UNIVERSIDADE FEDERAL FLUMINENSE

CAIO DE ANDRADE MEDEIROS

ANÁLISE DE VIÉS E VARIÂNCIA EM ALGORITMO PARA RECONHECIMENTO DE EXPRESSÕES FACIAIS

RIO DE JANEIRO

2023

CAIO DE ANDRADE MEDEIROS

ANÁLISE DE VIÉS E VARIÂNCIA EM ALGORITMO PARA RECONHECIMENTO DE EXPRESSÕES FACIAIS

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Orientador(a): Professor Dr. Rogerio Hart

RIO DE JANEIRO

2023

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

M488a Medeiros, Caio de Andrade
ANÁLISE DE VIÉS E VARIÂNCIA EM ALGORITMO PARA RECONHECIMENTO DE EXPRESSÕES FACIAIS / Caio de Andrade Medeiros. - 2023.
56 f.: il.

Orientador: Rogerio Costa Hart.
Trabalho de Conclusão de Curso (graduação)-Universidade Federal Fluminense, Instituto de Computação, Niterói, 2023.

1. Inteligência Artificial. 2. Reconhecimento de Expressão Facial. 3. Viés. 4. Variância. 5. Produção intelectual. I. Hart, Rogerio Costa, orientador. II. Universidade Federal Fluminense. Instituto de Computação.III. Título.

CDD - XXX

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

CAIO DE ANDRADE MEDEIROS

ANÁLISE DE VIÉS E VARIÂNCIA EM ALGORITMO PARA RECONHECIMENTO DE EXPRESSÕES FACIAIS

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Este trabalho foi defendido e aprovado pela banca em __/__/____.

BANCA EXAMINADORA

_____

Prof. Dr. Rogerio Costa Hart – Orientador

UFF - Universidade Federal Fluminense

_____

Prof Dr. Rogerio Costa Hart – Avaliador

UFF - Universidade Federal Fluminense

_____

Prof. Dr. Leandro Augusto Frata Fernandes – Avaliador

UFF – Universidade Federal Fluminense

Dedico este trabalho a todos(as) que já tenham sido vítimas de algoritmos de detecção facial.

AGRADECIMENTOS

Agradeço a todos os professores que proporcionaram a base de conhecimento para o desenvolvimento deste trabalho, e à toda minha família, que foi a principal estrutura que me sustentou para que eu chegasse até aqui.

“A criação bem-sucedida de inteligência artificial seria o maior evento da história da humanidade. Infelizmente, pode também ser o último, a menos que aprendamos a evitar os riscos”.

Stephen Hawking

RESUMO

Os algoritmos de reconhecimento facial em imagens e vídeos podem conter problemas relacionados ao viés e à variância que são provocados pela falta ou pelo excesso de imagens de treinamento. Para que possam aprender de forma satisfatória a detectar as classes de imagens de interesse que são treinados para classificar, os algoritmos de Aprendizado de Máquina podem utilizar mais imagens de treino ou mais iterações sobre estas imagens, assim seu viés pode ser reduzido, mas a consequência disso pode ser um aumento na sua variância à medida que este modelo se torna superajustado. Existem alguns métodos e métricas para identificar e resolver estes problemas, e ao ajustar um modelo, o principal objetivo de um desenvolvedor é encontrar um ponto de equilíbrio de confiança para que o modelo possa ser aplicado a novas detecções. O presente trabalho tem como objetivo principal avaliar se um algoritmo de reconhecimento de expressões faciais em imagens e vídeos, baseado em Redes Neurais, pode conter problemas relacionados ao viés e à variância, provocados pela falta ou pelo excesso de imagens de treinamento, e se estes problemas podem ser resolvidos utilizando técnicas que encontrem um ponto ideal onde este modelo possui uma melhor acurácia.

Palavras-chave: Inteligência Artificial, Reconhecimento de expressão facial, Viés e Variância.

ABSTRACT

Facial recognition algorithms on images and videos can have problems related to bias and variance that are caused by too little or too many training images. So that it can satisfactorily learn to detect the classes of images of interest that are trained to classify, Machine Learning algorithms can use more training images or more iterations over these images, so their bias can be reduced, but the consequence this could be an increase in your variance as this model becomes overfit. There are some methods and metrics to identify and solve these problems, and when tuning a model, a developer's main goal is to find a confidence break-even point so that the model can be applied to new detections. The main objective of this work is to evaluate whether an algorithm for recognizing facial expressions in images and videos, based on Neural Networks, may contain problems related to bias and variance, caused by the lack or excess of training images, and whether these problems can be solved using techniques that find an ideal point where this model has a better accuracy.

Keywords: Artificial Intelligence, Facial expression recognition, Bias and Variance.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

RNA	Rede Neural Artificial
RNC	Rede Neural Convolucional
UFF	Universidade Federal Fluminense
FN	Falso Negativo
FP	Falso Positivo
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

LISTA DE SÍMBOLOS

Σ	Somatório
σ	Função sigmoide
e	Número de Euller
b	*Bias* (viés)
w	*Weight* (Peso)

SUMÁRIO

1 INTRODUÇÃO

A Inteligência Artificial [IA] é um campo de conhecimento que começou a se desenvolver após a segunda guerra mundial e seus principais conceitos foram baseados na teoria da computação proposta por Alan Turing. Com a publicação do artigo "*Computing Machinery and Intelligence*", Turing fez a sua mais importante contribuição para a evolução dos estudos neste tema, onde projetou um teste que avaliaria se um computador seria inteligente o suficiente para responder a perguntas como um ser humano, este método de avaliação foi chamado de Teste de Turing (RUSSEL, Stuart; NORVIG, Peter; 2013, p. 24).

Atualmente, a evolução da IA fez surgir aplicações que alavancaram a capacidade dos computadores para realizar atividades que antes eram feitas de forma eficaz apenas por seres humanos e alguns animais, como distinguir objetos presentes no mundo real. A evolução desta capacidade computacional foi provocada pela disruptura tecnológica proporcionada por uma subárea da IA chamada de Aprendizado de Máquina ou "*Machine Learning*", que utiliza técnicas de análise confirmatória e exploratória para aprender a identificar padrões e realizar determinadas tarefas.

No aprendizado de máquina, a análise exploratória tem como objetivo encontrar padrões e tendências nos dados sem uma hipótese prévia, sendo útil para identificar correlações e padrões em grandes conjuntos de dados para criar modelos preditivos ou classificatórios. Já a análise confirmatória usa uma hipótese prévia e um conjunto de regras para testar e validar essa hipótese, sendo geralmente utilizada para validar modelos e avaliar a qualidade do ajuste do modelo aos dados. (HASTIE, T., TIBSHIRANI, R., Friedman, J, 2013).

Algoritmos de Aprendizagem de Máquina são capazes de "aprender" de duas maneiras: através da aprendizagem não supervisionada, onde realiza análises exploratórias com o objetivo de detectar características semelhantes em algum tipo de dado e da aprendizagem supervisionada, que realiza análises confirmatórias ou preditivas de algum tipo de fenômeno, evento ou objeto de estudo. O Aprendizado Profundo ou "*Deep Learning*" é considerado uma vertente de técnicas de Aprendizado

de Máquina e utiliza Redes Neurais Artificiais [RNA's] com múltiplas camadas para identificar padrões nos dados de entrada do modelo, e tentar prever algum tipo de saída como resposta.

Os algoritmos de Aprendizado Profundo utilizam dados, que servem como exemplos, para que através destas informações, o computador possa aprender, prever e classificar alguma informação. Estes algoritmos utilizam RNA's[1] que contém um modelo de regressão ou classificação de duas etapas, geralmente representado por um diagrama de rede. Essa rede se aplica tanto à regressão quanto à classificação. (HASTIE T., TIBSHIRANI R. Friedman J., 2013, p. 392)

1.1 DETECÇÃO FACIAL

Os primeiros sistemas automatizados de reconhecimento facial foram desenvolvidos na década de 1970, e inicialmente eles utilizavam algoritmos de reconhecimento baseados em técnicas simples de correspondência de padrões que comparavam características únicas de cada rosto, como a posição dos olhos e o formato do nariz para identificar uma pessoa. (Z. Li, Stan; JAIN, Anil K, 2011).

Com o avanço da tecnologia, surgiram técnicas mais sofisticadas baseadas em redes neurais e aprendizado de máquina. Essas técnicas permitem que o algoritmo aprenda a identificar características faciais complexas e assim, seja capaz de reconhecer uma pessoa com maior precisão (Z. Li, Stan; Jain, Anil K, 2011).

Uma das principais limitações dos primeiros algoritmos de reconhecimento facial era a dificuldade em lidar com variações na iluminação e na posição da face. No entanto, os algoritmos modernos são capazes de compensar essas variações, usando técnicas como iluminação de imagem, rotação de imagem e alinhamento de face.

Além disso, os algoritmos de reconhecimento facial agora são capazes de lidar com grandes bancos de dados de imagens faciais, permitindo uma identificação mais

[1] As RNA's são um modelo preditivo baseado na forma como o cérebro funciona. Estas RNAs podem resolver problemas como reconhecimento de caligrafia e detecção facial em imagens e vídeos, são muito usadas em Aprendizado Profundo [GRUS; 2016], utilizando dados que treinam o algoritmo para realizar estas predições.

rápida e precisa. Isso tem sido usado em diversas aplicações, desde a segurança pública até a identificação de pacientes em hospitais.

No entanto, estas grandes bases de dados que servem de insumo para os algoritmos de reconhecimento facial geram preocupações sobre a acurácia destes sistemas, especialmente em relação à identificação equivocada de pessoas de minorias étnicas e raciais quando as imagens utilizadas são adquiridas em ambientes não controlados. Este foi um problema que ocorreu em um projeto-piloto da Secretaria de Estado de Polícia Militar do Rio de Janeiro, onde admitiu que dentre onze casos de pessoas detidas com o uso da tecnologia de reconhecimento facial nas partidas do Maracanã, sete foram erros da máquina, ou seja: falsos positivos. Desta forma, o sistema errou em 63% dos casos[2].

