



UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
PRÓ-REITORIA DE GRADUAÇÃO
DEPARTAMENTO DE CIÊNCIAS EXATAS E NATURAIS
CURSO DE CIÊNCIA DA COMPUTAÇÃO

JOSÉ CARLOS DA SILVA

RECONHECIMENTO DE INDIVÍDUOS EM VÍDEO UTILIZANDO ESTIMATIVA DE POSE HUMANA

MOSSORÓ

2020

JOSÉ CARLOS DA SILVA

RECONHECIMENTO DE INDIVÍDUOS EM VÍDEO UTILIZANDO ESTIMATIVA DE POSE HUMANA

Monografia apresentada à Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Daniel Faustino Lacerda de Souza, Prof. Dr.

MOSSORÓ

2020

© Todos os direitos estão reservados a Universidade Federal Rural do Semi-Árido. O conteúdo desta obra é de inteira responsabilidade do (a) autor (a), sendo o mesmo, passível de sanções administrativas ou penais, caso sejam infringidas as leis que regulamentam a Propriedade Intelectual, respectivamente, Patentes: Lei nº 9.279/1996 e Direitos Autorais: Lei nº 9.610/1998. O conteúdo desta obra tomar-se-á de domínio público após a data de defesa e homologação da sua respectiva ata. A mesma poderá servir de base literária para novas pesquisas, desde que a obra e seu (a) respectivo (a) autor (a) sejam devidamente citados e mencionados os seus créditos bibliográficos.

S586r Silva, José Carlos da.
Reconhecimento de indivíduos em vídeo utilizando
estimativa de pose humana / José Carlos da Silva.
- 2020.
52 f. : il.

Orientador: Daniel Faustino Lacerda de Souza.
Monografia (graduação) - Universidade Federal
Rural do Semi-árido, Curso de Ciência da
Computação, 2020.

1. OpenPose. 2. Redes Neurais Artificiais.. 3.
OpenCV. 4. Reconhecimento de Indivíduos. 5. Visão
Computacional. I. Souza, Daniel Faustino Lacerda
de, orient. II. Título.

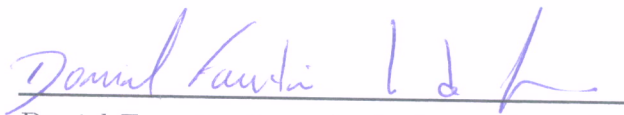
JOSÉ CARLOS DA SILVA

RECONHECIMENTO DE INDIVÍDUOS EM VÍDEO UTILIZANDO ESTIMATIVA DE POSE HUMANA

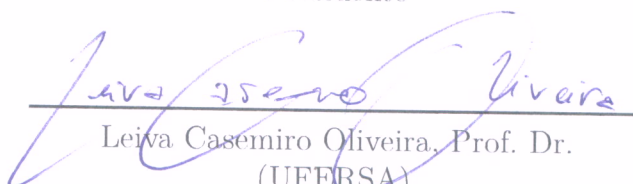
Monografia apresentada à Universidade
Federal Rural do Semi-Árido como requisito
para obtenção do título de Bacharel em
Ciência da Computação.

Defendida em: 06 / 02 / 2020.

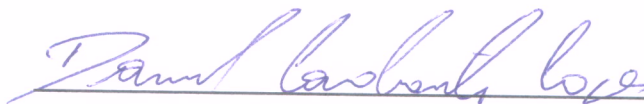
BANCA EXAMINADORA



Daniel Faustino Lacerda de Souza, Prof. Dr.
(UFERSA)
Presidente



Leiva Casemiro Oliveira, Prof. Dr.
(UFERSA)
Membro Examinador



Danniell Cavalcante Lopes, Prof. Dr.
(UFERSA)
Membro Examinador

MOSSORÓ

2020

Este trabalho é dedicado ao meu pai, o saudoso Sr. João Roseno da Silva, e à minha mãe, a Sra. Maria de Lourdes da Silva, que mesmo passando por muitas dificuldades me deram a oportunidade de estudar e ter uma vida melhor.

Agradecimentos

Agradeço primeiro ao Senhor meu Deus, que me deu a força, o discernimento e os recursos necessários para que esse sonho se tornasse realidade.

Agradeço também a minha esposa, Paula Ellen Fernandes Silva, que me prestou todo o apoio necessário durante a minha vida acadêmica.

Agradeço aos meus colegas de curso que me ajudaram quando eu passei por algumas dificuldades em continuar estudando, em especial a Ademar Neto e Ramiro Júnior, que sempre me incentivaram. Também agradeço ao colega Ítalo Resende que foi meu parceiro de estudos durante muito tempo.

E finalizando, agradeço aos meus professores, que durante a minha passagem pelo curso, acreditaram em mim e me incentivaram, especialmente o professor Hércio Wagner da Silva. Agradeço também ao meu professor orientador, Daniel Faustino Lacerda de Souza, pela excelente tutoria que me prestou durante o desenvolvimento deste trabalho e nas demais disciplinas que cursei com ele.

Resumo

A necessidade de identificar pessoas sempre esteve presente na sociedade, sendo utilizada em diversas áreas, das quais podemos citar a indústria de segurança como destaque. Essa tarefa se tornou mais rápida devido ao uso da computação, o que fez a sua utilização ser ampliada, estando o reconhecimento facial entre as técnicas mais precisas e utilizadas. Embora seja eficaz o reconhecimento facial requer imagens em alta resolução e com boa iluminação, além de precisar que, na maioria das aplicações, a face seja capturada de frente. Esses requisitos nem sempre são alcançados e, para muitas situações, precisamos recorrer a outras técnicas para identificação de indivíduos, utilizando outras partes do corpo. Neste trabalho abordamos a possibilidade de reconhecer indivíduos em imagens e vídeos, utilizando alguns pontos-chave do seu corpo, fornecidos por uma biblioteca de OpenPose. O objetivo é projetar uma Rede Neural Artificial capaz de classificar os indivíduos presentes em uma imagem utilizando a estrutura corporal formada por esses pontos. A metodologia usada consiste na definição de algumas relações de proporção entre as partes do corpo, calculadas a partir da distância entre os pontos-chave, de modo que se possa criar um modelo de classificação consistente, com um conjunto de pesos capaz de reconhecer um indivíduo. Os testes realizados apresentaram bons resultados para indivíduos que tiveram seu classificador treinado, reconhecendo estes com precisão de mais 90%. Em testes com imagens de indivíduos que não tinham classificador treinado na rede, a taxa de erros de classificação foi muito alta, onde um indivíduo desconhecido era classificado como um indivíduo que possui classificador treinado.

Palavras-chave: OpenPose. Redes Neurais Artificiais. OpenCV. Reconhecimento de Indivíduos. Visão Computacional.

Abstract

The need to identify people has always been present in society, being used in several areas, of which we can mention the security industry as a highlight. This task has become faster due to the use of computing, which has increased its use, with facial recognition being among the most accurate and used techniques. Although facial recognition is effective, it requires high resolution images with good lighting, in addition to requiring that, in most applications, the face be captured from the front. These requirements are not always met and, for many situations, we need to resort to other techniques for identifying individuals, using other parts of the body. In this work we discuss the possibility of recognizing individuals in images and videos, using some key points of their body, provided by an OpenPose library. The goal is to design an Artificial Neural Network capable of classifying the individuals present in an image using the body structure formed by these points. The methodology used consists in the definition of some proportion relations between the parts of the body, calculated from the distance between the key points, so that a consistent classification model can be created, with a set of weights capable of recognizing an individual. The tests performed showed good results for individuals who had their classifier trained, recognizing these with an accuracy of over 90 %. In tests with images of individuals who did not have a trained classifier on the network, the rate of classification errors was very high, where an unknown individual was classified as an individual with trained classifier.

Keywords: OpenPose. Artificial neural networks. OpenCV. Individual Recognition. Computer vision.

Lista de ilustrações

Figura 1 – Arquitetura da rede OpenPose (Learn OpenCV, 2018)	19
Figura 2 – Mapa de confiança para o ombro esquerdo (Learn OpenCV, 2018)	20
Figura 3 – Relação de afinidade de partes entre ombro esquerdo e pescoço (Learn OpenCV, 2018)	20
Figura 4 – Keypoints COCO e MPII, respectivamente (autoria própria)	22
Figura 5 – Modelo de neurônio artificial (http://www.scielo.br)	24
Figura 6 – Gráfico da função limiar. Equação 3.3.2 (autoria própria)	24
Figura 7 – Gráfico da função linear por partes. Equação 3.3.3 (autoria própria) . .	25
Figura 8 – Gráfico da função sigmóide. Equação 3.3.4 (autoria própria)	25
Figura 9 – Gráfico da função relu. Equação 3.3.5 (autoria própria)	25
Figura 10 – Arquitetura de camada única (https://www.embarcados.com.br/redes-neurais-artificiais/)	26
Figura 11 – Arquitetura multicamadas (https://www.embarcados.com.br/redes-neurais-artificiais/)	26
Figura 12 – Arquitetura Recorrente (https://www.embarcados.com.br/redes-neurais-artificiais/)	27
Figura 13 – Gráfico de utilização de frameworks de Deep Learning em 2018 (Keras, 2019)	30
Figura 14 – Homem Vitruviano de Leonardo da Vinci, 1490 (Wikipedia.org,)	31
Figura 15 – Partes do corpo formadas pelos keypoints extraídos da biblioteca de Openpose (autoria própria)	32
Figura 16 – Distância entre os pontos A e B (autoria própria)	33
Figura 17 – Medidas do corpo (autoria própria)	34
Figura 18 – Distribuição das camadas do Classificador, adaptada de (deep-learning-com-python-az-curso-completo)	36
Figura 19 – Etapas da aplicação (autoria própria)	37
Figura 20 – Interface da aplicação (autoria própria)	38
Figura 21 – Seleção do Classificador treinado (autoria própria)	38
Figura 22 – Seleção da imagem de entrada (autoria própria)	39
Figura 23 – Exibição de vídeo de reconhecimento, (autoria própria)	39
Figura 24 – Processo de geração de dados para treinamento (autoria própria) . . .	40
Figura 25 – Dados de um arquivo CVS com as proporções e rótulos para treinamento (autoria própria)	40
Figura 26 – Informações estatísticas dos dados de um arquivo CVS (autoria própria)	41
Figura 27 – Imagens com poses de treinamento (autoria própria)	42
Figura 28 – Etapas do reconhecimento de indivíduos (autoria própria)	43

Figura 29 – Reconhecimento Zeca (autoria própria)	45
Figura 30 – Reconhecimento de Alamo e Zeca (autoria própria)	45
Figura 31 – Reconhecimento de Antony e Zeca (autoria própria)	45
Figura 32 – Reconhecimento de Paulinha e Zeca (autoria própria)	46
Figura 33 – Reconhecimento de Pablo, sem modelo treinado (autoria própria) . . .	46
Figura 34 – Gráfico da linha do tempo de um vídeo com os indivíduos Alamo e Zeca (autoria própria)	48
Figura 35 – Gráfico da linha do tempo de um vídeo com os indivíduos Antony e Zeca (autoria própria)	48
Figura 36 – Gráfico da linha do tempo de um vídeo com o indivíduo Pablo (autoria própria)	48
Figura 37 – Gráfico da linha do tempo de um vídeo com Paulinha (autoria própria)	49
Figura 38 – Gráfico da linha do tempo de um vídeo com o indivíduo Zeca (autoria própria)	49

Lista de tabelas

Tabela 1	– Pontos-chave para os modelos COCO e MPII (autoria própria)		21
Tabela 2	– Relações de proporção entre as partes do corpo (autoria própria)	. . .	35
Tabela 3	– Parâmetros de configuração da rede neural (autoria própria)		36
Tabela 4	– Análise das linhas do tempo dos vídeos processados. (autoria própria)	.	46

Lista de abreviaturas e siglas

JSON	Javascript Object Notation, ou Notação de Objetos do Javascript
XML	Extensible Markup Language (linguagem de marcação)
YAML	Yet Another Markup Language (linguagem de marcação)
GPU	Graphics Processing Unit
OpenCV	Open Source Computer Vision Library
MLP	Multi Layer Perceptron
RBF	Radial Basis Function
LMS	Least Mean Square
SGD	Stochastic Gradient Descent
API	Application Programming Interface
CNTK	Computational Network Toolkit
CPU	Central Processing Unit

Lista de símbolos

φ	Letra grega Fi
Σ	Somatório
η	Letra grega Eta

Sumário

1	INTRODUÇÃO	14
1.1	Objetivos	15
1.2	Organização	15
2	TRABALHOS CORRELATOS	17
3	REFERENCIAL TEÓRICO	19
3.1	OpenPose	19
3.2	OpenCV	22
3.3	Redes Neurais Artificiais	23
3.3.1	O neurônio artificial	24
3.3.2	Arquitetura	26
3.3.3	Aprendizagem	27
3.3.4	Keras API	29
4	MÉTODO	31
4.1	Definições	31
4.2	Classificador	35
4.2.1	Aplicação	37
4.2.2	Geração de dados	39
4.2.3	Treinamento	40
4.2.4	Reconhecimento	42
5	RESULTADOS	44
6	CONCLUSÃO	50
	REFERÊNCIAS	52

1 INTROUÇÃO

Reconhecer pessoas é sem dúvida uma necessidade da sociedade moderna, e isso se deve a fatores que vão desde o simples entretenimento pessoal até questões de segurança pública. Existem muitas técnicas para reconhecimento de indivíduos: impressão digital, linhas da palma da mão, geometria da mão, reconhecimento de íris ocular, reconhecimento facial, etc. Essas técnicas são baseadas na biometria do corpo e conseguem identificar uma pessoa, pois esses traços, para algumas delas, se mantêm por toda vida toda do indivíduo. Com o auxílio da computação essas técnicas ganharam precisão e rapidez, ampliando consideravelmente as suas aplicações. Dentre elas, o reconhecimento facial ganha destaque por não haver necessidade de colaboração do indivíduo, podendo ser realizado à distância do mesmo e com custo um pouco menor que outras técnicas. Para as demais se exige uso de algum equipamento que requer proximidade maior, ou necessite que a pessoa submeta a parte do corpo em questão para análise, se fazendo necessário que a pessoa que será identificada colabore. Além disso alguns desses equipamentos, como leitores de impressão digital, por exemplo, costumam ser caros, não sendo portanto uma boa alternativa quando se pensa em custo e benefício.

