver” e o “enxergar” Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio 2006. Disponível em: < http://web.unifoa.edu.br/portal/plano_aula/arquivos/04054/Fisiologia%20da%20visao %20-%20MODULO%20I.pdf>.

SANTOS JÚNIOR, Zulmar Jofli dos. **A acessibilidade como veículo de inclusão social: proposta de dispositivo computacional para os deficientes visuais da cidade de Natal/RN.** 2009. 78 f. Tese (Mestrado) - Curso de Engenharia de Produção, Centro de Tecnologia, Universidade Federal do Rio Grande do Norte, Natal, 2009. Disponível em: <https://repositorio.ufrn.br/jspui/bitstream/123456789/14909/1/ZulmarJSJ.pdf>. Acesso em: 5 abr. 2019

SBC. **Machine Learning: Desafios para um Brasil competitivo.** Porto Alegre: Computação Brasil, v. 39, 2019. Disponível em: <http://www.sbc.org.br/images/flippingbook/computacaobrasil/computa_39/pdf/CompBrasil_39_180.pdf>. Acesso em: 25 abr. 2019.

SOUSA, Flávio & MOREIRA, Leonardo & MACHADO, Javam. (2019). **Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios**. 1Publicado no ERCEMAPI 2009. Todos os direitos reservados a EDUFPI. 2Versão revisada e estendida em Setembro de 2010. Universidade Federal do Ceará (UFC) cap 7 26 pgs. Disponível em: <https://www.researchgate.net/profile/Javam_Machado/publication/237644729_Computacao_em_Nuvem_Conceitos_Tecnologias_Aplicacoes_e_Desafios/links/56044f4308aea25fce3121f3.pdf>

TENSORFLOW. **Introduction to TensorFlow.** Disponível em: <https://www.tensorflow.org/learn>. Acesso em: 05 abr. 2019.

TENSORFLOW MODELS. Models: **Object Detection**. GitHub. Disponível em: <https://github.com/tensorflow/models/tree/master/research/object_detection>. Acesso em: 15 out. 2019.

THEO, John. **Google Collaboratory Overview.** 2018. Disponível em: <https://medium.com/johntheology/google-collaboratory-overview-2fc6a969dd5f>. Acesso em: 26 set. 2019.

TRELLO. **Conheça o Trello.** 2015. Disponível em: <https://trello.com/c/Bbpc1cRl/2-o-que-%C3%A9-trello>. Acesso em: 19 mar. 2019.