Este problema de detecções com resultado falso-positivo é um desafio abordado pela comunidade científica e pela indústria, com o objetivo de garantir que os algoritmos de reconhecimento facial sejam precisos ao identificar qualquer pessoa ou objeto de interesse.

Estas preocupações residem no fato do Aprendizado Profundo ser uma técnica de Aprendizado de Máquina Supervisionado, necessitando que os exemplos sejam fornecidos por um desenvolvedor para “ensinar” o computador a prever e classificar informações extraídas de alguma imagem de entrada. A fase inicial de treinamento é uma das mais importantes, e envolve riscos de falha humana.

As imagens de entrada, utilizadas pelo algoritmo em seu treinamento precisam ser imagens de qualidade, ou seja, devem possuir iluminação, alinhamento, resolução adequadas. No entanto, este processo é passível de erros caso não seja bem realizado e segundo (NG, 2018), estes erros podem provocar dois tipos de problemas, o viés e a variância, que são causados por falhas distintas provocadas pelo desenvolvedor do algoritmo. O aumento de um destes erros pode reduzir o outro, o que caracteriza estes erros como um “*trade-off*”, ou dilema que deve ser ajustado para não tornar o algoritmo ineficaz.

---

[2]. EBC, 2022. Disponível em: https://agenciabrasil.ebc.com.br/geral/noticia/2022-12/entidades-acionam-mp-contra-reconhecimento-facial-na-capital-paulista

Em um viés algorítmico, o modelo desempenha de modo insatisfatório a interpretação dos dados de treinamento. Isso ocorre porque o modelo não consegue capturar o relacionamento entre poucos exemplos de entrada provocando uma saída incoerente.

Enquanto o problema do viés algorítmico está relacionado com uma quantidade insuficiente de dados de entrada, o problema da variância está relacionado com algoritmos que possuem muitos dados de entrada e que são muito complexos (SKANSI, Sandro 2018 p.107). Esta grande quantidade de exemplos de classes que podem ser pessoas, animais, veículos, plantas, dentre outros objetos com características físicas únicas, podem fazer com que o algoritmo reduza o erro na detecção, mas caso surjam novos objetos para serem detectados este mesmo algoritmo, “que não quer errar” não será capaz de fazer uma nova predição pois ela terá uma alta parcela de chance de erro na classificação de saída.

Para testar a acurácia de um algoritmo de visão computacional, que realize o reconhecimento de expressões faciais, um vídeo pode ser considerado como um exemplo de dado não estruturado de entrada, que é capaz de oferecer uma grande amostra de frames para a análise da acurácia deste algoritmo.

São considerados dados não estruturados os que não têm uma estrutura pré-definida. Eles não se enquadram nos modelos de dados relacionais, orientados a objetos ou hierárquicos. Por exemplo, documentos textuais, vídeos e imagens serão considerados não estruturados devido à sua forma de organização, diversidade, tamanho e principalmente por cada tipo não estruturado conter diferentes dados e informações (SAES, Keylla Ramos, 2019, p. 17).

Como exemplo de treinamento, pode-se considerar um algoritmo que visa identificar expressões faciais de pessoas em um ambiente. Este ambiente pode ser aberto, como em uma rua ou campo, ou fechado como uma sala. Para que o algoritmo seja capaz de realizar o reconhecimento destas expressões em qualquer um destes ambientes, ele deve ser treinado com o maior número possível de imagens de diferentes expressões faciais, de pessoas de todas as etnias, gêneros, idades, alturas, dentre outras características inerentes à diversidade dos seres humanos.

Caso este algoritmo seja treinado com exemplos que não considerem esta diversidade, sua acurácia pode não ser satisfatória em aplicações que exigem acurácia, como no caso deste algoritmo ser utilizado na produção de carros inteligentes, sistemas de reconhecimento por imagens em câmeras de vigilância, dentre outras aplicações. Esta falha pode provocar um viés e/ou uma variância algorítmica, que pode causar acidentes em veículos autônomos incapazes de distinguir um objeto à sua frente, falhas em cadastros que utilizam identificação facial, discriminação e injustiças provocados por câmeras de reconhecimento, dentre outros problemas causados por modelos imprecisos.

Visando encontrar uma solução para o problema do viés e variância algorítmica, o presente trabalho tem como objetivo principal testar e avaliar um algoritmo de detecção de expressões faciais em imagens e vídeos, baseado em Redes Neurais, para identificar se este modelo possui tais problemas provocados pela falta ou pelo excesso de dados de treinamento.

O objetivo do presente trabalho pretende ser alcançado com a realização de treinamentos em um modelo de *Deep Learning* customizado, utilizando diferentes quantidades de imagens de treino e de teste. Estes treinamentos visam identificar se o aumento na amostra de imagens, melhoraria a acurácia e reduziria o viés do modelo ao detectar as expressões faciais do autor deste trabalho.

1.2 OBJETIVO

Com o intuito de testar e analisar a acurácia de um algoritmo de detecção de expressões faciais em tempo real, este trabalho irá avaliar se esse algoritmo possui algum tipo de problema de viés ou de variância. Após esta avaliação, caso o problema seja identificado, serão propostas soluções para o problema do viés e da variância algorítmica, e se estes problemas seriam provocados por falta de amostras representativas ou pelo excesso de dados que servem de insumo para o treinamento do algoritmo de Deep Learning.

Além disso, o presente Trabalho de Conclusão de Curso avaliará o algoritmo de Deep Learning utilizado para realizar reconhecimento facial em vídeos e imagens,

com o objetivo de identificar se existe algum viés que gere preconceito na detecção de pessoas em variados tipos de ambientes.

Como este algoritmo será desenvolvido na plataforma “Teachable Machine” utilizando imagens coletadas por uma câmera pessoal, caso seja identificado algum tipo de viés, esta biblioteca de imagens será melhorada. No entanto, este aumento de exemplos de dados de entrada não deverá gerar o problema de variância, que é causado pelo excesso de dados de entrada.

Durante os testes deste algoritmo, será criada uma base de dados com todas as classes de expressões faciais detectadas, assim como a probabilidade de acerto para cada uma das classes de expressões faciais detectadas. Esta base de dados servirá como insumo para análise da acurácia do algoritmo de visão computacional em distinguir as expressões faciais em tempo real.

Após finalizar os testes, serão analisadas as imagens das expressões faciais que não sejam identificados com eficácia e situações de viés e variância algorítmica. Listando o motivo pelo qual os objetos não são identificados para então propor soluções para melhorar a acurácia do algoritmo. Dentre estas soluções estão a utilização de mais dados de treino, para comprovar a hipótese de que quanto mais dados forem utilizados para treinar os algoritmos maior pode ser a sua acurácia e menor será o viés algorítmico para identificar um determinado objeto.

1.3 MATERIAL E MÉTODOS

As imagens da minha face foram detectadas através de um algoritmo desenvolvido na plataforma Teachable Machine (https://teachablemachine.withgoogle.com/), que correlaciona objetos detectados em vídeos com as imagens de treino e teste capturadas pela própria plataforma. A detecção facial ocorreu em tempo real através de uma câmera acoplada ao Notebook.

Ao detectar a classe de expressão facial de interesse, o algoritmo da plataforma também estima a probabilidade da expressão facial detectada ser realmente o que está na biblioteca de classes. Após realizar os testes, cada expressão detectada e

suas respectivas probabilidades foram analisadas para classificar em quais condições foram detectadas com mais facilidade, e quais não tiveram uma acurácia satisfatória, e se existe viés e/ou variância algorítmica na detecção.

Os dados foram coletados e analisados, e os resultados exibidos em gráficos, onde os objetos detectados foram agrupados em observações semelhantes entre si. Além disso, foram elaborados gráficos para evidenciar graficamente como o aumento ou a diminuição de amostras de entrada para treinamento do algoritmo podem influenciar no aumento ou na diminuição deste problema.

Também foram utilizados métodos de estatística descritiva como análise de distribuição de frequências para cada uma das classes de objetos. Todos os gráficos foram desenvolvidos utilizando o Excel.

2 REVISÃO TEÓRICA

Este capítulo apresenta os conceitos de Aprendizado de Máquina e Aprendizado Profundo. Para os fins deste trabalho o Aprendizado de Máquina será dividido entre duas grandes categorias: em Aprendizado supervisionado e Aprendizado não supervisionado, apesar de também existirem outras duas categorias como Aprendizado por reforço e Aprendizado semi-supervisionado (HASTIE T, TIBSHIRANI R. & Friedman J, 2013, p. 808). O Aprendizado Profundo é uma abordagem que utiliza redes neurais artificiais para resolver problemas complexos, simulando processos cognitivos humanos. Com o desenvolvimento da capacidade de processamento dos computadores, as Redes Neurais Artificiais evoluíram para modelos de Redes Neurais Profundas, com várias camadas intermediárias de neurônios e conexões ponderados por pesos e vieses.

Os neurônios nas camadas ocultas resolvem problemas matemáticos e tarefas de classificação de forma análoga ao processo de resolução de equações como por exemplo:

$$x_1 + x_2 + x_3 = 7{,}5 \quad (1)$$

Na equação 1, a única informação conhecida é a saída que deve ser 7,5. Para descobrir as entradas, é possível deduzir números como 3, 2 e 0 e colocá-los na equação, obtendo 5 como resposta. No entanto, o objetivo é obter 7,5, e uma boa estratégia é multiplicar as entradas por um número conhecido como peso w_i. Assim, é possível iniciar multiplicando as suposições originais por 2, obtendo:

$$(3 \times 2) + (2 \times 2) + (0 \times 2) = 10 \quad (2)$$

Depois de descartar esta saída, é necessário ajustar os pesos para baixo, iniciando um novo ciclo de ajuste dos pesos que é conhecido como *Epoch* ou Época

que o algoritmo realiza para ajustar sua saída. Uma rede neural usa o valor de “erro” ou viés b, para ajustar os pesos w_i de acordo com o objetivo da predição em um processo chamado “*backpropagation*”.