O reconhecimento de indivíduos por meio de sistemas computacionais faz parte da sociedade moderna, sendo muito utilizado, com destaque nas áreas de:

- segurança: aplicação em sistemas de vigilância para empresas privadas e órgãos públicos, sistemas usados em perícia forense, acesso à áreas físicas em instalações diversas, acesso a dispositivos mobile, acesso a serviços do governo, como emissão de documentos virtuais (habilitação, carteira de trabalho, carteira de estudante), etc;
- entretenimento: dispositivos mobile, com aplicações para redes sociais, etc;
- aplicações comerciais: empresas utilizam essa tecnologia para permitir acesso às suas lojas e processar pagamentos.

Esses sistemas utilizam inteligência artificial para analisar padrões presentes na fisionomia dos indivíduos e reconhecer os mesmos posteriormente. Devido ao seu alto poder de processamento e generalização, as redes neurais artificiais são frequentemente utilizadas em tarefas de classificação e reconhecimento de padrões. Uma tecnologia de identificação de pessoas consolidada e bastante utilizada é o reconhecimento facial. Ela utiliza padrões presentes na face para reconhecer indivíduos e, embora eficaz, essa tecnologia precisa de imagens com boa resolução e também depende, em muitos casos, da posição da face, que

deve estar posicionada de frente. Sendo assim, em situações com iluminação e resolução de imagem ruins, o reconhecimento facial não é possível.

1.1 Objetivos

Devido à limitação do reconhecimento facial, e à necessidade de identificação de pessoas presente nos dias atuais, surgiu a necessidade de buscar novas alternativas para reconhecer indivíduos em imagens e vídeos. Nesse sentido, temos como objetivos deste trabalho:

1. Investigar técnicas e ferramentas capazes de permitir o reconhecimento de pessoas em função de características que as identifique unicamente, apoiados no uso de algoritmos baseados em redes neurais para a classificação desses indivíduos;
2. Utilizar a OpenPose, que é uma biblioteca de estimativa de pose humana que fornece os pontos-chave do corpo e o grau de confiança entre estes;
3. Implementar uma rede neural capaz de processar esses dados e reconhecer indivíduos utilizando outras partes do corpo, em conjunto;
4. Estabelecer relações de proporção entre as partes do corpo formadas pelos keypoints fornecidos pela OpenPose, a fim de utilizá-las como parâmetros de entrada da rede

1.2 Organização

Iniciamos este trabalho no capítulo 1, na seção de Introdução, apresentando o problema das limitações na técnica de reconhecimento facial, enumerando algumas dessas limitações e sugerindo a busca de novas alternativas à essa técnica, finalizando esta seção com uma proposta de método para reconhecer indivíduos em vídeos e imagens.

Em seguida, no capítulo 2, relacionamos alguns trabalhos, que assim como este, buscam novas formas de identificar indivíduos. Alguns autores utilizaram características biométricas presentes na geometria das mãos, outros recorreram a padrões presentes na forma de caminhar e no deslocamento dos indivíduos, e por fim, alguns combinaram técnicas para obter maior precisão no reconhecimento de indivíduos.

Mais adiante, no capítulo 3, abordamos os conceitos teóricos necessários para embasar o método proposto, bem como a apresentação das tecnologias e ferramentas utilizadas no processo de criação do nosso protótipo.

No capítulo 4 definindo as diretrizes do método utilizado neste trabalho, mostramos como foi criado o classificador utilizado e o protótipo de aplicação para validação do método.

Na sequência o capítulo 5 nos apresenta os resultados obtidos e a relação entre eles, mostrando a forma como foram gerados e analisados.

No capítulo 6 concluímos o trabalho com base na análise dos resultados e sugerimos algumas observações para trabalhos futuros, como forma de aprimorar o método. E por fim, apresentamos as referências bibliográficas utilizadas como fonte de pesquisa.

2 TRABALHOS CORRELATOS

Existem vários trabalhos que abordam o reconhecimento de indivíduos em imagens utilizando padrões presentes na biometria do corpo.

Características biométricas leves (SOUZA; MARANA, 2004), como altura, cor da pele e cabelo, dentre outras, podem auxiliar na identificação de indivíduos, e assim obter resultados mais rápidos e mais precisos para buscas em grandes bases de dados. Aqui o autor utiliza essas características apenas como forma auxiliar para outras técnicas de identificação, não sendo possível utilizá-las separadamente. Se ele usasse mais parâmetros poderia identificar pessoas sem depender de outras técnicas, apenas combinando tais parâmetros.

No trabalho de Vieira (2009), foi usado um classificador para identificar pessoas pela dinâmica do caminhar, relatando que a forma de caminhar de uma pessoa difere de outras, sendo portanto um padrão atrativo para sistemas de classificação, combinado ou não com outros classificadores. Neste trabalho a autora utiliza a velocidade como parâmetro para a forma caminhar, afirmando que algumas pessoas andam devagar e outras mais rapidamente, o que pode não ser adequado, já que essa velocidade pode depender da ocasião em que a pessoa se encontra, e portanto essa técnica não deve ser utilizada de forma isolada, e sim com outro tipo de classificador. Os resultados poderiam ser melhores se a autora combinasse essa velocidade com a distância das passadas. Já em (NASCIMENTO, 2015), foi utilizada pelo autor a geometria das mãos para identificar indivíduos, extraindo delas 84 atributos que foram usados para reconhecimento, obtendo 100% de precisão em diferentes estratégias de classificação. Ele conseguiu aqui parâmetros de boa qualidade, explorando todas as medidas possíveis, o que refletiu diretamente nos seus resultados.

Em (TAKEDA; et al., 2012) os autores utilizaram um dispositivo Kinect para mapear 20 pontos do corpo humano (semelhantes aos da biblioteca de OpenPose), e monitorar os seus deslocamentos no espaço de coordenadas, com o objetivo de identificar o padrão biométrico do movimento desses indivíduos e identificá-los posteriormente, obtendo taxas de acerto que variam de 70% a 96%, dependendo do método utilizado para classificação dos indivíduos. Este trabalho, por também se utilizar de pontos que mapeiam o corpo humano, é o que mais se assemelha ao proposto neste trabalho acadêmico, porém aqui utilizamos a distância dos pontos apenas para definir medidas do corpo.

Em seu trabalho, Matos (2011) também utiliza a geometria das mãos para identificar indivíduos, porém com métodos que analisam as pontas dos dedos e os vales entre estes, chegando a obter taxas 76% de acerto utilizando essa técnica, a qual considera viável já que a aquisição dos dados é de baixo custo se comparado a outras alternativas de

reconhecimento biométrico como impressão digital e iris. A aquisição dos dados proposta aqui pelo autor, embora melhor que outras citadas por ele em seu trabalho, ainda não é adequada, pois além de precisar da colaboração do indivíduo, precisa ainda que este ponha a mão em uma posição pré-definida, o que é passível de erros caso este queira burlar o processo de identificação. O autor ([FREITAS, 2014](#)) utilizou limiarização adaptativa para extrair dados biométricos das linhas da palma da mão, concluindo que essa técnica obteve precisão de 98,5% em seus testes.

Em ([SALOMAO, 2007](#)) é proposto utilizar um sistema bi-modal de identificação, baseado na face e na forma de caminhar de indivíduos, para fazer o reconhecimento de pessoas em vídeos, validando sua tese com obtenção de taxas de acerto com precisão de 92,86%. Este autor utiliza em seu trabalho mais parâmetros para distinguir a forma de caminhar, não somente a velocidade, e isso já confere mais precisão ao seu método. Ao combinar essas características com os padrões já aceitos de reconhecimento facial pode se produzir uma ferramenta de identificação promissora. E, por fim, no trabalho de ([FIGUEIREDO, 2018](#)), foi utilizada OpenPose aliada ao reconhecimento facial para rastrear pedestres e reconhecê-los, buscando melhorar a interação de robôs com seres humanos, onde obteve taxas de precisão que variaram de 80% a 98%. Apesar de também estar utilizando OpenPose, esse autor a usa apenas para localizar uma pessoa, de modo que o robô saiba que existe um indivíduo em seu ambiente, porém o reconhecimento de fato, é feito por meio da face. Pretendemos utilizá-la aqui para extração das medidas do corpo, de modo a se obter uma nova forma de identificação de pessoas.

Todos esses autores conseguiram boas taxas de precisão com seus métodos, mostrando que se pode utilizar outros caminhos para identificar pessoas. No capítulo 3 adiante este trabalho aborda os conceitos teóricos necessário para implementação do método proposto aqui.

3 REFERENCIAL TEÓRICO

3.1 OpenPose

OpenPose é uma biblioteca para estimativa de pose humana multi-pessoas (HIDALGO; et al., 2019) desenvolvida por Gines Hidalgo, Zhe Chao, e outros colaboradores do CMU-Perceptual-Computing-Lab. Trata-se de uma biblioteca multiplataforma, que pode fazer uso de GPU para aceleração e obtenção de respostas em tempo real. Ela dispõe de funcionalidades que vão desde a calibração da câmera até a obtenção da triangulação de pontos 3D. Também possibilita salvar as imagens com os pontos-chave (cotovelos, joelhos, pescoço, ombro, quadris, tórax, etc) sobrepostos em vários formatos de imagem e vídeo, além de salvar os pontos em arquivos nos formatos JSON, XML, YML, etc, para utilizações em outras aplicações. A biblioteca utiliza modelos com 15, 18 e 25 pontos para o corpo, além de outros conjuntos de pontos para os pés, mãos e face. Foram utilizados os modelos de 15 (Base MPII - Multi Person Dataset) e 18 (Base COCO - Common Objects in Context) pontos para implementação do método proposto no capítulo 4.

A OpenPose utiliza modelos treinados em uma rede VGG19(Learn OpenCV, 2018), cuja a arquitetura é mostrada na Figura 1, tendo como entrada uma imagem colorida de três dimensões, e como saída os locais 2D de pontos chave para cada pessoa na imagem, conforme etapas a seguir:

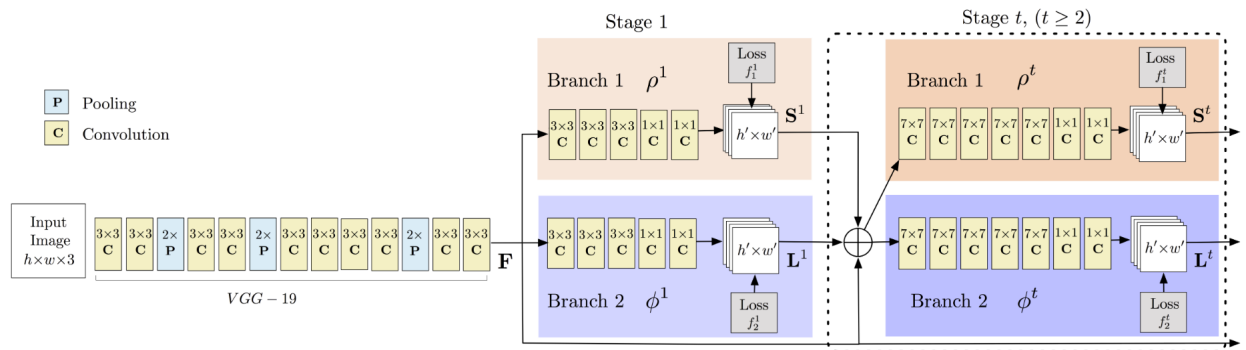


Figura 1 – Arquitetura da rede OpenPose (Learn OpenCV, 2018)

- Estágio 0 – As 10 primeiras camadas criam os mapas de recursos;
- Estágio 1 – duas ramificações preveem um conjunto 2D de mapas de confiança (S) das localizações das partes do corpo(cabeça, pescoço, etc) e um conjunto de vetores 2D para o mapa de afinidade das partes (L); e

- Estágio 2 – Os mapas de confiança e afinidade são analisados por inferência para produzir os pontos-chave para todas as pessoas na imagem.

Abaixo, na Figura 2, temos a representação do mapa de confiança para a região do ombro esquerdo, e na Figura 3, vemos a representação da afinidade de partes entre o ombro esquerdo e o pescoço.



Figura 2 – Mapa de confiança para o ombro esquerdo ([Learn OpenCV, 2018](#))

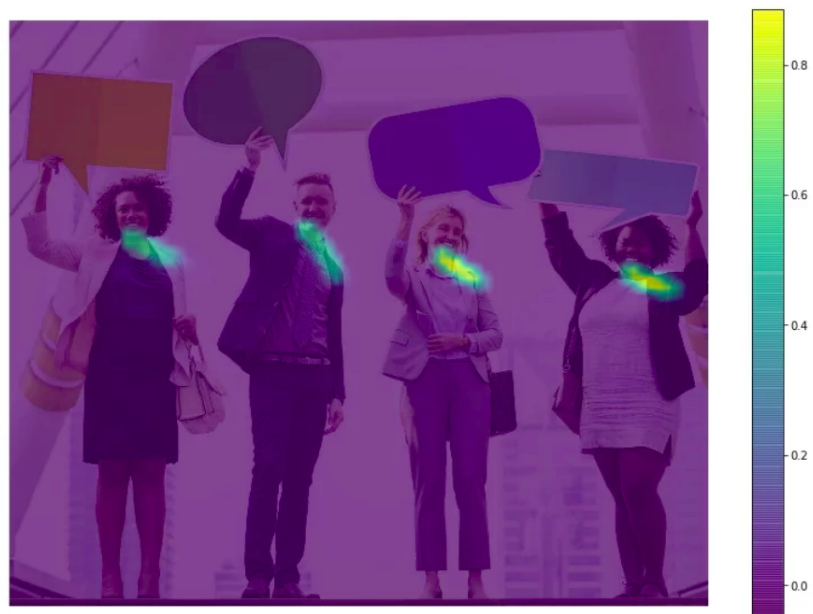


Figura 3 – Relação de afinidade de partes entre ombro esquerdo e pescoço ([Learn OpenCV, 2018](#))

A saída da rede é composta por:

- a) ID da imagem;
- b) índice do pontos-chave, sendo que para o modelo COCO temos 57 partes(18 mapas de confiança de pontos-chave + 1 plano de fundo + 19*2 mapas de afinidade de partes) e para o modelo MPII temos apenas 44;
- c) altura do mapa de saída; e
- d) largura do mapa de saída.

Os arquivos treinados dos modelos COCO e MPII, podem ser baixados no repositório do Github¹, sendo cada modelo dividido em dois arquivos:

- .prototxt – especifica a arquitetura da rede, organização das camadas e parâmetros;
- .caffemodel – armazena os pesos do modelo treinado.