Nesta equação, o viés b e os pesos w são parâmetros gerados a partir de variáveis aleatórias, e para simplificar o modelo, o viés pode ser zerado, considerando que seu valor é absorvido pelo valor dos pesos. Desta forma, utilizando o peso 1.5 e assumindo o intercepto da equação $\mathrm{b} = 0$ é possível finalmente obter as entradas desejadas para chegar ao resultado:

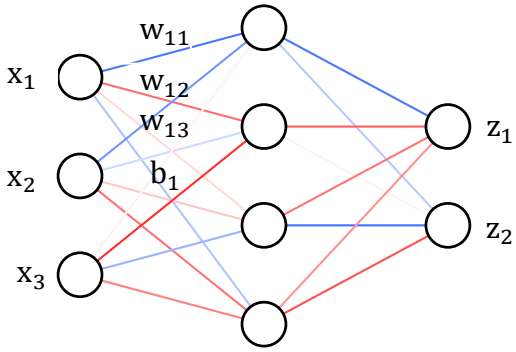
$$(\mathrm{x}_1 \times \mathrm{w}_{11}) + (\mathrm{x}_2 \times \mathrm{w}_{12}) + (\mathrm{x}_3 \times \mathrm{w}_{13}) + \mathrm{b}_1 = 7.5 \qquad (3)$$

$$(3 \times 1{,}5) + (2 \times 1{,}5) + (0 \times 1{,}5) + 0 = 7.5 \qquad (4)$$

Portanto, apesar de não conhecer as entradas originais é possível escolher valores aleatórios que multiplicados por pesos, possibilitam o modelo obter uma saída desejada, e é assim que as redes neurais simples funcionam.

O desenvolvimento da capacidade de processamento dos computadores atuais, criou avanços significativos na velocidade das operações matemáticas, possibilitando a elaboração de modelos mais complexos e por consequência, mais precisos. Como uma Rede Neural é composta por camadas intermediárias com conexões ponderadas por pesos e vieses, sua capacidade de desenvolver modelos precisos está na quantidade de conexões entre estes neurônios artificiais, e à medida em que várias camadas intermediárias são utilizadas, a complexidade do modelo pode tornar o treinamento da rede neural mais difícil. Portanto, o uso de várias camadas intermediárias leva à convergência para modelos de Redes Neurais Profundas.

Figura 1 - Rede Neural Simples

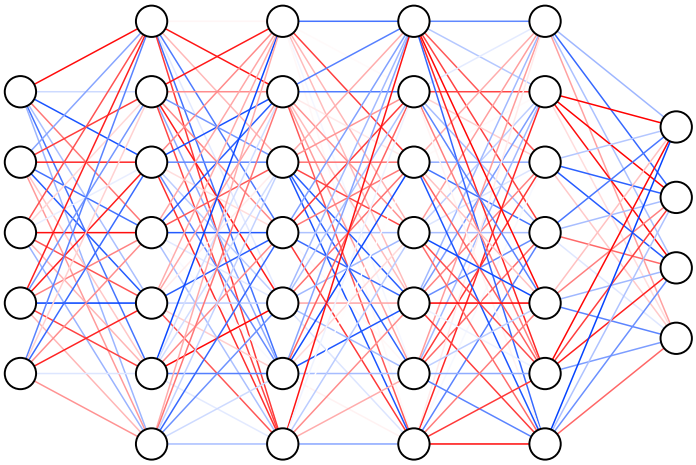


Fonte: Elaboração própria desenvolvida em: http://alexlenail.me/NN-SVG/index.html

Na estrutura de uma rede neural simples de três camadas mostrada na figura acima, cada neurônio de uma camada está conectado a todos os neurônios da próxima camada, que são multiplicados por pesos e vieses que determinam o valor que deve ser transmitido para o neurônio da próxima camada [SKANS, 2018, p. 80].

Este modelo simples de apenas uma camada intermediária não é tão eficiente para realizar operações complexas como o reconhecimento de objetos em imagens, que é uma operação que requer a utilização de mais camadas de neurônios.

Figura 2 - Rede Neural Profunda



Fonte: Elaboração própria desenvolvida em: http://alexlenail.me/NN-SVG/index.html

O que diferencia uma Rede Neural Artificial Simples de uma Rede Neural Profunda é a quantidade de camadas, sendo maior no segundo caso. A rede neural da figura 2 possui 5 neurônios na camada de entrada, 7 neurônios em cada uma das 4 camadas ocultas e 4 neurônios na camada de saída. Portanto, esta rede pode ser caracterizada por uma Rede Neural Profunda pois possui várias camadas ocultas.

2.1 REDES NEURAIS CONVOLUCIONAIS

Uma Rede Neural Convolucional [RNC] ou *Convolutional Neural Network* [CNN], é um algoritmo de Aprendizado Profundo que é utilizado como principal técnica para detecção facial em vídeos e imagens. Uma RNC é uma classe de rede neural artificial do tipo *feed-forward*[3] que atualmente vem sendo aplicada com sucesso no processamento e na análise de imagens digitais.

Uma RNC composta por apenas uma camada convolucional utiliza uma técnica de Aprendizado Profundo que capta uma imagem de entrada, atribuindo pesos w e vieses b a partes da imagem para diferenciar estas partes umas das outras, e desta forma, fornece uma interpretação da importância de cada uma das características de uma imagem para realizar uma classificação.

Esta classificação de imagens de entrada é realizada através da transformação de uma imagem em matrizes e vetores, onde cada elemento deste vetor é processado por uma rede neural que utiliza uma regressão logística[4] para realizar uma predição para estes dados de entrada. Portanto, a regressão logística é um algoritmo de aprendizado supervisionado que pode ser dividida em duas equações:

$$z = b + w_1 x_1 + w_2 x_2 + w_3 x_3 \qquad (5)$$

[3] Uma rede neural é uma rede de neurônios artificiais interconectados. As redes neurais convolucionais utilizam redes do tipo 'feed-forward', o que significa que os dados passam por eles apenas em uma direção.
[4] [RUSSEL, Stuart; NORVIG, Peter. p. 121.]

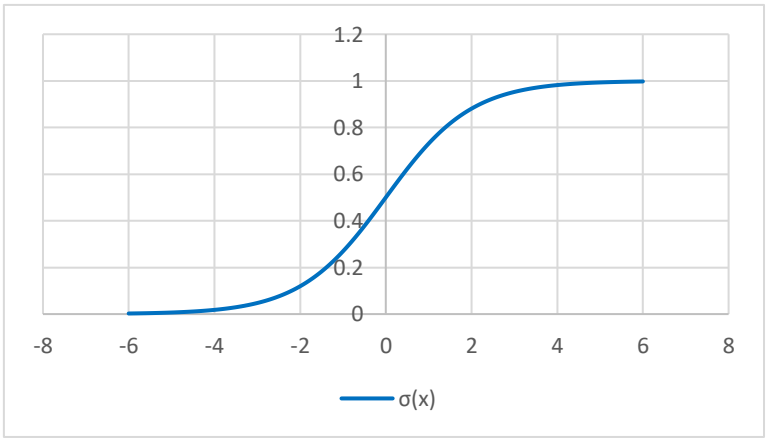
A equação acima é uma soma ponderada que calcula o valor *Logit* z a partir das entradas dos dados de uma imagem. Estas entradas são denotadas por x, e a saída é denotada por z. Para calcular o *Logit*, são necessários os parâmetros de pesos w e do viés b. Nesta equação, estes parâmetros não são dados como as entradas, e são gerados a partir de variáveis aleatórias gaussianas (SKANSI, Sandro 2018 p. 64). No entanto, para simplificar o modelo, o viés pode ser considerado 0, considerando que seu valor é absorvido pelo valor dos pesos.

A segunda equação é representada por uma função logística também conhecida como função sigmóide. Esta função pode ser representada da seguinte forma:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (6)$$

A equação 5 representa uma função não-linear descrita acima com a característica de ter um estágio inicial de crescimento exponencial, e uma posterior redução no crescimento após o intercepto de x = 0.

Gráfico 1 - Curva Sigmóide



Fonte: Elaboração própria

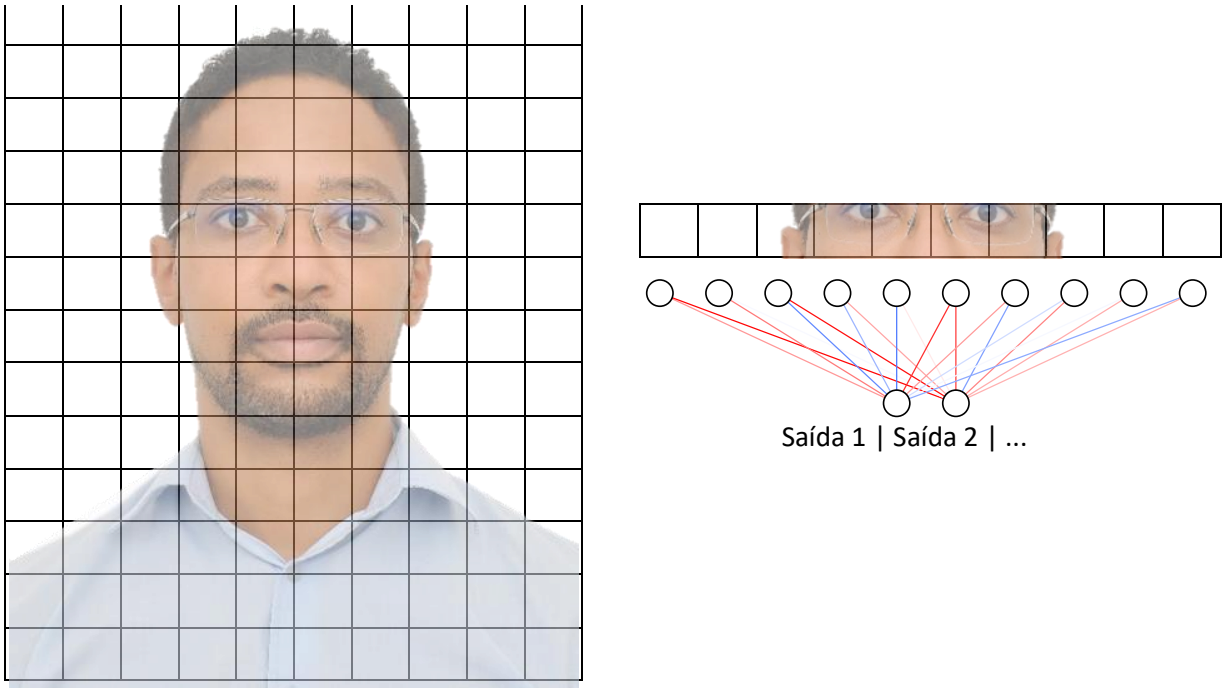
Em modelos de redes neurais, a regressão logística é utilizada quando o objetivo é classificar uma variável qualitativa, ou seja, uma variável dependente