Os ponto-chave dos dois modelos estão distribuídos na Tabela 3.1 abaixo:

Tabela 1 – Pontos-chave para os modelos COCO e MPII (autoria própria)

Dados	keypoints	
COCO	0-Nariz	1-Pescoço
	2-Ombro direito	3-Cotovelo direito
	4-Pulso direito	5-Ombro esquerdo
	6-Cotovelo esquerdo	7-Pulso esquerdo
	8-Quadril direito	9-Joelho direito
	10-Tornozelo direito	11-Quadril esquerdo
	12-Joelho esquerdo	13-Tornozelo esquerdo
	14-Olho direito	15-Olho esquerdo
	16-Orelha direita	17-Orelha esquerda
	18-Fundo	
MPII	0-Cabeça	1-Pescoço
	2-Ombro direito	3-Cotovelo direito
	4-Pulso direito	5-Ombro esquerdo
	6-Cotovelo esquerdo	7-Pulso esquerdo
	8-Quadril direito	9-Joelho direito
	10-Tornozelo direito	11-Quadril esquerdo
	12-Joelho esquerdo	13-Tornozelo esquerdo
	14-Tórax	15-Fundo

Abaixo a Figura 4 mostram esses pontos dispostos sobre um fundo preto formando os esqueletos da pose:

¹ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master>

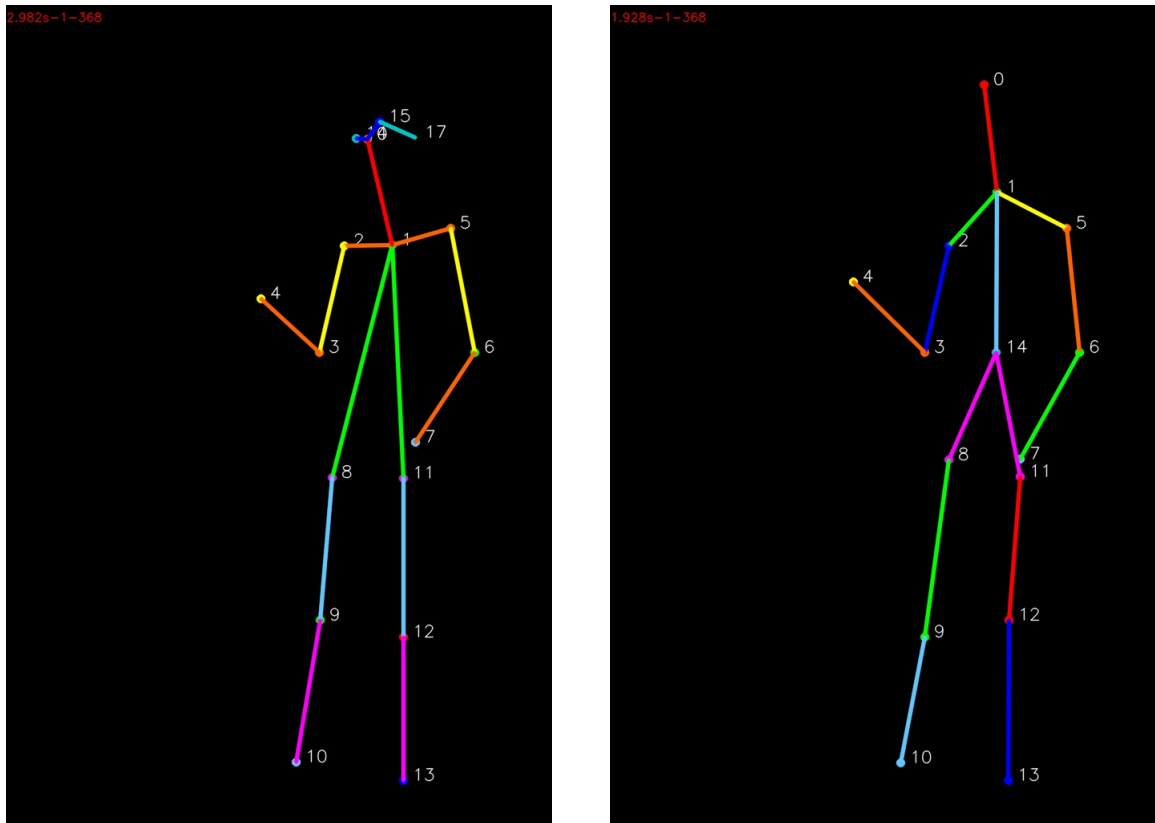


Figura 4 – Keypoints COCO e MPII, respectivamente (autoria própria)

3.2 OpenCV

OpenCV (Open Source Computer Vision Library) é uma biblioteca multiplataforma desenvolvida pela Intel em 2000, com licença BSD de código aberto que dispõe centenas de algoritmos de visão computacional ([OpenCV.org](https://opencv.org), 2019) organizados em estrutura modular, com funcionalidades que englobam desde a calibração da câmera e aquisição da imagem até processamento de operações sobre as imagens e vídeos, sendo possível aplicar filtros de correção de ruído, aplicar convoluções para extração de características específicas de uma imagem, detectar objetos e criar redes de aprendizado profundo, as quais podem utilizar GPU para aceleração de seu processamento.

Essa biblioteca é utilizada para aplicações de visão computacional, em áreas como:

- robótica - robôs para atividades que oferecem risco a humanos, etc;
- indústria automobilística - carros autônomos, aplicação de multas de trânsito e verificação da situação de veículos pelos órgãos regulatórios, etc;
- indústria agropecuária - seleção de sementes, estudo de áreas para plantio, manejo de animais, etc;

- medicina – análise e correção de imagens para detecção e diagnóstico de doenças, etc;
- aplicações militares – obtenção de alvos, estudo de áreas inimigas, etc;
- segurança – reconhecimento facial em câmeras de segurança, análises de placas de veículos em portões automáticos de condomínios, etc; e
- Redes sociais – aplicação de filtros diversos, correções e melhorias em imagens adquiridas com baixa qualidade, etc.

A OpenCV é uma ferramenta que dispõe de vasta documentação em seu site oficial, com exemplos de utilização em várias linguagens de programação, com ênfase para C/C++, Java e Python. Além da documentação oficial temos muitas comunidades ativas na internet que ajudam novos utilizadores a implementarem seus primeiros códigos. Frequentemente encontramos exemplos em Python, já que esta linguagem é muito usada para tratar problemas que exigem visão computacional, devido à forma rápida e eficaz que a mesma trabalha com listas e matrizes de qualquer natureza e dimensionalidade. No desenvolvimento da aplicação deste trabalho também utilizaremos essa linguagem devido à sua facilidade de integração com a OpenCV e ao dinamismo com que lida com as estruturas de dados mencionadas anteriormente.

3.3 Redes Neurais Artificiais

O trabalho em redes neurais artificiais é motivado pela forma como o cérebro humano processa informações, sendo que ele consegue organizar seus neurônios de forma a processar certas informações muito mais rápido que um computador. [Haykin \(2001, p. 28\)](#) define rede neural como um processador massivo paralelamente distribuído de unidades simples de processamento, que têm propensão natural para armazenar conhecimento experimental e torná-lo disponível para uso, se assemelhando ao cérebro humano em dois aspectos:

1. O conhecimento é adquirido pela rede de seu ambiente através de um processo de aprendizagem.
2. Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido.

Assim, devido ao poder de processamento, através de sua estrutura paralela, e à sua capacidade de generalização para novos dados, por meio do processo de aprendizagem, as

redes neurais conseguem resolver problemas complexos, com dados linearmente separáveis ou não, e por isso são largamente utilizadas para classificação e reconhecimento de padrões.

3.3.1 O neurônio artificial

Baseado no modelo do neurônio do cérebro humano e nas suas ligações sinápticas, foi desenvolvido um modelo de neurônio artificial para compor as redes neurais. Esse neurônio é a unidade básica de formação dessas redes e temos sua representação mostrada na Figura 5.

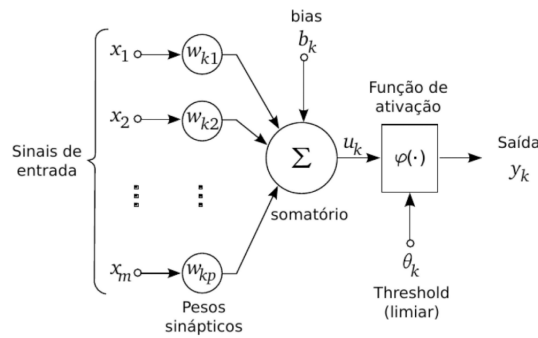


Figura 5 – Modelo de neurônio artificial (<http://www.scielo.br>)

Dessa forma temos um conjunto x de valores de entrada, sobre os quais atua um conjunto w de pesos sinápticos, passando por uma função de ativação $\varphi(x)$, que definirá o seu valor de saída y , onde $k_0 = b$ (bias com valor fixo), conforme a Equação 3.3.1 abaixo:

$$y = \varphi\left(\sum_{k=0}^n x_k \times w_k\right) \quad (3.3.1)$$

Existem várias funções de ativação $\varphi(x)$, mas mostramos aqui apenas alguns tipos básicos (HAYKIN, 2001), definidas nas Equações abaixo, cujo gráficos são mostrados nas Figuras 6, 7 e 8 e 9 respectivamente.

$$\text{Limiar, } \varphi(x) = \begin{cases} 1 & \text{se } x \geq 0 \\ 0 & \text{se } x < 0 \end{cases} \quad (3.3.2)$$

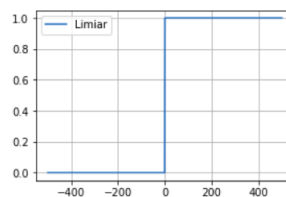


Figura 6 – Gráfico da função limiar. Equação 3.3.2 (autoria própria)

$$\text{Linear por partes, } \varphi(x) = \begin{cases} 1, & x \geq +\frac{1}{2} \\ x, & -\frac{1}{2} < x < +\frac{1}{2} \\ 0 & x \leq -\frac{1}{2} \end{cases} \quad (3.3.3)$$

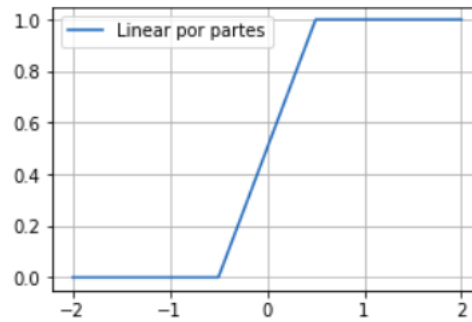


Figura 7 – Gráfico da função linear por partes. Equação 3.3.3 (autoria própria)

$$\text{Sigmóide } \varphi(x) = \frac{1}{1 + \exp(-x)} \quad (3.3.4)$$

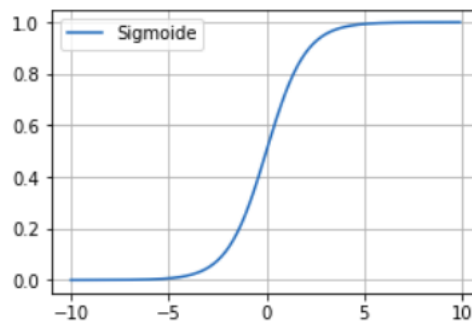


Figura 8 – Gráfico da função sigmóide. Equação 3.3.4 (autoria própria)

$$\text{e Relu, } \varphi(x) = \max(0, x) \quad (3.3.5)$$

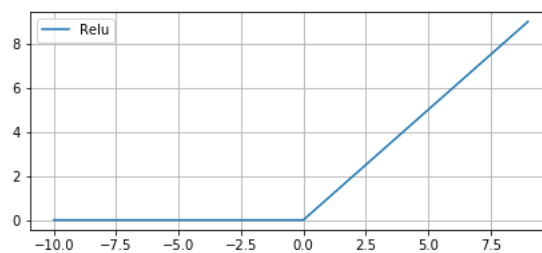


Figura 9 – Gráfico da função relu. Equação 3.3.5 (autoria própria)

3.3.2 Arquitetura

A arquitetura é a forma como os neurônios estão organizados. Ela está fortemente relacionada com o algoritmo de aprendizagem utilizado para o treinamento da rede (HAYKIN, 2001) e em geral temos três tipos fundamentais:

1. **Camada única**, tendo a camada de entrada diretamente conectada à camada de saída (ex. Perceptron, Adaline), como mostra a Figura 10. Essa arquitetura é usada para problemas com classes linearmente separáveis.

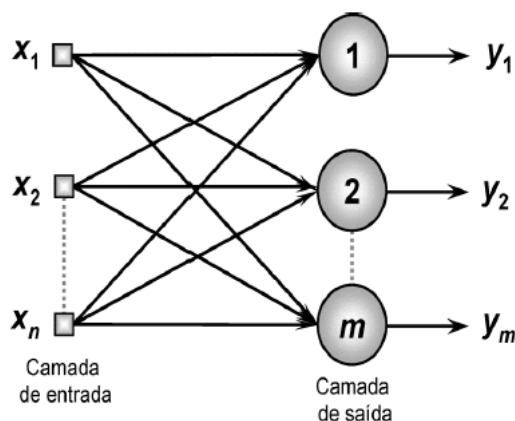


Figura 10 – Arquitetura de camada única (<https://www.embarcados.com.br/redes-neurais-artificiais/>)

2. **Múltiplas camadas**, com uma ou mais camadas ocultas, conectadas entre a entrada e a saída (ex. MLP, RBF). Essas camadas ocultas dão à rede uma perspectiva global e a tornam capaz de resolver problemas de ordem elevada, não-lineares. Podem ser parcialmente conectadas, quando neurônios de uma camada se ligam apenas com alguns neurônios de outras camadas próximas, ou podem ser totalmente conectada e, nesse caso, todos os neurônios de uma camada se conectam com todos de suas camadas de ligação. A Figura 11 ilustra essa arquitetura.

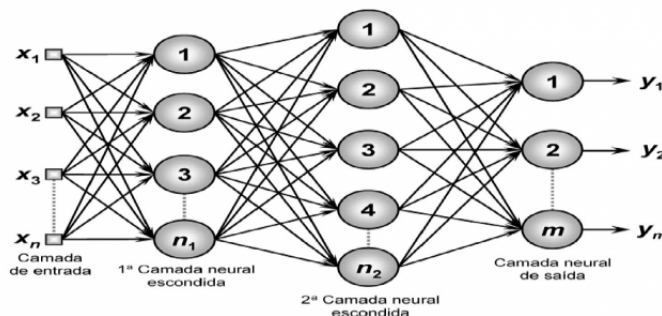


Figura 11 – Arquitetura multicamadas (<https://www.embarcados.com.br/redes-neurais-artificiais/>)

3. **Arquitetura recorrente**, onde cada neurônio é realimentado com a saída dos demais. Essa estrutura impacta profundamente no aprendizado e desempenho da rede, e é frequentemente utilizada em problemas que envolvem séries temporais e previsões.