categórica ou não métrica que possua apenas dois grupos ou classificações como resultados possíveis como, por exemplo, branco ou preto, azul ou marrom, alto ou baixo, homem e mulher, sim ou não etc.

Neste sentido, a regressão logística estima a probabilidade de ocorrência de um evento, assumindo que a variável dependente categórica $f(z)$ é a saída, sendo um número entre 0 e 1, e que pode ser interpretada como uma probabilidade de pertencer à classe rotulada (RUSSEL, Stuart, NOVIG, Peter, pág. 841).

O cérebro humano funciona de forma similar para classificar algum tipo de imagem, sendo assim, a arquitetura de uma RNC é análoga ao padrão de conectividade de neurônios humanos onde existe uma resposta a estímulos visuais, com a extração de características de uma determinada imagem de entrada, para uma posterior classificação desta imagem a partir das suas características, finalizando com a saída da possível classe desta imagem de entrada.

Figura 3 - Construindo uma camada convolucional utilizando regressão logística



Fonte: Dados originais da pesquisa

No exemplo acima, a camada convolucional recebe a imagem com tamanho de

entrada igual a 10, e utiliza uma regressão logística por toda a imagem[5]. Isso significa que a primeira entrada consiste nos componentes numerados 1-10 do vetor, a segunda entrada são os componentes 2-11, a terceira são os componentes 3-12 e assim por diante até chegar ao componente 41-50 que contém os pixels da região da orelha, olhos e nariz, partes importantes para o aprendizado do modelo. Esse processo cria vetores de saída que são menores do que o vetor de entrada.

Neste exemplo, um componente foi movido de cada vez, mas também poderia ser movido por dois ou mais campos. Estas variações de movimentações mais rápidas ou mais lentas são chamadas de passo da camada convolucional.

Nesta técnica, a extração dos dados realizada pela regressão logística, escolhe a cada passo, quais partes da imagem será usada para classificação e cria um resultado que será comparado com a classe de interesse. Ou seja, matematicamente, a convolução é uma operação de somatório do produto entre duas funções, ao longo da região em que elas se sobrepõem.

$$(f \times g)(x, y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i, j) g(x - i, y - j) \quad (7)$$

A equação 7, indica que f e g são entradas de imagens de dimensão $\mathrm{i} \times \mathrm{j}$, onde é somado o resultado da multiplicação de cada posição da área de interesse por uma área correspondente a partir da posição x e y da entrada, obtendo como resultado uma resposta de ativação. Segundo (SKANSI, Sandro, 2018, p. 123), este é o modelo clássico para redes neurais convolucionais, e as camadas contidas neste modelo são chamadas de camadas convolucionais 2D ou camadas convolucionais planares.

Em uma rede neural convolucional de três camadas convolucionais e uma camada totalmente conectada, caso ela esteja processando uma imagem de tamanho

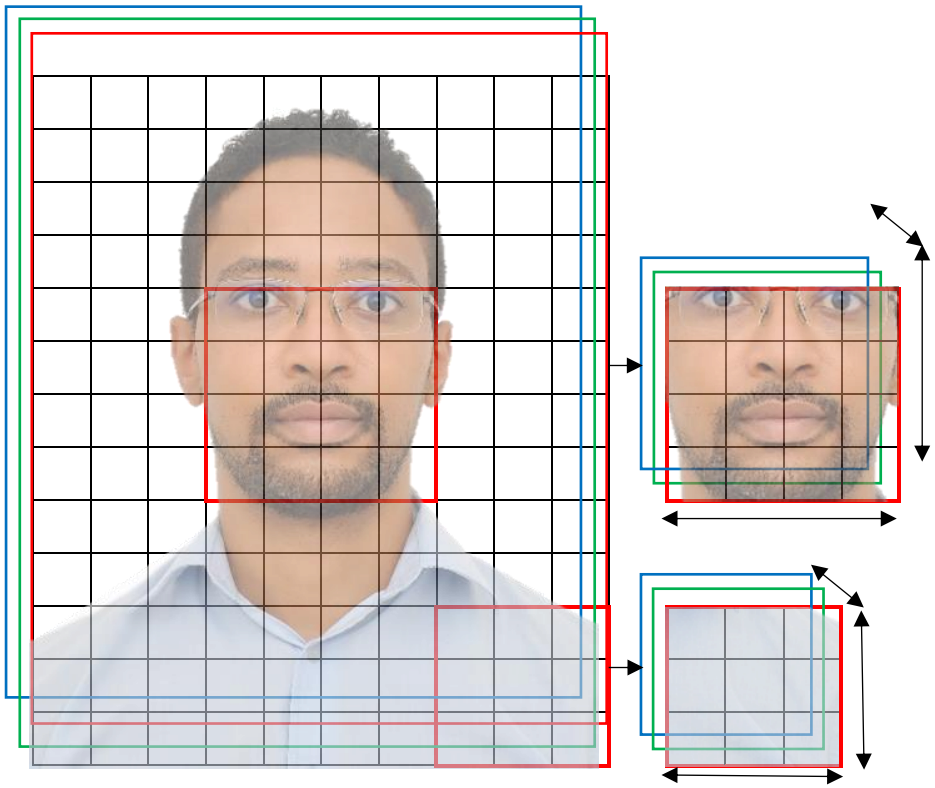
[5] Uma imagem neste sentido é qualquer matriz com cada pixel variando entre valores contidos no intervalo entre 0 e 255 nos canais de cores vermelho, verde e azul (RGB). No exemplo, o componente 1 seria representado por pixels com valor decimal RGB para branco (255,255,255). O componente 26 seria representado por pixels com RGB correspondente a um tom de marrom de aproximadamente (160,82,45). [RUSSEL, Stuart; NORVIG, Peter. p.122.]

10 e que todas as três camadas tenham um campo perceptivo local $i \times j$ de tamanho 4 x 4, a função da RNC é decidir qual parte da imagem tem ou não uma face.

Para esta tarefa, diferentemente do modelo anterior, o modelo a seguir é conhecido por ser localmente conectado, e possuir várias camadas convolucionais, onde cada parâmetro local é multiplicado por seus pesos onde este valor não está conectado à imagem inteira. Este modelo que captura blocos locais realiza cálculos convolucionais considerando apenas a região destacada.

Uma RNC multicamada é composta por várias camadas, que são conectadas com diferentes pesos e vieses. Estas camadas possuem altura, largura e profundidade, e neste modelo, os canais de cores das imagens estão na dimensão de profundidade. Portanto, cada camada convolucional significa treinar seus campos receptivos locais com uma função de ativação, e com o conceito de "*max-pooling*", que conforme demonstrada na Figura 4, reduz o tamanho da imagem em blocos menores de pixels agrupados[6].

Figura 4 - Camadas Convolucionais



[6] (SKANSI, Sandro 2018 p. 125).

Fonte: Dados originais da pesquisa

A principal diferença entre uma regressão logística e um campo receptivo local é que em um campo receptivo local é possível usar uma função de ativação, enquanto em uma regressão logística pode usar uma função logística[7]. A função de ativação mais usada é chamada de Unidade Linear Retificada ou ReLU. Um ReLU de x é simplesmente o valor máximo de 0 e x, o que significa que ele retornará 0 se a entrada for negativa ou a entrada bruta caso contrário. Esta função pode ser demonstrada da seguinte forma:

$$p(x) = max(X, 0) \quad (8)$$

Normalmente, uma RNC é composta por uma camada convolucional seguida por uma camada "*max-pooling*", seguida por uma camada convolucional, e assim sucessivamente. A transformação da imagem em uma matriz ou tensor de vários canais acontece nas camadas convolucionais, e posteriormente uma regressão logística é usada para calcular a probabilidade de cada classe de saída após a passagem da imagem pelas camadas convolucionais. Assim são extraídas quais partes da imagem são relevantes para resolver o problema de classificação.

Em uma RNC, a dimensão de profundidade do tensor de entrada corresponde ao número de canais de cores da imagem. Cada canal de cor é uma matriz bidimensional que representa a intensidade da cor em cada ponto da imagem. Portanto, a dimensão de profundidade do tensor de entrada corresponde ao número de canais de cores da imagem, que geralmente é 3 para imagens RGB (*Red*, *Green*, *Blue*).

[7] (RUSSEL, Stuart; NORVIG, Peter. p. 124)

2.2 VIÉS E VARIÂNCIA EM APRENDIZADO DE MÁQUINA

A quantidade média de erros de um modelo de Aprendizado de Máquina ao capturar a relação entre variáveis de uma classe em que é realizada uma predição é chamada de viés, ou *bias* em inglês. (SKANSI, Sandro 2018 p.36). O erro total que ocorre durante um treinamento do modelo de Aprendizado de Máquina é composto pelos erros de viés (bias) e de variância. Durante o treinamento, o que geralmente ocorre é que, à medida que o modelo aprende com os dados de treinamento, tanto o erro de viés quanto o erro de variância são reduzidos, melhorando a capacidade de generalização do modelo. Portanto, é esperado que o erro total diminua gradualmente à medida que o modelo se ajusta aos dados.

No entanto, se o erro de treinamento diminuir, enquanto o erro de validação (ou erro em dados nunca vistos) aumenta, pode indicar que o modelo está sofrendo de *overfitting*, ou seja, o modelo está se tornando muito específico para os dados de treinamento e não generaliza bem para novos dados. Esta situação pode ser observada como um aumento na variância do modelo.

A variância é a capacidade de um modelo para realizar predições futuras em dados diferentes dos utilizados durante o treinamento. Se o modelo de aprendizado de máquina é muito sensível aos dados de treinamento, ou seja, identificou tão bem a relação entre os dados de treinamento que quando é testado irá errar ao não conseguir realizar uma estimativa para dados nunca vistos.

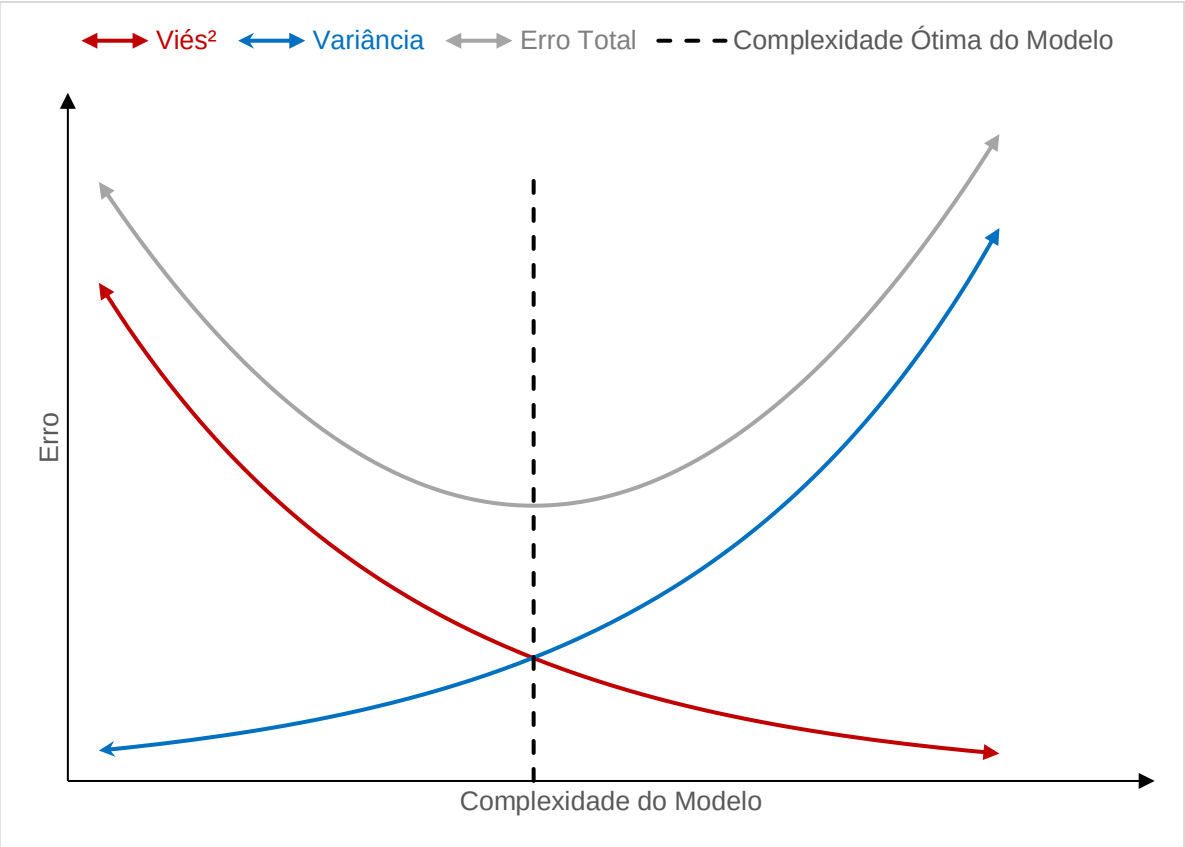
Neste sentido, um alto viés indica uma alta quantidade de erros, o que significa que o modelo não aprendeu de forma satisfatória a realizar predições. Uma alta variância significa que o modelo não aprendeu o suficiente para conseguir generalizar e detectar classes de dados que não fazem parte do conjunto de dados de treinamento. A presença destes problemas revela quando é necessário rever a forma como foi realizado o treinamento.

No caso inverso, um modelo de Aprendizado de Máquina pode ter uma baixa acurácia para um conjunto de dados treinamento, configurando uma condição de subajuste (*underfitting*). Essa situação pode ocorrer, quando os dados de treinamento

são pouco representativos ou quando o modelo é muito simples e não captura padrões existentes nos dados.

Conforme mencionado anteriormente, alguns modelos aprendem muito bem com o conjunto de treino, mas este tipo de modelo superajustado aos dados de treinamento, apesar de ser altamente preciso é caracterizado por um problema chamado de *Overfitting*. Isso faz com que o modelo esteja do lado direito do Gráfico 2, entretanto, o ideal é que fique mais próximo do meio. Para diminuir a variância um pouco de viés pode ser acrescentado e assim o modelo pode "desaprender" o suficiente para conseguir generalizar e ser utilizado em outros dados de treino.

Gráfico 2 - Viés e variância em função da complexidade do modelo



Fonte: Elaboração própria adaptada de http://scott.fortmann-roe.com/docs/BiasVariance.html

No gráfico 3 à esquerda é possível identificar um modelo delimitado pela linha vermelha, onde todos os pontos representados por x estão sendo detectados

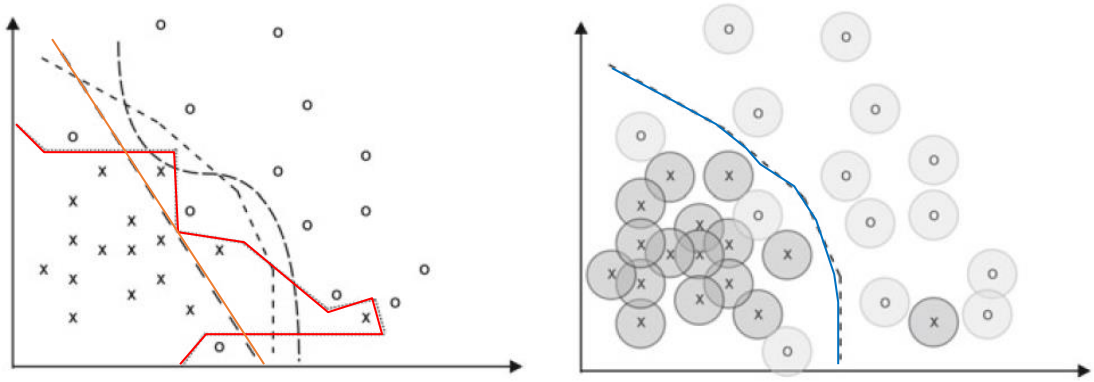
corretamente. Este tipo de detecção é caracterizado por um alto *Overfitting*, Neste exemplo, o modelo acaba por “aprender demais” sobre um determinado aspecto “x” e “aprender de menos” sobre outro aspecto “o”.

Um outro modelo identificado neste mesmo gráfico é o delimitado pela linha laranja, onde é traçada uma linha reta que divide as classes em duas amostras. Este tipo de detecção não é precisa e possui um alto viés.

Um modelo de Aprendizado de Máquina ideal seria algo parecido com o modelo representado pela linha azul do meio do gráfico à direita, gerando uma curva mais generalizada. (SKANSI, Sandro 2018 p.108)

O processo que visa reduzir a variância de um modelo, é chamado de Regularização e pode ser entendido como o processo realizado para ajustar o modelo para que ele consiga realizar generalizações para novos dados não utilizados no treinamento do modelo.

Gráfico 3 - Regularização



Fonte: Adaptado de (SKANSI, Sandro 2018 p.108)

Uma das formas mais comuns de regularização em redes neurais é a adição de um termo de regularização na função de perda da rede neural. Esse termo é uma função dos pesos, e serve para penalizar pesos com altos valores absolutos. A ideia é que, ao adicionar esta penalização, a rede neural seja incentivada a manter pesos menores, o que pode ajudar a evitar o *overfitting*. (FELTRIN, Fernando. p. 225).

Utilizando como exemplo o detector de expressões faciais, é possível fazer uma analogia ao gráfico 3, associando os pontos com “x” a classe Muito Feliz e as outras expressões faciais a cada um dos pontos “o” do gráfico. Sendo assim, a principal tarefa do modelo de Aprendizado de Máquina seria encontrar a classe “Muito Feliz” dentre outras classes de expressões faciais. Para isso, o classificador deve ser capaz de localizar a classe “Muito Feliz” usando as propriedades que são comuns para este tipo de expressão facial, mas que não esteja presente em outros tipos de expressões.

No entanto, segundo (SKANSI, Sandro 2018 p.57), não é uma tarefa fácil encontrar todas as propriedades comuns a uma determinada classe de interesse como uma expressão facial. Por isso, o que um algoritmo de Aprendizado de Máquina faz é encontrar uma propriedade que defina uma classe da melhor forma possível incluindo apenas os parâmetros relevantes na definição de uma classe.

Neste sentido, parâmetros não adequados usados na modelagem podem ser excluídos através de regularizações do modelo, este processo também realiza a escolha dos parâmetros úteis para a modelagem.