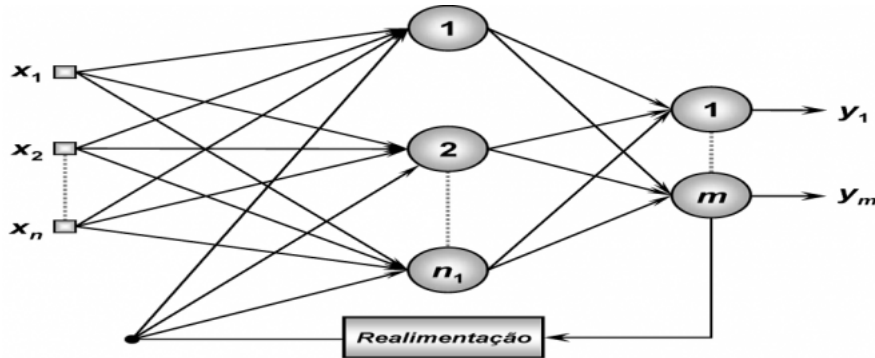


Figura 12 – Arquitetura Recorrente (<https://www.embarcados.com.br/redes-neurais-artificiais/>)

3.3.3 Aprendizagem

Para Haykin (2001), a aprendizagem é o processo pelo qual os parâmetros livres da rede se ajustam ao ambiente onde a mesma está inserida. O tipo de aprendizagem é determinado pela forma como ocorre esse ajuste de pesos, implicando em regras e paradigmas de aprendizado:

- Regras

1. Correção de erro: avalia-se a diferença entre a saída do neurônio e o valor-alvo, gerando um erro que deve ser minimizado pela atualização dos pesos sinápticos. Assim, de forma genérica, para um neurônio k com valor de entrada n tem-se,

$$e_k(n) = d_k(n) - y_k(n), \quad (3.3.6)$$

onde $e_k(n)$ é o erro encontrado, $d_k(n)$ é o valor-alvo e $y_k(n)$ é o valor de saída do neurônio. A cada iteração é feito o ajuste dos pesos sinápticos baseado no valor do erro anterior.

Esse ajuste é feito utilizando a regra delta (HAYKIN, 2001), também conhecida com LMS (least mean square), Gradiente Descendente ou Mínimos Quadrados, dada pela equação 3.3.7.

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n), \quad (3.3.7)$$

em que $\eta e_k(n) x_j(n)$ é a taxa de aprendizagem η aplicada ao erro $e_k(n)$ obtido na amostra $x_j(n)$ do treinamento. A taxa de aprendizagem η define a forma

(lenta ou rápida) com que a rede irá aprender, impactando diretamente no desempenho do treinamento e na precisão dos resultados. Assim, se a taxa for muito baixa a rede demorará a aprender (treinamento lento) e as suas respostas serão mais precisas, ao passo que uma taxa muito alta irá treinar a rede mais rapidamente, porém com resultados de precisão mais baixos. Visando manter o equilíbrio entre desempenho e precisão durante o treinamento, são usados algoritmos otimizadores (Adam, SGD, RMSprop, Adagrad, etc) que buscam atingir a melhor configuração de precisão e desempenho possíveis durante o ajuste do Gradiente.

Então, aplicando a regra delta aos pesos anteriores, encontramos os valores atualizados desses pesos sinápticos w_{kj} ;

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n) \quad (3.3.8)$$

2. Baseada em memória: as experiências passadas são armazenadas para servir de modelo na tomada de novas decisões, analisando-se a vizinhança e a proximidade com o padrão. Para tanto aplica-se o mínimo sobre o valor de todas as distâncias euclidianas dos vizinhos x (valor de teste) e X (valor padrão), sendo que o valor que mais se aproximar do padrão será tomado como vizinho mais próximo (y);

$$y = \min d(x, X) \quad (3.3.9)$$

3. Hebbiana: fortalece as conexões onde neurônios se ativam simultaneamente e frequentemente, e as enfraquece caso ocorra o contrário. Assim essa aprendizagem se baseia na correlação entre os neurônios;
4. Competitiva: os neurônios respondem de forma diferente ao mesmo conjunto de dados e por isso competem entre si para ficarem ativos, baseados na sua força de conexão;
5. Boltzman: baseada na mecânica estocástica, os neurônios assumem um estado binário (ligado ou desligado) ao longo da rede, de forma que o estado global desta seja dado pelo equilíbrio destes valores.

- Paradigmas

1. Supervisionado: existem exemplos rotulados para guiar as respostas da rede durante a aprendizagem. Compara-se a saída com o valor rotulado, e baseada na diferença entre eles, denominada de erro, se faz o ajuste dos pesos, até que essa diferença seja minimizada;
2. Não supervisionado: não há exemplos rotulados mas a rede irá formar os seus padrões para generalização, baseada numa medida de qualidade, independente

da tarefa que lhe foi atribuída, observando as condições do ambiente e ajustando seus parâmetros para otimizar a solução.

3. Por reforço: o mapeamento de entra-saída é feito através da interação com o ambiente, visando minimizar um índice escalar de desempenho ([HAYKIN, 2001](#)). Nesse tipo de aprendizado uma ação correta diminui o valor da função de custo e uma ação incorreta aumenta esse resultado. Logo a função de custo será dada pela soma dos reforços (positivos e negativos), o que implicará diretamente na sua precisão.

Assim, ao fazermos o treinamento de uma rede neural, estamos buscando sempre a minimização de sua função de custo, o que se traduz em maior precisão ao lidar com novos dados.

3.3.4 Keras API

Keras é uma API de alto nível para desenvolvimento de redes neurais ([Keras, 2019](#)), escrita em Python e capaz de rodar em cima do TensorFlow, CNTK ou Theano. Engloba muitos modelos de aprendizado, além de permitir a criação de novos ou fácil integração entre eles. Suas funcionalidades vão desde o pré-processamento dos dados, até a definição, treinamento e persistência dos modelos.

Podemos então elencar algumas vantagens para o uso do Keras:

1. Prototipagem rápida e acesso fácil a resultados, com o mínimo de esforço;
2. Utiliza CPU ou GPU, alternando automaticamente entre esses recursos, sem necessidade de programação adicional, aumentando seu desempenho;
3. Facilidade de aprendizado e utilização, aumentando a produtividade;
4. Fácil integração e transferência de conhecimento entre os modelos;
5. Ampla adoção no mercado de desenvolvimento e na comunidade de pesquisa, o que se traduz em muitas oportunidades, conforme mostra o gráfico da Figura 13;
6. Os modelos podem ser facilmente convertidos em produtos pois podem ser implementados em várias plataformas: IOS, Android, Web, Nuvem, Microcontroladores, etc;
7. O seu desenvolvimento é apoiado por empresas-chave do ecossistema de aprendizado profundo: Google, Microsoft, NVidia, Amazon, Uber e Apple.

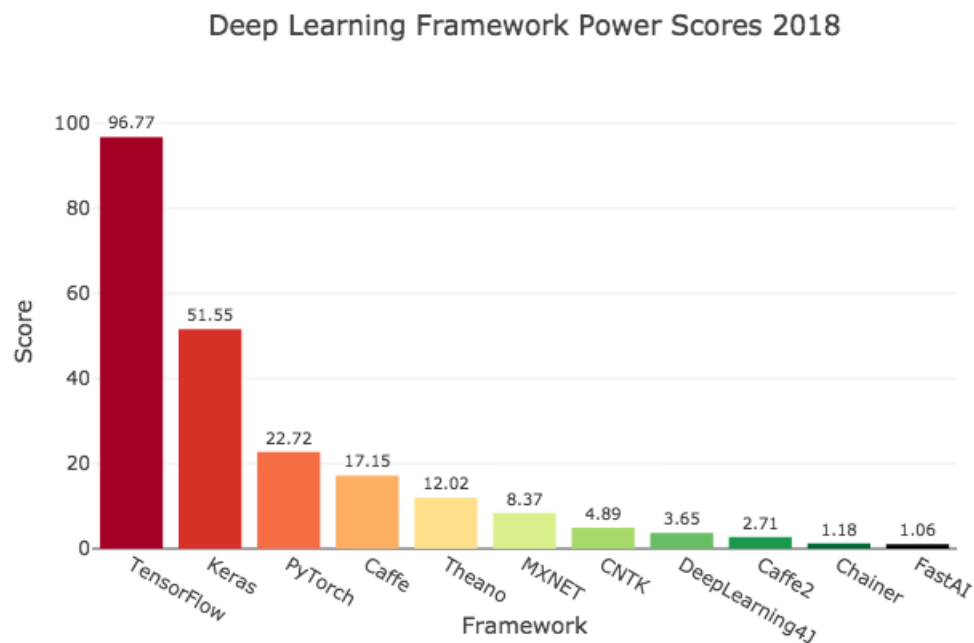


Figura 13 – Gráfico de utilização de frameworks de Deep Learning em 2018 ([Keras, 2019](#))

A utilização do Keras é aconselhada em situações onde se deseja prototipagem e resultados de forma rápida, testes de modelos, etc. Isso não significa que a ferramenta não possa ser usada em aplicações comerciais, e de fato ela é sim utilizada para este fim, o que faz com que o conhecimento dessa ferramenta possa ser considerado um diferencial no mercado.

4 MÉTODO

4.1 Definições

Para reconhecer indivíduos em imagens/vídeos propomos a utilização de partes do corpo, formadas a partir da ligação dos pontos obtidos pela biblioteca de OpenPose. Inicialmente buscávamos encontrar relações de natureza biométrica entre essas partes, para extrairmos parâmetros que mais tarde seriam usados para treinamento de uma rede neural. Assim, surgiu a ideia de utilizar os conceitos de Leonardo da Vinci, na sua obra do Homem Vitruviano ([Wikipedia.org](https://pt.wikipedia.org/wiki/Homem_Vitruviano),). Nesta obra ele determina algumas relações de proporção encaixando o corpo de um homem em um estrutura geométrica formada por um círculo e um quadrado, conforme mostra na Figura 14.

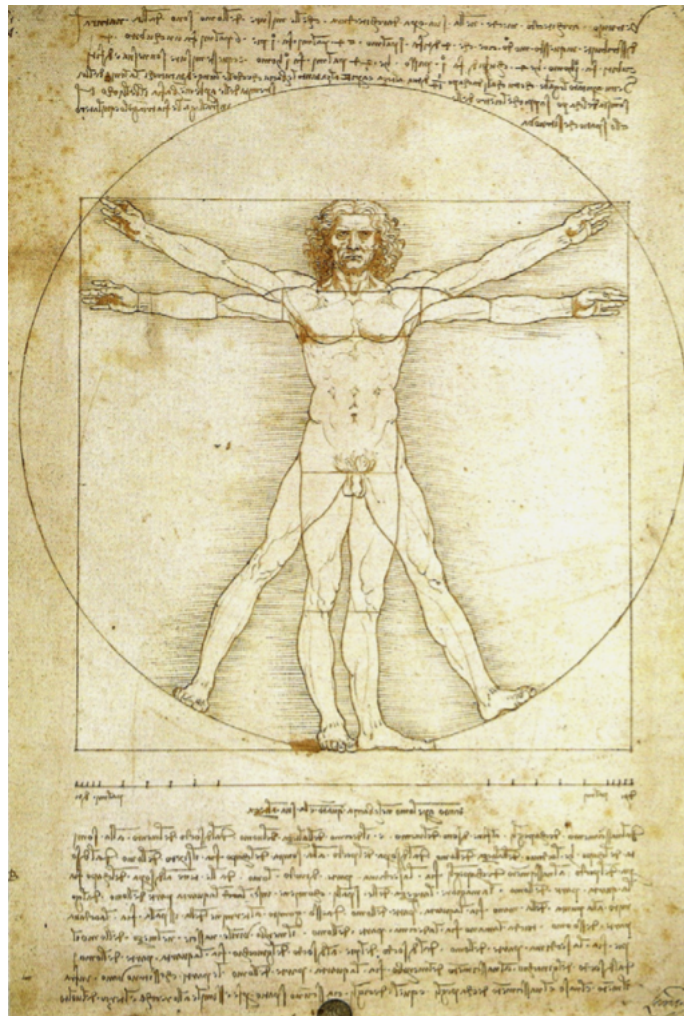


Figura 14 – Homem Vitruviano de Leonardo da Vinci, 1490 ([Wikipedia.org](https://pt.wikipedia.org/wiki/Homem_Vitruviano),)

No entanto, foi possível usar apenas a relação de proporção entre a envergadura

(braços abertos) e a altura, que segundo Da Vinci tem o mesmo comprimento. Devido à limitação do conjunto de pontos fornecidos pelo modelo de OpenPose utilizado no projeto, não pudemos explorar mais desse trabalho. Diante disso foi necessário estabelecer outras relações que possibilitassem a utilização do método proposto aqui, e para isso, dividimos o corpo formado pelos keypoints em partes (cabeça, tronco e membros), de acordo com a Figura 15 exibida abaixo:

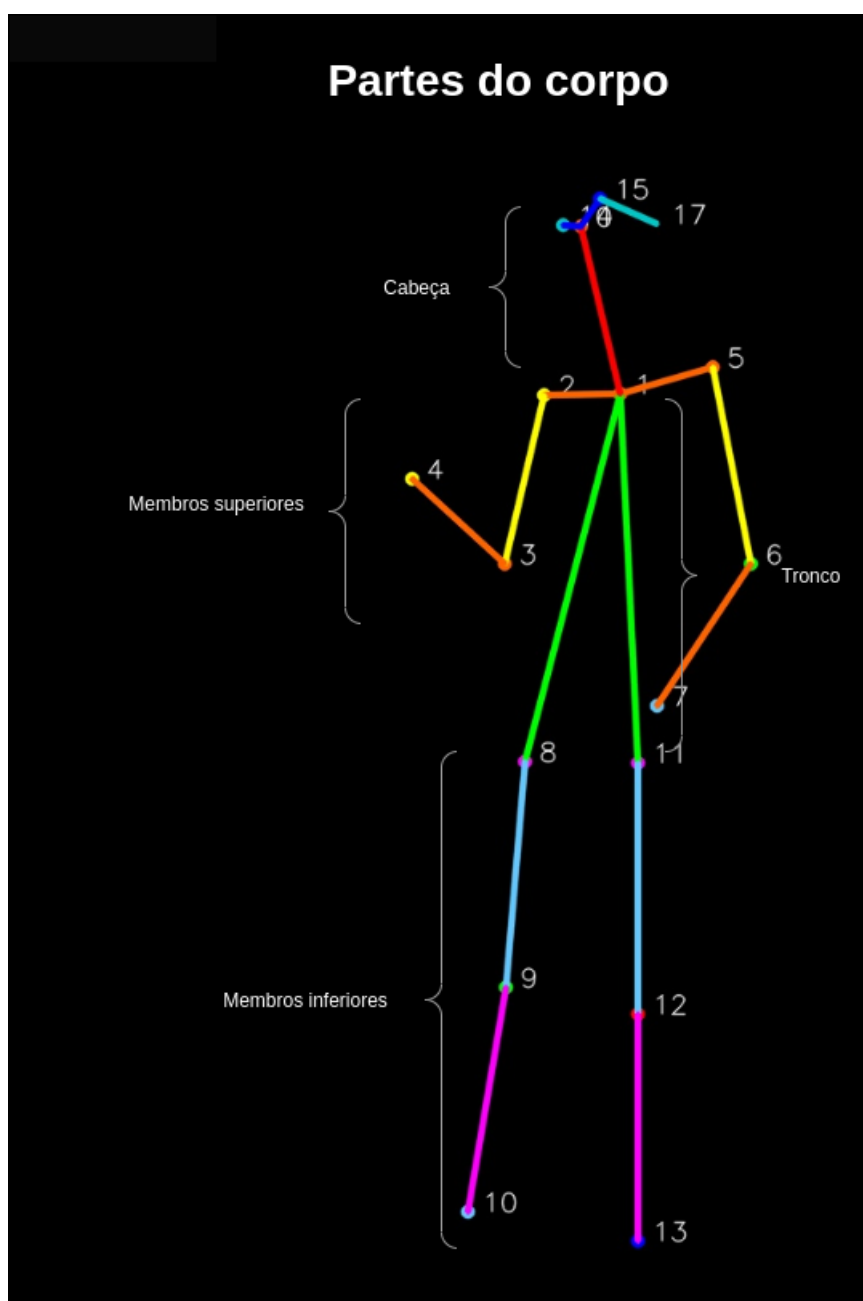


Figura 15 – Partes do corpo formadas pelos keypoints extraídos da biblioteca de Openpose (autoria própria)

Para chegarmos às relações definidas na Tabela 2, foi necessário calcular todas as distâncias entre os pontos que formam uma determinada parte do corpo, obtendo assim

o tamanho dessas partes. Essa medida foi calculada utilizando a fórmula da distância Euclidiana (d) entre os pontos A e B, como podemos ver na Figura 16 e na Equação 4.1.1 abaixo:

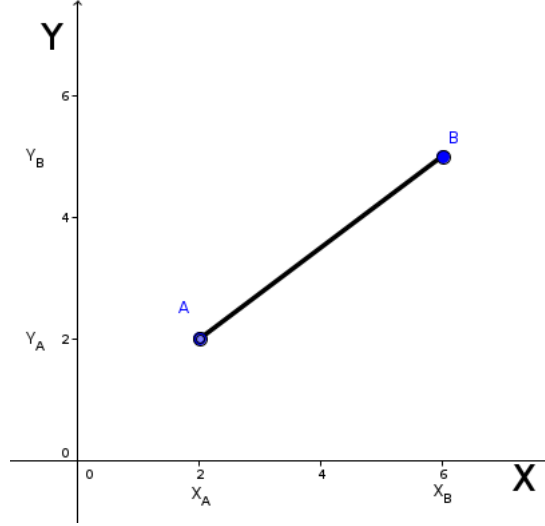


Figura 16 – Distância entre os pontos A e B (autoria própria)

$$d = \sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2} \quad (4.1.1)$$

Os tamanhos das partes do corpo foram definidas abaixo, de acordo com os modelos OpenPose (COCO e MPII) explicados na seção 3.1 do capítulo 3 :

$$(\text{COCO}), \quad \text{Cabeça} = \text{Pesçoço} + \text{Olhos} + \text{Nariz} \quad (4.1.2)$$

$$(\text{MPII}), \quad \text{Cabeça} = \text{Pesçoço} \quad (4.1.3)$$

$$\text{Ombros} = \text{Ombro}_{\text{Direito}} + \text{Ombro}_{\text{Esquerdo}} \quad (4.1.4)$$

$$\text{Braço} = \text{Braço} + \text{Antebraço} \quad (4.1.5)$$

$$(\text{COCO}), \quad \text{Tronco} = \text{Tronco}_{\text{Direito}} + \text{Tronco}_{\text{Esquerdo}} \quad (4.1.6)$$

$$(\text{MPII}), \quad \text{Tronco} = \text{Tronco} + \text{Quadril}_{\text{Direito}} + \text{Quadril}_{\text{Esquerdo}} \quad (4.1.7)$$

$$\text{Perna} = \text{Coxa}_{\text{Direita}} + \text{Coxa}_{\text{Esquerda}} \quad (4.1.8)$$

$$Largura = d(Quadri\!l_{Direito} + Quadri\!l_{Esquerdo}) \quad (4.1.9)$$

onde d é a distância euclidiana.

Como podemos observar na Figura 17, o tamanho de algumas partes não pode ser calculado diretamente. Se tomarmos como exemplo o tamanho do braço, veremos que essa medida sofre pequenas variações de um braço pra outro.

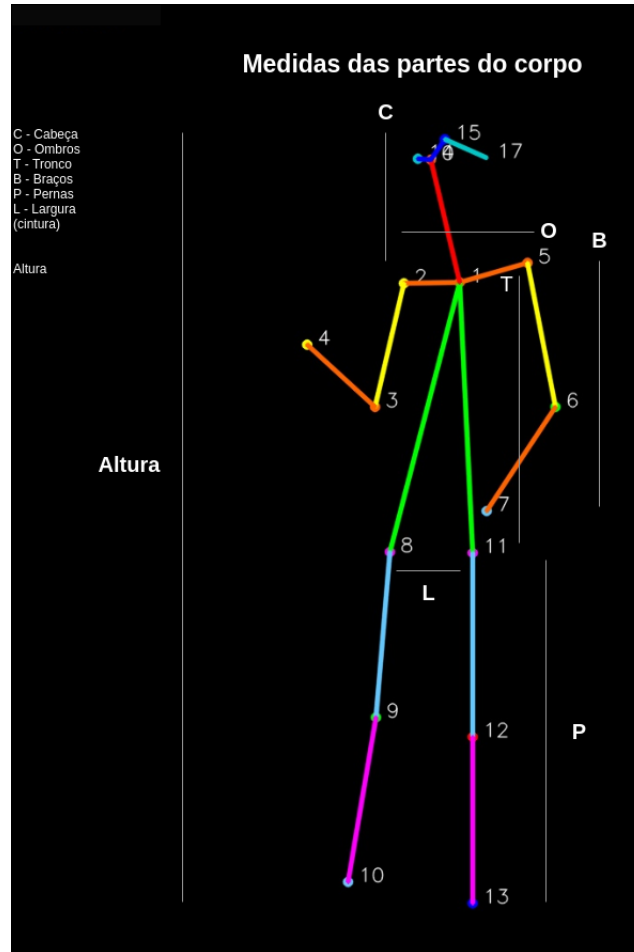


Figura 17 – Medidas do corpo (autoria própria)

A mesma coisa ocorre com a medida das pernas e do tronco devido à forma como o modelo foi idealizado e à variação da localização dos pontos na saída da rede OpenPose. Essa variação é de certa forma natural, já que se trata de uma estimativa desses pontos, dependendo assim da precisão do modelo e também da qualidade da entrada. Logo, a solução para o cálculo dessas partes foi usar a média dos valores obtidos em ambos, como mostra a Equação 4.1.10 abaixo:

$$Braço = \frac{Braço_{Direito} + Braço_{Esquerdo}}{2} \quad (4.1.10)$$

Com os tamanhos dos braços, tronco e pernas definidos foi possível calcular a altura somando os mesmos:

$$Altura = Cabeça + Tronco + Perna \quad (4.1.11)$$

Em seguida obtivemos o tamanho da envergadura:

$$Envergadura = braços + ombros \quad (4.1.12)$$

A partir da definição dessas partes criamos as relações de proporção dividindo cada uma delas pelas demais, totalizando 40 parâmetros, conforme temos na Figura 17 e na Tabela 2 abaixo:

Tabela 2 – Relações de proporção entre as partes do corpo (autoria própria)

Partes	Relações						soma
C - Cabeça	C_O	C_B	C_T	C_P	C_altura	C_L	S_C
O - Ombro		O_B	O_T	O_P	O_altura	O_L	S_O
B - Braço		B_O	B_T	B_P	B_altura	B_L	S_B
T - Tronco		T_O	T_B	T_P	T_altura	T_L	S_T
P - Pernas		P_O	P_T	P_B	P_altura	P_L	S_P
L - Largura		L_O	L_T	L_B	L_altura	L_P	S_L
Outras		E_A	P_M		EA_PM		

X_Y, corresponde à divisão X/Y.

E se refere à envergadura e A representa a altura.

(E_A) é a envergadura dividida pela altura.

A proporção média (P_M) corresponde à $(S_C + S_O + S_B + S_T + S_P + S_L)/6$, sendo S_X a soma das proporções da parte X do corpo.

Os parâmetros definidos nesta seção foram utilizados como entrada para o Classificador definido na seção 4.2.

4.2 Classificador

A rede neural do classificador foi implementada utilizando o framework Keras mencionado na seção 3.3.4 do capítulo 3. Trata-se de uma rede MLP - Multi Layer Perceptron totalmente conectada, tendo uma camada de entrada com seus parâmetros de entrada definidos na Tabela 2, duas camadas ocultas, com utilização do recurso de Dropout fornecido na implementação do modelo Sequential do Keras, e uma camada de saída.

O recurso Dropout é utilizado para zerar a saída de parte dos neurônios de uma camada, arbitrado aqui em 20%, evitando assim a ocorrência de overfitting¹.

Os parâmetros de configurações de inicialização², otimizador³ e ativação⁴ da rede estão descritos na Tabela 3 abaixo:

Tabela 3 – Parâmetros de configuração da rede neural (autoria própria)

Camadas	Nº. Neurônios	Inicialização	Otimizador	Ativação
Entrada	40	random_uniform	Adam	relu
Ocultas (2)	240	random_uniform	Adam	relu
Saída	1	random_uniform	Adam	sigmóide

Valores do otimizador Adam(lr⁵=0.001, decay⁶=0.0001, clipvalue⁷ = 0.5).

Utilizamos neste trabalho valores padrão do Keras, mas estas configurações podem ser alteradas para se ter uma descida de gradiente mais eficiente, prototipagem e testes. O número de neurônios nas camadas ocultas foi definido por meio de testes, nos quais a precisão aumentava em uma função da quantidade. Iniciamos com quantidade igual a metade da entrada e fomos aumentando gradualmente, até percebermos que após 240 a precisão não sofria variação considerável, e portanto definimos essa quantidade como padrão da rede.

A arquitetura da rede do Classificador está representada na Figura 18 a seguir:

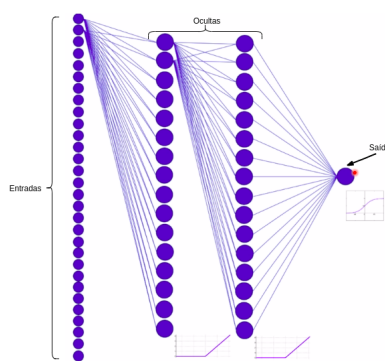


Figura 18 – Distribuição das camadas do Classificador, adaptada de (deep-learning-com-python-az-curso-completo)

¹ Overfitting - Treinamento excessivo provocando vício da rede nos dados de treinamento, o que leva a erros em novos dados

² random_uniform - especifica o tipo de inicialização em distribuição uniforme, dentro do intervalo

³ Adam - define o comportamento da descida do gradiente, visto em 3.3.3

⁴ relu - função que retorna 0 ou a entrada e Sigmóide - função de ativação que retorna probabilidade, conforme 3.3.1

⁵ taxa de aprendizagem

⁶ redução gradual da taxa de aprendizagem

⁷ trava o valor dos pesos para evitar oscilação indevida no Gradiente

4.2.1 Aplicação

A aplicação gerada neste trabalho tem basicamente quatro funções principais:

1. Capturar as imagens (foto ou vídeo) da webcam ou do disco rígido;
2. Gerar os dados para o treinamento e salvá-los em disco;
3. Treinar o classificador com os dados gerados persistindo o mesmo; e
4. Reconhecer novos dados através do classificador treinado, além de exibir a saída do processamento e salvá-la em disco, conforme configurado pelo usuário.

A Figura 19 abaixo, ilustra essas funções.



Figura 19 – Etapas da aplicação (autoria própria)

A interface gráfica é composta de uma tela principal onde o usuário pode escolher algumas configurações de entrada e saída de dados, e o tipo de processamento desejado.

Conforme o tipo de processamento escolhido serão exibidas na tela algumas informações do classificador e durante o processo será exibido um ou mais players com a saída escolhida. Assim, na tela principal temos:

- Fonte - determina o local de onde os vídeos serão lidos;
- Tipo de rede - se refere aos modelos de Openpose explicados na seção 3.1 do capítulo 3;
- Salvar saída - define a forma como os vídeos serão salvos com os pontos sobrepostos;
- Fator de redução - permitiria reduzir a dimensionalidade da entrada mantendo as suas características e melhorando o desempenho, mas não obtivemos melhora no desempenho que garantisse a precisão da saída dos keypoints e portanto não aconselhamos a utilização deste recurso;
- Tipo de processamento - permite escolher que função da aplicação iremos utilizar, bem como a escolha dos modelos treinados e se estes serão usados em conjunto ou sozinhos.