3 ARQUITETURA DO MODELO DE APRENDIZADO DE MÁQUINA

O *Teachable Machine* é um aplicativo que pode ser acessado via navegador que possibilita o treinamento de modelos de Aprendizado de Máquina que utilizam imagens capturadas pela webcam para reconhecer sons, objetos ou expressões.

Nesta plataforma on-line não é necessária codificação do usuário, e nenhum hiperparâmetro ou arquitetura de rede deve ser obrigatoriamente definido ou configurado. Todo o treinamento é realizado no navegador utilizando a biblioteca de de código aberto para Aprendizado Profundo *deeplearn.js*. Esta biblioteca criada por uma equipe de desenvolvedores *Google* na linguagem de programação *Javascript* é acelerada por hardware, e está disponível gratuitamente.

O treinamento na plataforma é acelerado através de uma rede neural convolucional pré-treinada chamada "*squeezenet*", que possui 50 vezes[8] menos parâmetros treináveis do que outras redes neurais, mas que preserva uma acurácia satisfatória.

Isso significa que o modelo final que utiliza esta arquitetura pode ser compactado em um arquivo menor que meio megabyte. Embora esta rede neural tenha menos parâmetros, a qualidade da saída é comparável com redes neurais maiores, possibilitando uma boa performance para aplicativos que funcionem em um navegador web. Estas características justificaram a escolha desta arquitetura para o treinamento.

[8] (IANDOLA, Forrest N.; HAN, Song. 2017, p. 1)

4 TREINAMENTO DO MODELO DE APRENDIZADO DE MÁQUINA

O treinamento do modelo de deteção facial para análise de expressões faciais utilizou imagens com diferentes ângulos para possibilitar que cada uma das expressões fosse detectada de forma satisfatória, independentemente do posicionamento da face em relação à câmera no momento dos testes.

Para isso, foram realizados 4 treinamentos que utilizaram respectivamente 50, 100, 300 e 600 fotos contendo diferentes expressões faciais. Além disso, para cada treinamento, foram utilizadas fotos contendo um fundo para que o modelo pudesse diferenciar quando uma face estivesse aparecendo ou não em frente a câmera. Estas classes de imagens foram divididas de acordo com as tabelas abaixo

Tabela 1- Número de imagens utilizadas para teste e validação das classes

Treinamento 1			
CLASSE	**VALIDAÇÃO**	**TREINO**	**TOTAL**
Muito Feliz	2	8	10
Feliz	2	8	10
Neutro	2	8	10
Triste	2	8	10
Muito Triste	2	8	10
Fundo	2	8	10
Total	**12**	**48**	**60**

Treinamento 2			
CLASSE	**VALIDAÇÃO**	**TREINO**	**TOTAL**
Muito Feliz	3	17	20
Feliz	3	17	20
Neutro	3	17	20
Triste	3	17	20
Muito Triste	3	17	20
Fundo	2	8	10
Total	**17**	**83**	**110**

Treinamento 3			
CLASSE	**VALIDAÇÃO**	**TREINO**	**TOTAL**
Muito Feliz	9	51	60
Feliz	9	51	60
Neutro	9	51	60
Triste	9	51	60
Muito Triste	9	51	60
Fundo	2	8	10
Total	**47**	**263**	**310**

Treinamento 4			
CLASSE	**VALIDAÇÃO**	**TREINO**	**TOTAL**
Muito Feliz	18	102	120
Feliz	18	102	120
Neutro	18	102	120
Triste	18	102	120
Muito Triste	18	102	120
Fundo	2	8	10
Total	**92**	**518**	**610**

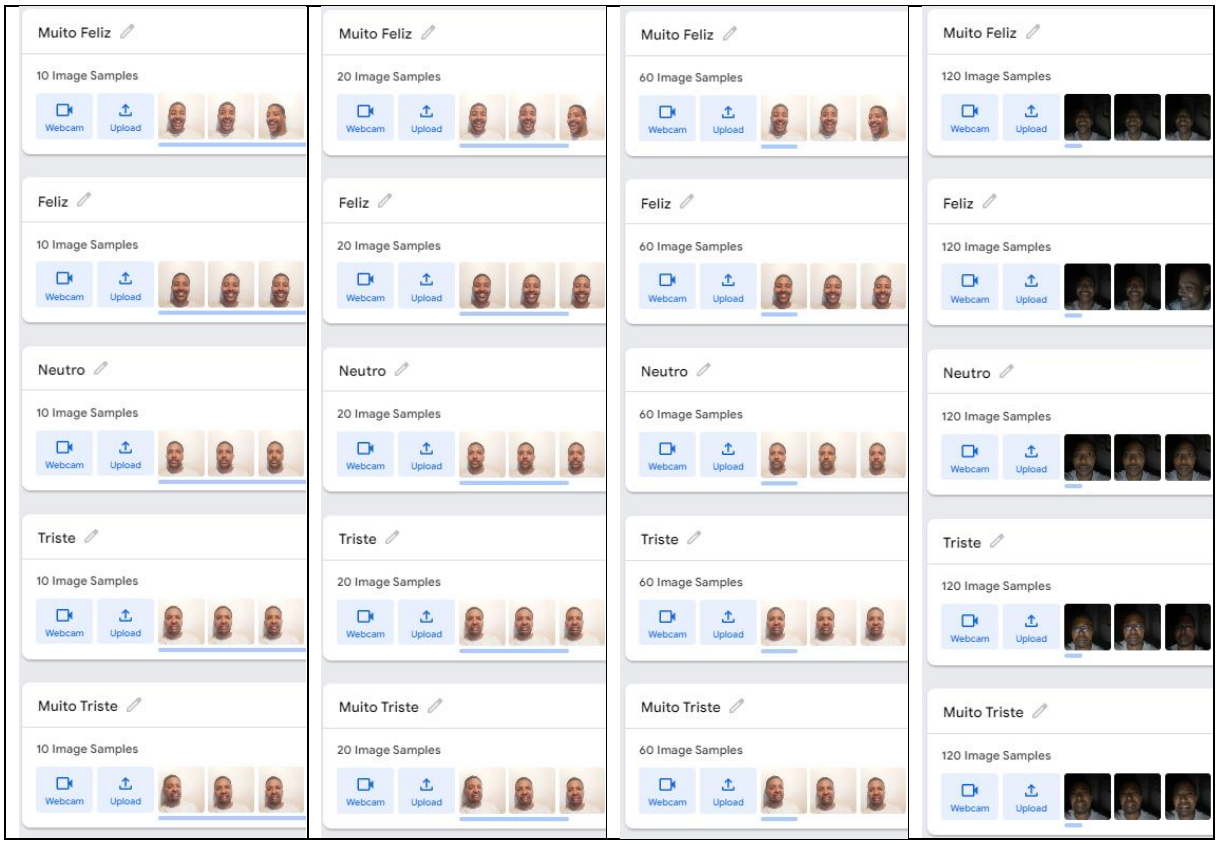
Fonte: Dados originais da pesquisa

Os treinamentos do modelo utilizaram o método "*Holdout*", que consiste em separar um conjunto de dados entre dados de Treino e dados de Validação. Neste

método, foram utilizadas amostras de imagens rotuladas e divididas entre 6 classes de imagens para Treino e em 6 classes para Validação, mantendo uma proporção de 85% - 15%.

Neste método, o conjunto de 15% de imagens de validação é diferente do conjunto de imagens de treinamento usado pelo algoritmo de aprendizado profundo, esta separação visa obter uma estimativa concreta de generalização do modelo, ou seja, visa identificar se o modelo aprendeu a diferenciar as imagens utilizadas no treinamento das imagens que de fato devem ser identificadas. Desta forma, 85% das amostras de cada uma das classes foram utilizadas para treinar a classificação correta das classes.

Figura 5 - Treinamentos do modelo na plataforma Teachable Machine



Fonte: Dados originais da pesquisa

Depois de realizar a captura das imagens utilizadas para treino e validação, foram ajustados alguns parâmetros para definir como o modelo deveria ser treinado.

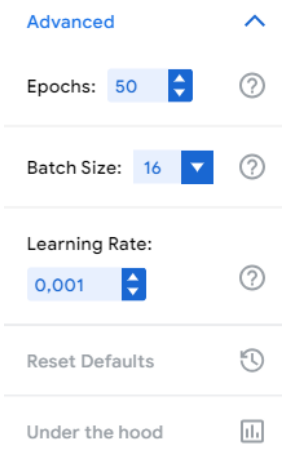
Nesta configuração foram definidos os hiperparâmetros de Iterações ou “*Epochs”*, tamanho do lote ou “*Batch Size*” e a Taxa de aprendizado ou “*Learning Rate*”.

O tamanho do lote “*Batch Size*” refere-se ao número de imagens usadas para treinamento por iteração. Portanto, se foram utilizadas 50 imagens para treinamento e um tamanho de lote de 16, isso significa que foram necessárias 3,125 iterações para compor uma época.

A ferramenta *Teachable Machine* utiliza como padrão o hiperparâmetro *“Batch Size”* de tamanho 16, pois um lote menor proporciona um treinamento que utiliza um processamento mais eficiente. Uma das vantagens em utilizar um tamanho de lote menor é uma maior eficiência no processamento, já que o modelo é atualizado após cada lote, assim a rede tende a ser treinada mais rapidamente.

Como alternativa a estes parâmetros pré-definidos, poderia ser realizado um método de varredura de hiperparâmetros para otimizar os valores utilizados e proporcionar um melhor ajuste para o modelo. No entanto, foi decidida a utilização destes valores com o objetivo de manter a performance de velocidade de treinamento padrão utilizada pela ferramenta.