Abaixo vemos ilustrações de sua interface gráfica, nas Figuras 20, 21, 22 e 23, respectivamente:

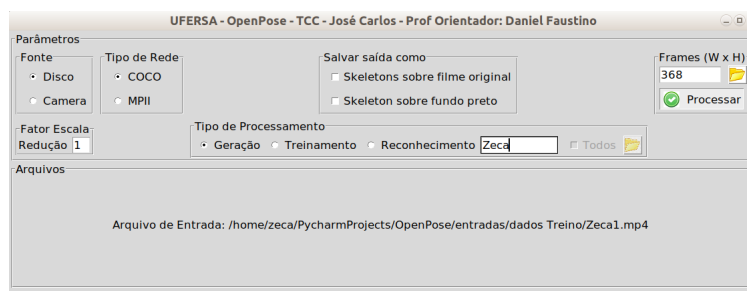


Figura 20 – Interface da aplicação (autoria própria)

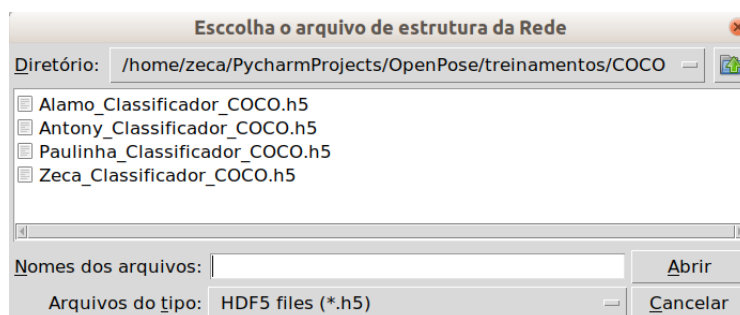


Figura 21 – Seleção do Classificador treinado (autoria própria)

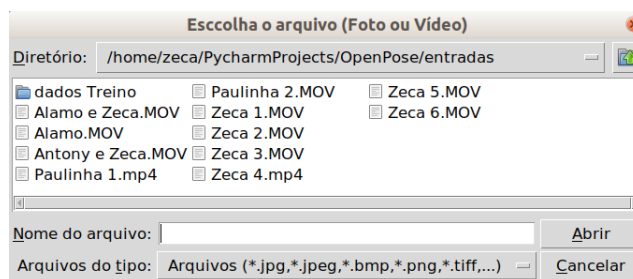


Figura 22 – Seleção da imagem de entrada (autoria própria)



Figura 23 – Exibição de vídeo de reconhecimento, (autoria própria)

Na próxima seção discorreremos acerca do processo de geração dos dados de treinamento.

4.2.2 Geração de dados

Os dados utilizados para o treinamento da rede neural foram coletados em vídeos capturados a partir de câmeras de segurança residencial, modelo Intelbras IC4 e foram processados utilizando um notebook Lenovo Ideapad 330, equipado com processador Intel® Core™ i7-8550U CPU @ 1.80GHz 8, memória de 8GB e placa de vídeo NVidia GeForce MX150/PCIe/SSE2 2GB, utilizando sistema operacional Linux Ubuntu 18.04.03 LTS 64 bits.

Para geração dos dados de treinamento foram capturados vários vídeos de pessoas e transferidos para um computador por meio de um dispositivo mobile. Com esses vídeos salvos em disco, é feita a leitura de cada frame e processamos na aplicação, calculando as medidas e proporções e salvando tudo em arquivos CSV. Cada arquivo corresponde a um vídeo de uma pessoa, sendo que cada linha do arquivo contém as informações das proporções e o rótulo para treinamento, ou seja, uma linha corresponde exatamente a um frame do vídeo. A Figura 24 mostra o processo de geração dos dados e a Figura 25 exibe os dados um arquivo CSV.

A Figura 26 mostra informações sobre a distribuição estatística dos dados, para

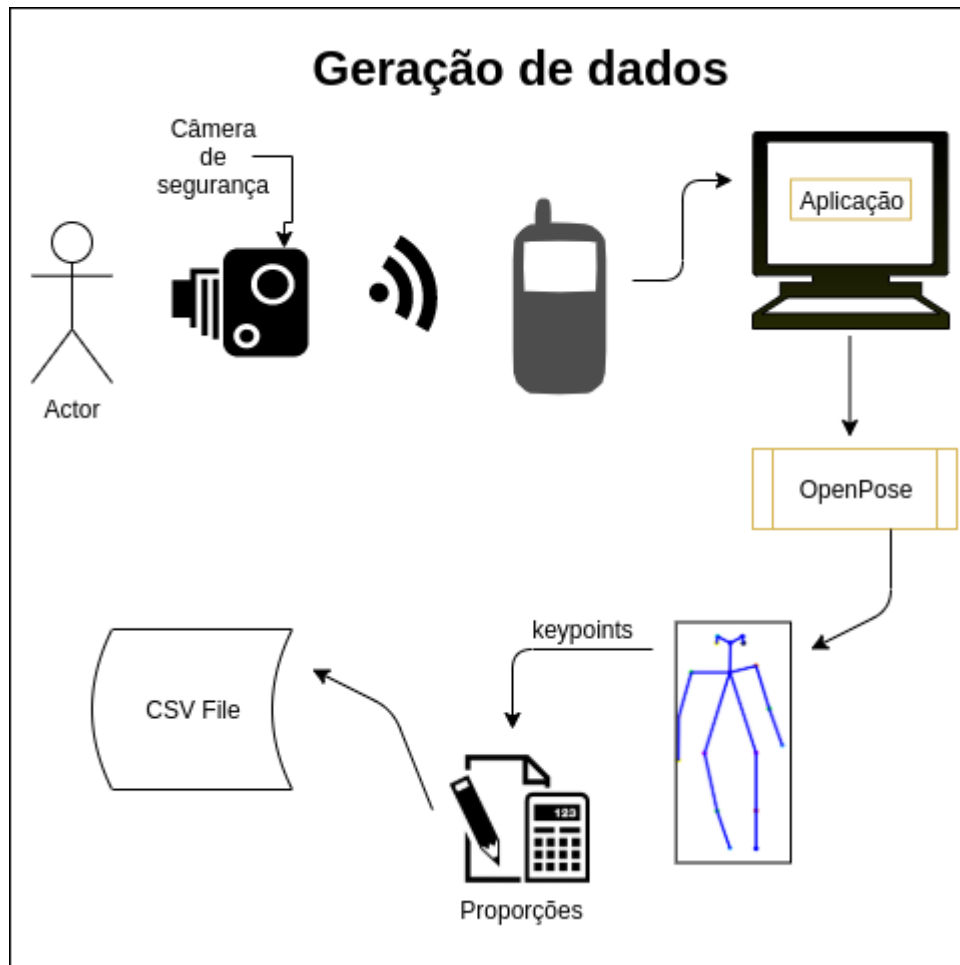


Figura 24 – Processo de geração de dados para treinamento (autoria própria)

	enverg_altura	C_O	C_B	C_T	C_P	C_L	C_Altura	O_B	O_T	O_P	...	L_Altura	SC	SO	SB	ST	SP	SL	Proporcao_media	EA_PM	nome
0	1.09	0.55	0.28	0.31	0.26	0.55	0.12	0.52	0.57	0.47	...	0.22	2.07	2.80	6.31	5.68	7.08	2.74	4.45	0.244944	1
1	1.09	0.55	0.28	0.31	0.26	0.55	0.12	0.52	0.57	0.47	...	0.22	2.07	2.80	6.31	5.68	7.08	2.74	4.45	0.244944	1
2	1.07	0.63	0.33	0.36	0.30	0.64	0.14	0.52	0.57	0.47	...	0.22	2.40	2.79	6.31	5.67	7.07	2.74	4.50	0.237778	1
3	1.09	0.63	0.33	0.36	0.30	0.65	0.14	0.52	0.57	0.48	...	0.22	2.41	2.81	6.34	5.71	6.91	2.74	4.49	0.242762	1
4	1.09	0.63	0.33	0.36	0.30	0.65	0.14	0.52	0.57	0.48	...	0.22	2.41	2.81	6.37	5.71	6.90	2.73	4.49	0.242762	1

Figura 25 – Dados de um arquivo CVS com as proporções e rótulos para treinamento (autoria própria)

todos os parâmetros das proporções.

Devido à demora no processo de extração das informações dos vídeos, levando cerca de 3 segundos, em média, para passar pela rede OpenPose, fizemos a coleta de dados apenas para o modelo COCO para usarmos nos treinamentos.

4.2.3 Treinamento

Para o treinamento do Classificador foram utilizados os dados de vários arquivos. Esses arquivos foram lidos todos de uma vez e as suas informações guardadas em uma

----- Média -----		----- Desvio -----		----- Máximos -----		----- Mínimos -----	
enverg_altura	1.078534	enverg_altura	0.114556	enverg_altura	1.930000	enverg_altura	0.550000
C_O	0.417974	C_O	0.129468	C_O	0.960000	C_O	0.010000
C_B	0.287283	C_B	0.081089	C_B	0.790000	C_B	0.010000
C_T	0.290330	C_T	0.082173	C_T	0.540000	C_T	0.010000
C_P	0.231023	C_P	0.062560	C_P	0.440000	C_P	0.000000
C_L	0.631698	C_L	0.217300	C_L	1.770000	C_L	0.010000
C_Altura	0.112954	C_Altura	0.028119	C_Altura	0.180000	C_Altura	0.000000
O_B	0.711572	O_B	0.156841	O_B	2.540000	O_B	0.340000
O_T	0.711439	O_T	0.119426	O_T	2.090000	O_T	0.410000
O_P	0.567710	O_P	0.098261	O_P	0.930000	O_P	0.300000
O_L	1.533851	O_L	0.341329	O_L	3.860000	O_L	0.730000
O_Altura	0.279276	O_Altura	0.045630	O_Altura	0.510000	O_Altura	0.160000
B_O	1.466852	B_O	0.292792	B_O	2.900000	B_O	0.390000
B_T	1.020447	B_T	0.150977	B_T	2.950000	B_T	0.320000
B_P	0.813232	B_P	0.118334	B_P	1.160000	B_P	0.220000
B_L	2.213924	B_L	0.536108	B_L	5.110000	B_L	0.750000
B_Altura	0.399670	B_Altura	0.051232	B_Altura	0.710000	B_Altura	0.120000
T_O	1.441856	T_O	0.225844	T_O	2.420000	T_O	0.480000
T_B	1.003169	T_B	0.167447	T_B	3.120000	T_B	0.340000
T_P	0.801911	T_P	0.090797	T_P	1.170000	T_P	0.320000
T_L	2.178943	T_L	0.455959	T_L	4.530000	T_L	0.650000
T_Altura	0.393561	T_Altura	0.028455	T_Altura	0.500000	T_Altura	0.230000
P_O	1.814454	P_O	0.315686	P_O	3.350000	P_O	1.070000
P_B	1.260834	P_B	0.223850	P_B	4.450000	P_B	0.870000
P_T	1.264403	P_T	0.158794	P_T	3.130000	P_T	0.850000
P_L	2.752713	P_L	0.670718	P_L	6.210000	P_L	1.550000
P_Altura	0.493915	P_Altura	0.030643	P_Altura	0.750000	P_Altura	0.420000
L_O	0.681842	L_O	0.143318	L_O	1.370000	L_O	0.260000
L_B	0.477224	L_B	0.114496	L_B	1.330000	L_B	0.200000
L_T	0.478326	L_T	0.098963	L_T	1.530000	L_T	0.220000
L_P	0.382955	L_P	0.086951	L_P	0.650000	L_P	0.160000
L_Altura	0.188155	L_Altura	0.039634	L_Altura	0.370000	L_Altura	0.080000
SC	1.971263	SC	0.548444	SC	3.880000	SC	0.040000
SO	3.803847	SO	0.594913	SO	7.100000	SO	2.050000
SB	5.914125	SB	0.922197	SB	9.700000	SB	1.800000
ST	5.819439	ST	0.635828	ST	8.820000	ST	2.030000
SP	7.586319	SP	1.064512	SP	13.540000	SP	5.170000
SL	2.208502	SL	0.422275	SL	3.680000	SL	1.000000
Proporcao_media	4.550644	Proporcao_media	0.308785	Proporcao_media	6.210000	Proporcao_media	3.940000
EA_PM	0.238408	EA_PM	0.032669	EA_PM	0.421397	EA_PM	0.103383

Figura 26 – Informações estatísticas dos dados de um arquivo CVS (autoria própria)

estrutura de dados em memória, como se fossem um único arquivo CSV com todas as proporções e rótulos de várias pessoas. O modelo implementado suporta o treinamento apenas uma pessoa por vez, gerando o classificador desta. Assim foi necessário para cada pessoa informar qual o valor do rótulo que desejávamos treinar, de forma manual ou automaticamente. Ao informarmos um rótulo a aplicação atribui para este um valor verdadeiro e para os demais um valor falso, tornando possível o treinamento do modelo.

Para efetivar o treinamento essa estrutura em memória, a base de dados, foi dividida automaticamente em duas outras:

1. Base de treinamento - corresponde a 75% do total de dados e foi usada para treinar a rede com as informações dos rótulos; e
2. Base de testes - corresponde a 25% do total de dados e foi usada para testar o modelo, atribuindo a este o valor da acurácia e do erro.

O treinamento foi feito para quatro classificadores de pessoas diferentes, usando vários vídeos e com várias posições entre os frames, como forma de extrair o máximo de informações possíveis para a entrada da rede. Abaixo, a Figura 27 mostra algumas poses usadas para o treinamento. Cada modelo foi treinado por 10.000 (dez mil) épocas, sendo que a cada época foi salvo o modelo que tivesse atingido a maior acurácia, de modo que fosse mantido apenas o modelo mais preciso de todo o treinamento para cada pessoa. O



Figura 27 – Imagens com poses de treinamento (autoria própria)

processo levou cerca de 3 dias para completar o treinamento dos 4 modelos, obtendo as seguintes métricas na base de testes:

- Alamo - erro = 0.18413213383436852 e precisão = 0.9362397789955139;
- Antony - erro = 0.16938286273453476 e precisão = 0.9411444067955017;
- Paulinha - erro = 0.3510515952909221 e precisão = 0.8362389206886292; e
- Zeca - erro = 0.23091627178630103 e precisão = 0.894541323184967.

4.2.4 Reconhecimento

O reconhecimento de indivíduos é feito nas seguintes etapas: leitura dos frames, cálculo das medidas e proporções, processamento no classificador, geração e exibição dos resultados. Primeiro é feita a leitura dos frames da webcam ou do disco e passamos pela rede OpenPose, recebendo como saída os keypoints das partes do corpo. Com esses dados calculamos os tamanhos das partes do corpo conforme vimos na Equação de distância euclidiana 4.1.1 e em seguida geramos as proporções das mesmas. Depois esses esses parâmetros foram enviados para o Classificador que vai retornar a probabilidade de determinado indivíduo, quando será então exibido o nome e o percentual se essa probabilidade atingir 90% para a resposta. Caso contrário será exibido apenas o percentual em questão. O resultado será então uma lista contendo o nome da pessoa classificada, a probabilidade e a posição no frame.