Figura 6 Parâmetros utilizados para o treinamento



Fonte: Dados originais da pesquisa

A taxa de aprendizado *“Learning Rate*” refere-se ao quanto é necessário mudar o modelo com base no erro estimado, ou seja, é o parâmetro utilizado para calcular o

valor da correção do peso no momento do processo de treinamento. O valor da taxa de aprendizado varia entre 0 e 1. Quanto mais próximo de 1, mais rápido o processo de treinamento será executado, mas a precisão da rede neural deste modelo diminuirá. Por outro lado, quanto mais próximo de 0, maior a precisão da rede neural do modelo, consequentemente, o processo de treinamento levará mais tempo.

O tamanho do lote "*Batch Size*" refere-se ao número de imagens usadas para treinamento por iteração[9]. Ou seja, o tamanho do lote controla o número de amostras de imagens de treinamento a serem processadas antes que outros parâmetros do modelo sejam atualizados.

O parâmetro "*Epochs*" refere-se ao número de vezes que cada imagem é utilizada pela rede neural no treinamento passa por uma iteração completa do algoritmo. Quando o número de épocas é alto, uma mesma imagem é utilizada várias vezes, e neste caso, não seriam necessárias muitas imagens de amostra. O principal motivo é a adaptação excessiva que muitas épocas podem causar, fazendo com que a rede neural seja precisa em predições com o conjunto de imagens de treinamento, mas falhe em novos dados.

Este é o resultado do *tradeoff* viés-variação, onde um modelo com alto viés, tem um bom desempenho para o conjunto de dados de treinamento, mas não em um novo. Por outro lado, se houver alta variação, pode não funcionar tão bem nos dados de treinamento, mas pode ter mais flexibilidade quando se trata de novos dados. Portanto, determinar o ponto de equilíbrio entre variação e viés é um procedimento complexo, mas a plataforma *Teachable Machine* realiza este processo com seu algoritmo, cabendo ao desenvolvedor calibrar apenas os parâmetros supracitados.

4.1 MÉTRICAS DE AVALIAÇÃO DO MODELO

4.1.1 Acurácia

De acordo com (SKANSI, Sandro 2018 p.57), a acurácia é a porcentagem de classificações que um modelo acerta durante o treinamento, ou seja, é uma métrica

[9] Se forem utilizadas 80 imagens e um tamanho de lote de 16, isso significa que serão necessárias 5 iterações para compor uma *epoch*.

utilizada para avaliar a qualidade do modelo. Esta métrica pode ser definida como o número de previsões corretas de Verdadeiro-Positivos [VP] e Verdadeiro-Negativos [VN] sobre o número total de tentativas, sejam elas certas ou erradas[10].

$$Acurácia = \frac{VP+VN}{VP+VN+FP+FN} \quad (9)$$

A acurácia do modelo treinado foi calculada usando as amostras de validação que correspondem a 15% das amostras de imagens capturadas e que não foram utilizadas para treinar cada uma das classes. Portanto, após o modelo ter sido treinado com uma parcela de amostras, o restante destas amostras foi utilizado para verificar o desempenho do modelo em dados novos, nunca vistos antes.

A acurácia obtida após estes treinamentos indica quão boa foi a predição do modelo de Machine Learning na classificação das classes de expressões faciais de interesse. Este indicador é representado abaixo pelo número de verdadeiros-positivos somado ao número de verdadeiros negativos e dividido pelo número total de tentativas. Aplicando a fórmula para obter a acurácia total para cada um dos treinamentos, o resultado obtido é o seguinte:

Quadro 1 - Cálculos de acurácia

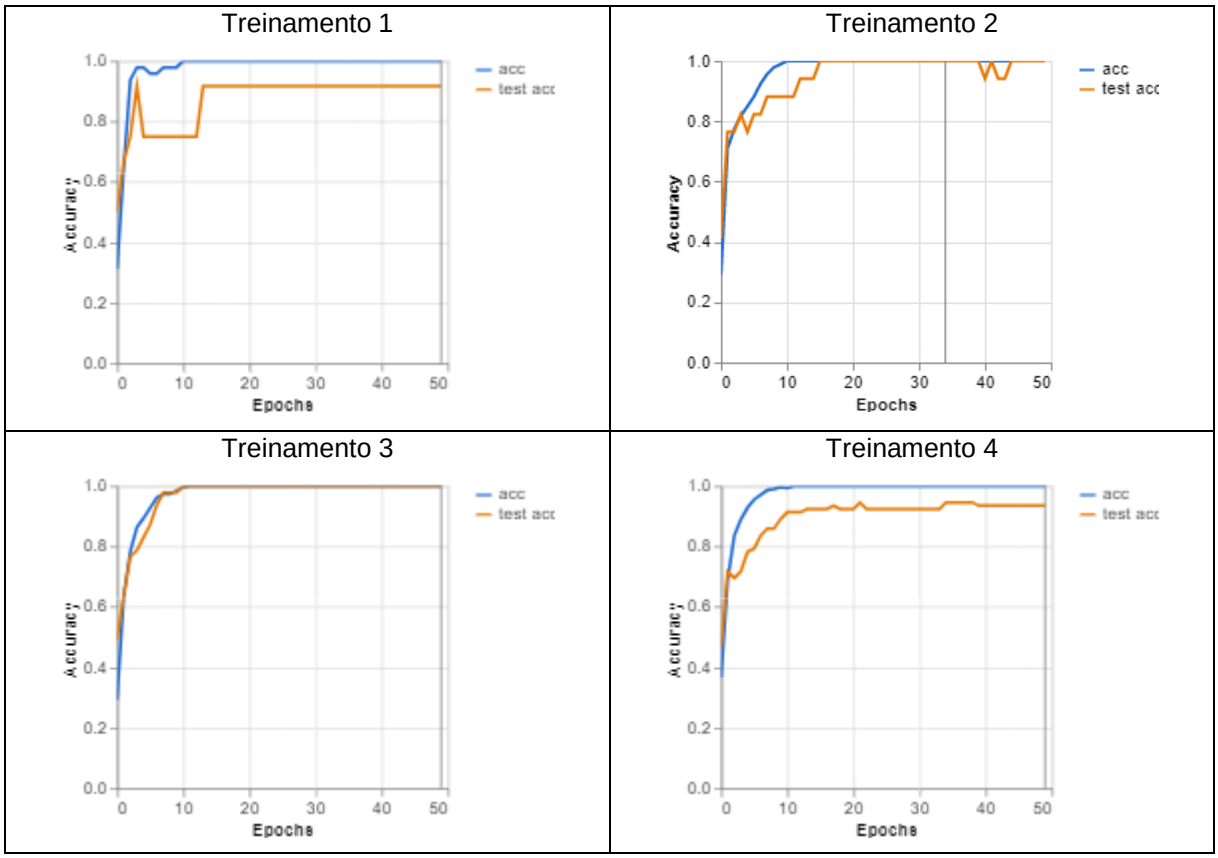
Treinamento 1	Treinamento 2
$\text{Acurácia} = \frac{11+1}{11+1+1+0}$ $\text{Acurácia} = 0{,}857143$	$\text{Acurácia} = \frac{16+1}{16+1+1+0}$ $\text{Acurácia} = 0{,}944444$
Treinamento 3	**Treinamento 4**
$\text{Acurácia} = \frac{47+0}{47+0+0+0}$ $\text{Acurácia} = 1$	$\text{Acurácia} = \frac{86+6}{86+6+6+0}$ $\text{Acurácia} = 0{,}938775$

Fonte: Dados originais da pesquisa

[10] Exemplo: Se o modelo classifica 70 amostras de 100, a acurácia é de 70/100 = 0,7 ou 70%.

O Gráfico 4 mostra a evolução da acurácia do modelo de acordo com a quantidade de iterações (Epochs) realizadas para as etapas de treinamento (linha azul) e de validação (linha laranja). Analisando os resultados, é possível verificar que o indicador de acurácia atingiu valores maiores no treinamento 3, atingindo rapidamente uma acurácia máxima de 1 aproximadamente na iteração de número 10, o que é um indicativo de *Overfitting*.

Gráfico 4 - Acurácia de treinamento (Azul) e validação (Laranja)



Fonte: Dados originais da pesquisa

4.1.2 Matriz de Confusão

A matriz de confusão é uma outra métrica utilizada para avaliar o desempenho do modelo. É possível usar essa matriz para descobrir com quais classes o modelo

possui mais acurácia e com qual pode ficar “confuso”. O eixo y (Classe) representa a classe de amostras, e o eixo x (Previsão) representa a classe que o modelo, após aprender consegue prever a que pertencem.

Nesta modelo, possíveis confusões podem ocorrer se uma classe de amostra for "Muito Feliz", mas sua previsão for "Feliz", isso significa que, após aprender com as imagens de treino, o modelo classificou incorretamente essa amostra de “Muito Feliz” como “Feliz”, tendo um resultado falso-positivo.