É possível utilizar os classificadores em conjunto, como se fossem um único classificador multiclases atuando no processo de reconhecimento. Isso permite classificar mais de uma pessoa no mesmo vídeo. Quando se tem mais de uma pessoa no vídeo o classificador irá gerar um resultado para cada indivíduo e caso haja mais de um classificador atuando sobre

a mesma pessoa, vencerá aquele que atingir a maior precisão para a pessoa classificada, mesmo que esta não seja a classificação correta, ou seja, a prioridade é a precisão mais alta. Essa técnica permite que computar os erros e acertos para vídeos que tem somente uma pessoa. Esses dados são guardados em uma estrutura denominada Linha do tempo, utilizada mais tarde para exibição e verificação dos resultados. A Figura 28 fornece uma ideia geral do processo de reconhecimento.

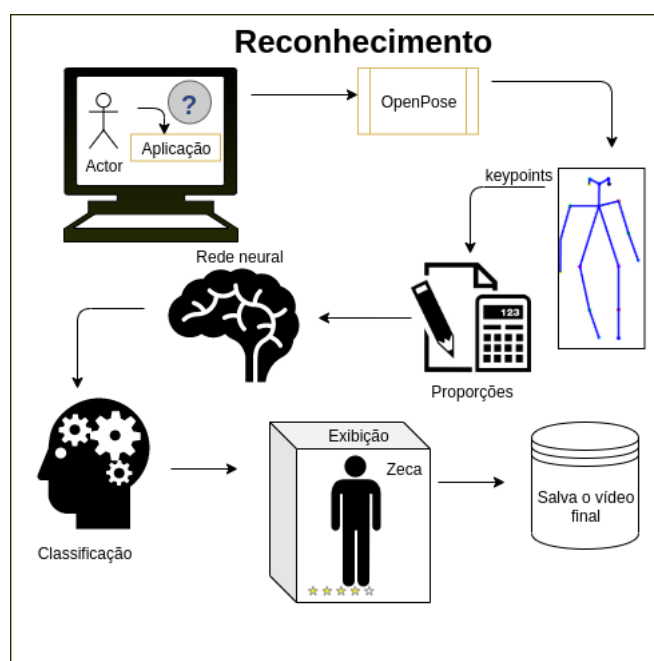


Figura 28 – Etapas do reconhecimento de indivíduos (autoria própria)

A exibição, assim como a captura dos frames, é lenta devido à demora na resposta da rede OpenPose, e por isso foi necessário salvar os resultados com uma taxa muito baixa de exibição, cerca de 1 a 3 frames por segundo, pois de outra forma os vídeos passariam muito rápido e não seria possível observar os resultados.

5 RESULTADOS

Para validação do classificador foram utilizados 5 vídeos, os quais não fizeram parte das bases de treinamento e testes, sendo três vídeos com somente uma pessoa e dois com duas pessoas em cena. Fizemos verificações com atuação de apenas um classificador por vez, depois com dois classificadores e por último com os quatro juntos. Essa técnica permitiu analisar o comportamento dos classificadores no reconhecimento da pessoa para o qual foram treinados e também para pessoas de outros classificadores, bem como observar o comportamento com vídeos de pessoas que não tiveram nenhum classificador treinado na rede, ou seja, pessoas totalmente desconhecidas pelo modelo.

Com o intuito de testar os classificadores como se estivessem em situações reais de reconhecimento, definimos que o valor limiar para classificação seria de pelo menos 90% de precisão. Ao atingir essa taxa, a aplicação exibiria o resultado e o nome da pessoa classificada, e abaixo disso somente mostraria os percentuais. Assim foi possível verificar em um determinado vídeo, quantas vezes a rede classificou os frames e destas vezes, quantas foram corretas, baseado na linha do tempo do vídeo e no nome do mesmo. Ex. Se estivermos classificando um vídeo com o nome Zeca.mp4 e neste vídeo aparece apenas o indivíduo Zeca, utilizando os quatro classificadores em conjunto, poderemos computar os acertos, quando o classificador_Zeca atingir 90%, ou computar erro quando um dos demais classificadores atingirem esse percentual. Esses dados são guardados na estrutura de linha do tempo do vídeo e serializados para uso posterior.

O comportamento da rede variou conforme descrito abaixo:

1. Indivíduos com modelo treinado:

- a) um classificador ativo: precisão acima de 90%; e
- b) todos os classificadores ativos: precisão acima de 90%, com alguns erros de falso positivo.

2. Indivíduos sem modelo treinado:

- a) um classificador ativo: poucas classificações, ou seja, poucos erros; e
- b) todos os classificadores ativos: muitas classificações, o que representa muitos erros.

A seguir, nas Figuras 29, 30, 31, 32 e 33 , apresentamos algumas classificações utilizando um ou mais classificadores:



Figura 29 – Reconhecimento Zeca (autoria própria)

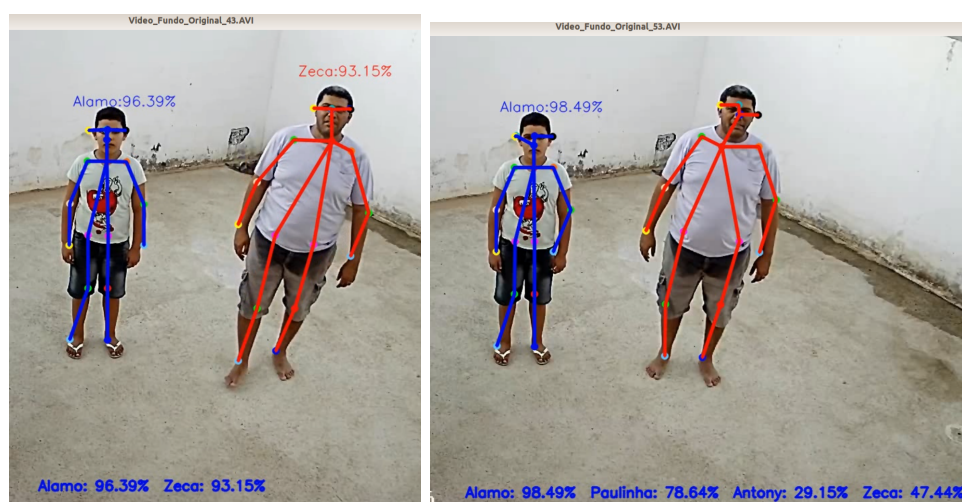


Figura 30 – Reconhecimento de Alamo e Zeca (autoria própria)

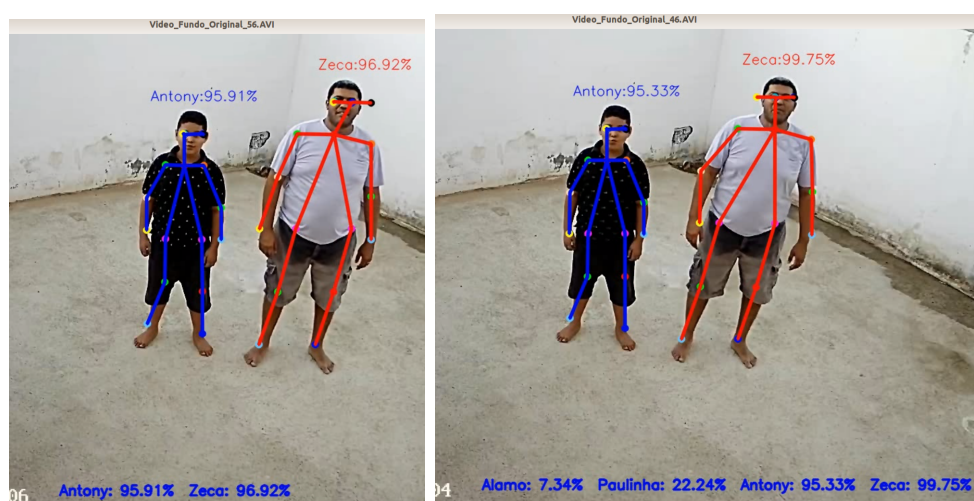


Figura 31 – Reconhecimento de Antony e Zeca (autoria própria)

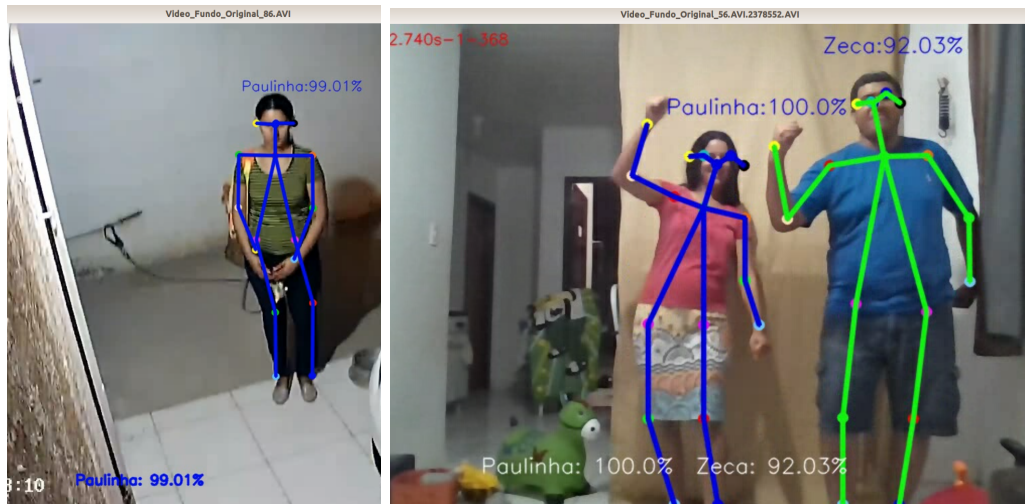


Figura 32 – Reconhecimento de Paulinha e Zeca (autoria própria)

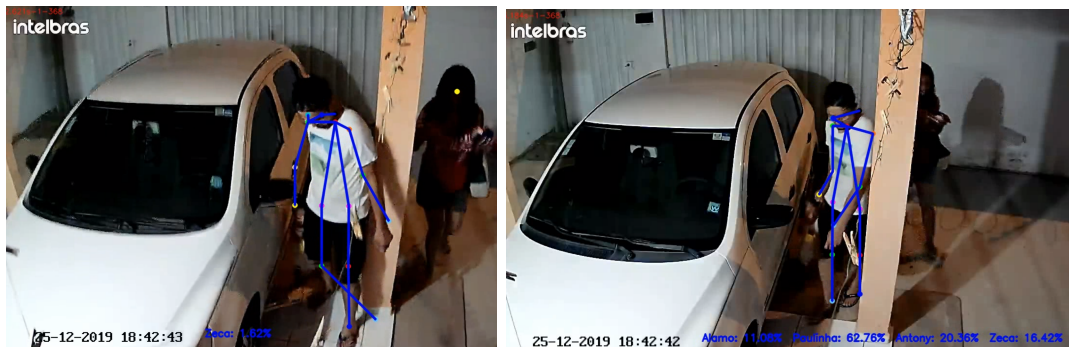


Figura 33 – Reconhecimento de Pablo, sem modelo treinado (autoria própria)

A análise das linhas do tempo dos vídeos que passaram pelo classificador está disposta na Tabela 4 abaixo:

Tabela 4 – Análise das linhas do tempo dos vídeos processados. (autoria própria)

Indivíduos	Modelos treinados	Frames	Classificações	Acertos	Erros
Alamo e Zeca	Alamo.H5 e Zeca.H5	30	18 (60%)	18 (100%)	0 (0%)
Alamo e Zeca	Todos	16	8 (50%)	8 (100%)	0 (0%)
Antony e Zeca	Antony.H5 e Zeca.H5	14	13 (93%)	13 (100%)	0 (0%)
Antony e Zeca	Todos	23	10 (43%)	10 (100%)	0 (0%)
Pablo	Zeca.H5	13	3 (23%)	10 (77%)	3 (23%)
Pablo	Todos	13	12 (92%)	1 (8%)	12 (92%)
Paulinha	Paulinha.H5	18	7 (39%)	7 (100%)	0 (0%)
Paulinha	Todos	18	17 (94%)	7 (41%)	10 (59%)
Zeca	Zeca.H5	10	7 (70%)	7 (100%)	0 (0%)
Zeca	Todos	10	7 (70%)	7 (100%)	0 (0%)

Analizando os dados acima, na Tabela 4, percebemos que os classificadores se comportam bem quando estão classificando os indivíduos para os quais foram treinados, o que é esperado, porém ao adicionarmos classificadores de outros indivíduos em vídeos de

pessoas não correspondentes a estes, temos aumento na taxa de erro, pois os classificadores das pessoas não presentes no vídeo ficam "competindo" com os classificadores dos indivíduos que pretendemos reconhecer. Quando o percentual desses classificadores é maior que o dos indivíduos no vídeo, eles acabam classificando errado em alguns casos, como é o exemplo do classificador de Paulinha que acaba perdendo muito quando atua em conjunto, gerando uma taxa de erros de 59% em seu próprio vídeo.

Cada classificador foi utilizado sozinho, reconhecendo o indivíduo para o qual foi treinado, e também em conjunto com outros classificadores. O número de frames classificados variou entre 23% e 94%, dependendo do classificador e dos indivíduos presentes no vídeo. Já o percentual de acerto foi calculado somente para os frames classificados, para os quais definimos que o mínimo de precisão seria de 90% para a classificação. Assim, para estes, o percentual ficou entre 77% e 100%, sendo os melhores resultados para classificadores que atuaram sozinhos em vídeos de indivíduos conhecidos, e os piores resultados para atuações de classificadores em conjunto. Os gráficos dessas linhas do tempo também mostram esse comportamento e, embora o número de frames classificados ainda seja baixo em relação ao total de frames, as classificações corretas nos mostram que os classificadores treinados se sobressaem. Se observarmos nesses gráficos os valores acima de 90%, veremos que em sua maioria tivemos as classificações corretas. É interessante observar que durante a passagem dos frames, quando temos uma mudança brusca de pose, entre uma pose estável e outra os classificadores tem uma queda de precisão devido a localização incorreta de alguns keypoints, e por isso temos certas oscilações inesperadas nos gráficos.

Os gráficos desta análise, na mesma ordem da Tabela 4, são exibidos abaixo em dois gráficos, onde à esquerda temos os classificadores das pessoas presentes no vídeo e à direita o gráfico de todos os classificadores treinados atuando juntos. No eixo X temos a quantidade de frames analisados e no eixo Y vemos os valores das probabilidades para cada classificador. Assim, na Figura 34 é mostrada a linha do tempo para os indivíduos Álamo e Zeca, na qual observamos que o classificador de Álamo foi ativado mais vezes no início e menos vezes no final, ao contrário do classificador de Zeca, que no início classificou pouco mas no final se manteve sempre acima do limiar definido. Já para a atuação em conjunto, apenas Álamo classificou. Os demais ficaram abaixo de 90%, sendo que o classificador de Paulinha foi o que mais se aproximou desse valor.

Na figura 35 observamos à esquerda os classificadores de Zeca e Antony. Nesse gráfico o número de ativações foi bem satisfatório, estando na maior parte do tempo acima do limiar de 90% para os dois classificadores. A atuação em conjunto é mostrada à direita, onde vemos que Antony continuou ativando muito, ao passo que Zeca classificou apenas três vezes. Os demais se mantiveram abaixo do limiar, e o classificador de Paulinha mais uma vez esteve próximo de ser ativado.

Podemos observar na Figura 36 o comportamento dos classificadores ao processarem

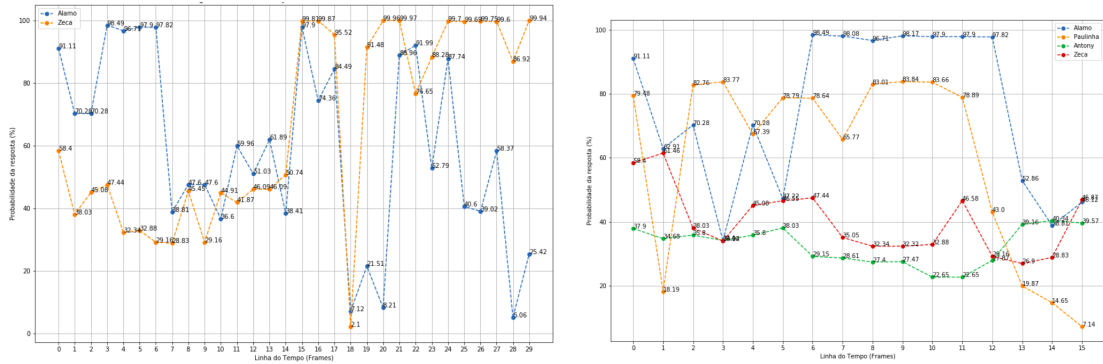


Figura 34 – Gráfico da linha do tempo de um vídeo com os indivíduos Alamo e Zeca (autoria própria)

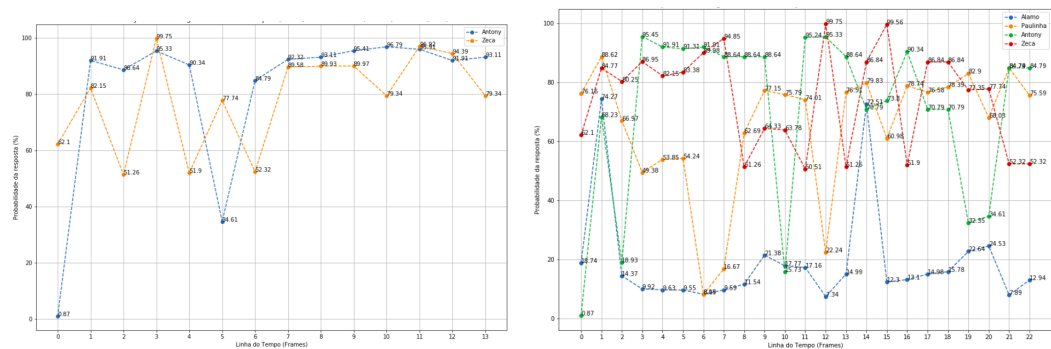


Figura 35 – Gráfico da linha do tempo de um vídeo com os indivíduos Antony e Zeca (autoria própria)

um vídeo de um indivíduo para o qual não houve nenhum treinamento. Vemos à direita o classificador Zeca atuando no vídeo do indivíduo Pablo e sendo ativado apenas três vezes. Ou seja, tivemos apenas três erros, o que significa 77% de acerto, com percentuais abaixo de 20% para estas probabilidades, o que nos remete certo grau de eficácia para este classificador. Na atuação de todos os classificadores Zeca se comportou da mesma forma. Os demais classificadores, exceto o de Paulinha, também se comportaram bem.

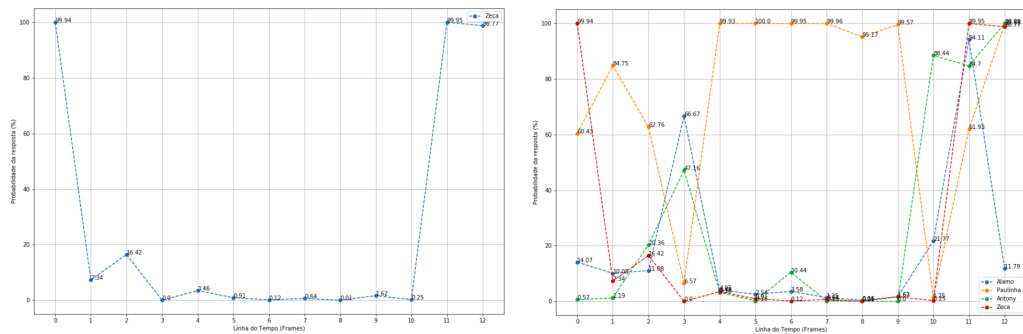


Figura 36 – Gráfico da linha do tempo de um vídeo com o indivíduo Pablo (autoria própria)

Na Figura 37, temos à esquerda o classificador de Paulinha atuando sozinho.

Mesmo classificando poucos frames a precisão foi muito alta, atingindo algumas vezes 100%. Usando todos os classificadores, no gráfico da direita, vemos que Paulinha, Álamo e Antony tiveram bons resultados, porém Zeca errou bastante, o que não era esperado para este classificador já que ele tem se comportado bem em diversos outros vídeos.

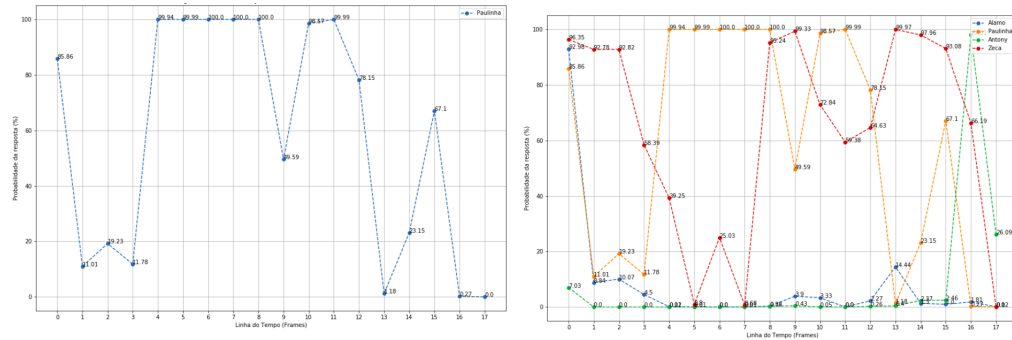


Figura 37 – Gráfico da linha do tempo de um vídeo com Paulinha (autoria própria)

Por fim, na Figura 38, temos um vídeo do indivíduo Zeca sendo processado, à esquerda, pelo classificador Zeca, onde obtivemos resultados de ativação bem próximos de 100%, e à direita temos a atuação conjunta dos classificadores, na qual tivemos sucesso em todos eles, sendo que Zeca se comportou da mesma maneira e os demais estiveram abaixo do limiar, na maior parte do tempo com valores abaixo de 50% de precisão.

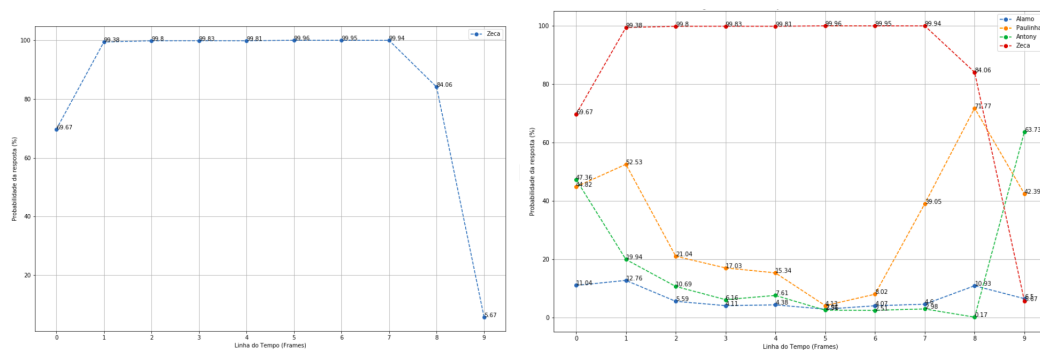


Figura 38 – Gráfico da linha do tempo de um vídeo com o indivíduo Zeca (autoria própria)

6 CONCLUSÃO

É importante fazer algumas considerações importantes acerca do método utilizado para reconhecer indivíduos utilizando estimativa de pose humana. Ao se analisar as linhas do tempo verificou-se que é possível reconhecer pessoas utilizando as medidas do seu corpo, uma vez que chegamos, em alguns casos, a precisão de mais de 90% ao classificar os frames, com taxas de acerto que variaram de 77% a 100%. A taxa de classificação ainda é baixa, mas isso, do ponto de vista prático, não representa problemas, pois se em um vídeo de 30 frames, por exemplo, conseguirmos reconhecer um indivíduo com acerto em pelo menos 25% destes, teremos uma classificação positiva. Para casos de vídeos de pessoas que não tiveram um modelo de classificador treinado, principalmente ao utilizar os modelos treinados de outras pessoas em conjunto, observou-se muitas classificações, o que representa erros já que a rede não deveria reconhecer um indivíduo desconhecido. Esse aumento de erros também pode significar que a escolha de usar proporções pode não ter sido a melhor, mas isso só pode ser respondido de fato, ao se testar outras hipóteses, em trabalhos futuros.

Deve-se destacar também que pessoas com estaturas muito parecidas podem ser classificadas de forma errada devido às similaridades das medidas. Também não há como fazer distinção entre pessoas adultas e crianças já que do ponto de vista do processamento digital de imagens essa relação de tamanho e proporção pode ser vista apenas como a distância de observação das pessoas na imagem, se estão ao fundo ou mais próximas.

Com relação à aplicação, tivemos problemas com o desempenho que ficou prejudicado pela demora em obter os keypoints da rede OpenPose, algo em torno de 2s a 3s para entregar os pontos. Por outro lado, o classificador implementado leva apenas alguns milésimos de segundo para reconhecer uma pessoa e completar a tarefa de classificação. Também perdemos em precisão, pois os keypoints variaram muito de posição no plano para uma mesma parte do corpo e isso prejudicou muito a extração das medidas e proporções, tornando-as muito próximas e deixando o conjunto de dados de uma pessoa muito parecido com o de outra, refletindo diretamente no processo treinamento da rede e consequentemente na sua precisão.

Devido à demora em alguns processos não foi possível explorar mais parâmetros e configurações na implementação da rede neural, uma vez que se perdia muito tempo para extrair os dados e posteriormente para treinar o modelo com esses dados. Por fim, mesmo com esses problemas, o método e os parâmetros utilizados, são de modo geral, uma boa escolha.

Para projetos futuros pode-se rever os parâmetros de configuração da implementação

da rede neural e treinar novamente o classificador. Além disso é apropriado revisar os dados de entrada da rede, e criar novos parâmetros que possibilitem uma extração de características mais concisas.

Referências

SOUZA, G. B.; MARANA, A. N. Reconhecimento de pessoas em imagens de vídeo utilizando características biométricas leves. *FCT/UNESP - P. Prudente VI Workshop de Visão Computacional*, 2004. Citado na página 17.

VIEIRA, R. M. T. Identificação de indivíduos pela dinâmica do caminhar: Dissertação (mestrado). *Engenharia Elétrica, Centro Tecnológico da Universidade Federal do Espírito Santo*, 2009. Citado na página 17.

NASCIMENTO, M. V. P. do. Um método de reconhecimento de indivíduos por geometria da mão: Dissertação (mestrado). *UFPB Universidade Federal da Paraíba*, 2015. Citado na página 17.

TAKEDA, R. Y.; et al. Identificação biométrica de indivíduos baseada em parâmetros dinâmicos extraídos de marcha humana. *Departamento de Engenharia Elétrica, Instituto de Ciências Exatas e Aplicadas, UFOP Universidade Federal de Ouro Preto*, 2012. Citado na página 17.

MATOS, H. J. da S. Reconhecimento biométrico baseado na geometria da mão: Dissertação (mestrado). *Engenharia Eletrotécnica e de Computadores, Major Automação, FEUP Faculdade de Engenharia da Universidade do Porto*, 2011. Citado na página 17.

FREITAS, C. C. S. de. Um estudo de reconhecimento de linhas palmares utilizando pca e limiarização local adaptativa: Dissertação (mestrado). *Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas*, 2014. Citado na página 18.

SALOMAO, J. M. Implementação de métodos biométricos bi-modais baseados em fusão de características para reconhecimento de indivíduos: Dissertação (doutorado). *Engenharia Elétrica, Centro Tecnológico da Universidade Federal do Espírito Santo*, 2007. Citado na página 18.

FIGUEIREDO, R. M. G. Deep recognition-by-tracking in a robot environment: Dissertação (mestrado). *Electrical and Computer Engineering, Técnico Lisboa*, 2018. Citado na página 18.

HIDALGO, G.; et al. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. disponível em <https://github.com/cmu-perceptual-computing-lab/openpose/blob/master>. *CMU-Perceptual-Computing-Lab*, 2019. Citado na página 19.

Learn OpenCV. Deep learning based human pose estimation using opencv cpp python (c++ / python) <https://www.learnopencv.com/deep-learning-based-human-pose-estimation-using-opencv-cpp-python>, acessado em 05/12/2019. *BIG VISION LLC*, 2018. Citado 3 vezes nas páginas 8, 19 e 20.

OpenCV.org. Documentação da biblioteca opencv, disponível em <https://docs.opencv.org/4.1.2/d1/dfb/intro.html>. *OPENCV TEAM*, 2019. Citado na página 22.

HAYKIN, S. *Redes Neurais, princípios e prática 2ª Edição. Tradução: Paulo Martins Engel*. [S.l.]: Bookman - Porto Alegre, 2001. Citado 5 vezes nas páginas 23, 24, 26, 27 e 29.

Keras. Documentação keras disponível em <https://keras.io/>, acessado em 17/12/2019. KERAS TEAM, 2019. Citado 3 vezes nas páginas 8, 29 e 30.

Wikipedia.org. Homem vitruviano, disponível em [https://pt.wikipedia.org/wiki/homem_vitruviano_\(desenho_de_leonardo_da_vinci\)](https://pt.wikipedia.org/wiki/homem_vitruviano_(desenho_de_leonardo_da_vinci)), acessado em 17/12/2019. Citado 2 vezes nas páginas 8 e 31.