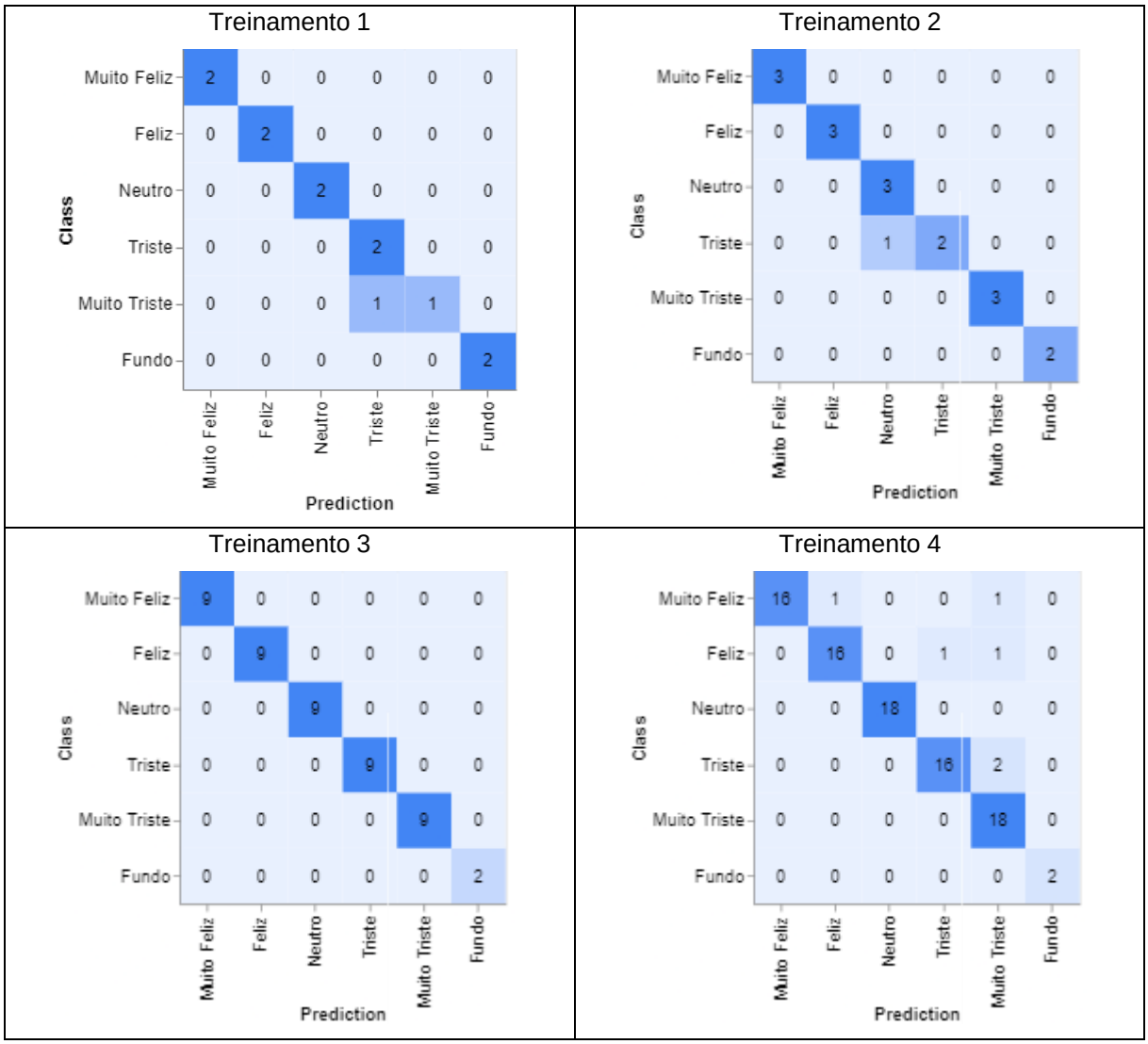
Isso geralmente significa que essas duas classes compartilham características em comum que o modelo captou, e que a amostra "Muito Feliz" específica era mais semelhante às amostras da classe "Feliz".

Os treinamentos 1, 2 que utilizaram respectivamente 50, 100 imagens apresentaram uma acurácia satisfatória ao detectar corretamente na predição da maioria das classes. Nestes casos as matrizes são representadas com apenas 1 resultado falso-positivo em cada treinamento, como por exemplo a detecção de uma imagem rotulada como “Muito Triste” sendo detectada como “Triste” no Treinamento 1, uma imagem rotulada como “Triste” sendo detectada como “Neutro” no Treinamento 2.

O treinamento 3, foi o que apresentou a melhor acurácia dentre todos os treinamentos, não apresentando resultados falso-positivos, e detectando corretamente todas as classes de imagens. Em contrapartida, o treinamento 4, apresentou mais resultados falso-positivos, onde imagens rotuladas como pertencentes a uma determinada classe, foram detectadas como pertencentes a outras classes de forma incorreta.

Neste sentido, o quarto e último treinamento foi o que utilizou a maior quantidade de imagens. No entanto, foi o que apresentou a pior acurácia dentre todos os treinamentos, o que evidencia que uma maior quantidade de imagens não significa uma maior acurácia no modelo de Aprendizado de Máquina.

Gráfico 5 - Matriz de confusão



Fonte: Dados originais da pesquisa

Sendo assim, a matriz de confusão e o gráfico de acurácia demonstram que após os treinamentos, o modelo que utilizou 300 imagens seria o modelo de Machine Learning com melhor acurácia, e deveria ser o modelo escolhido. No entanto, esta acurácia exageradamente alta, onde 100% das detecções foram verdadeiro-positivas, são um indicativo de *overfitting*.

4.1.3 Outros indicadores de qualidade: “Precision”, “Recall” e “F1-Score”

Antes de iniciar esta seção, e importante destacar que acurácia e precisão são conceitos distintos, a precisão de um modelo de aprendizado profundo é uma métrica que indica a capacidade do algoritmo em detectar o objeto de interesse levando em consideração apenas os resultados verdadeiro-positivos, diferentemente da acurácia, que além dos resultados verdadeiro-positivos, também leva em consideração os resultados verdadeiro-negativos, para expurgar resultados negativos do resultado.

Apesar de um modelo apresentar bons resultados de detecções verdadeiro-positivas, tais detecções nem sempre são corretas. Por isso, a métrica de Precisão não pode ser analisada de forma independente, mas sim em conjunto com outros indicadores, que também avaliam se o modelo foi capaz de realizar tais detecções com precisão e acurácia.

A avaliação do modelo de detecção de expressões faciais utilizando os indicadores de "*Precision*" ou precisão e "*Recall*" permite mensurar um outro indicador de desempenho do modelo com uma métrica de confiança denominada "*F1-Score*". Sendo assim, o indicador "*F1-Score*" é útil para determinar o nível de confiança de equilíbrio para os valores de precisão e "*Recall*". Estes indicadores variam entre valores de confiança de 0 a 1, onde uma métrica de valor 1 no "F1-Score" pode ser entendida no indicador como um bom indicador do desempenho do modelo.

Normalmente, à medida que o limite de confiança aumenta, a precisão também aumenta e o "*Recall*" diminui, este padrão será avaliado a seguir, com o objetivo de analisar o ponto de equilíbrio ideal para o nível de confiança do modelo.

Como o modelo resultante do terceiro treinamento possui 100% de resultados verdadeiro positivos, a precisão e o "Recall", que são o número de elementos classificados como Verdadeiro-Positivo dentre o número total de instâncias rotuladas como pertencentes à classe positiva, foram iguais a 1. Logo, o F1-score também será igual a 1.

Quadro 2 Calculos de Precisão, Recall e F1-Score

Treinamento 1	Treinamento 2
$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}} = \frac{11}{12} = 0{,}916667$	$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}} = \frac{16}{17} = 0{,}941176$

$\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}} = \frac{11}{11 + 1} = 0{,}916667$ $\text{F1} - \text{Score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}}$ $\text{F1} - \text{Score} = 2 \times \frac{0{,}840278389}{1{,}833334}$ $\text{F1} - \text{Score} = 0{,}916667$	$\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}} = \frac{16}{16 + 1} = 0{,}941176$ $\text{F1} - \text{Score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}}$ $\text{F1} - \text{Score} = 2 \times \frac{0{,}885812263}{1{,}882352}$ $\text{F1} - \text{Score} = 0{,}941176$
Treinamento 3 $\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}} = \frac{47}{47 + 0} = 1$ $\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}} = \frac{47}{47 + 0} = 1$ $\text{F1} - \text{Score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}}$ $\text{F1} - \text{Score} = 2 \times \frac{1}{2}$ $\text{F1} - \text{Score} = 1$	Treinamento 4 $\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}} = \frac{86}{92} = 0{,}916667$ $\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}} = \frac{86}{86 + 6} = 0{,}916667$ $\text{F1} - \text{Score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}}$ $\text{F1} - \text{Score} = 2 \times \frac{0{,}873817388}{1{,}869564}$ $\text{F1} - \text{Score} = 0{,}934782$

Fonte: Dados originais da pesquisa

4.1.4 Relação entre erros de treino e de validação

O objetivo de realizar os treinamentos do modelo de Aprendizado de Máquina com diferentes quantidades de imagens, é a redução do viés algorítmico que, segundo (OLESZAK, Michal; 2019), tem o custo de introduzir uma maior variância ao modelo preditivo, tornando o algoritmo igualmente ineficaz fazendo com que o modelo "decore" os dados, sem conseguir fazer a generalização em exemplos novos não vistos antes. Este outro problema é comumente chamado em Aprendizado de Máquina de *Overfitting*.

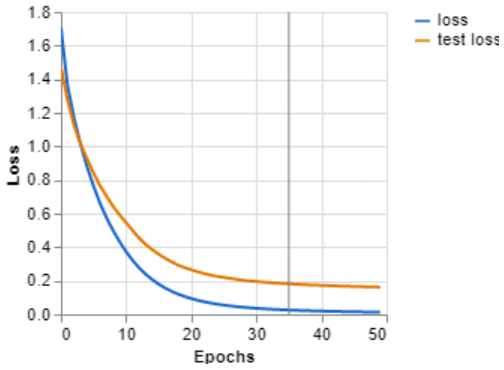
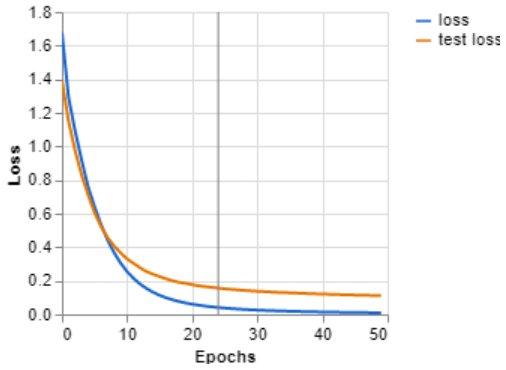
O gráfico 6 fornece os dados necessários para identificar o problema de *Overfitting* no modelo de detecção facial. O gráfico apresenta no eixo x a quantidade de iterações utilizadas no treinamento do modelo e no eixo y a proporção de erros que

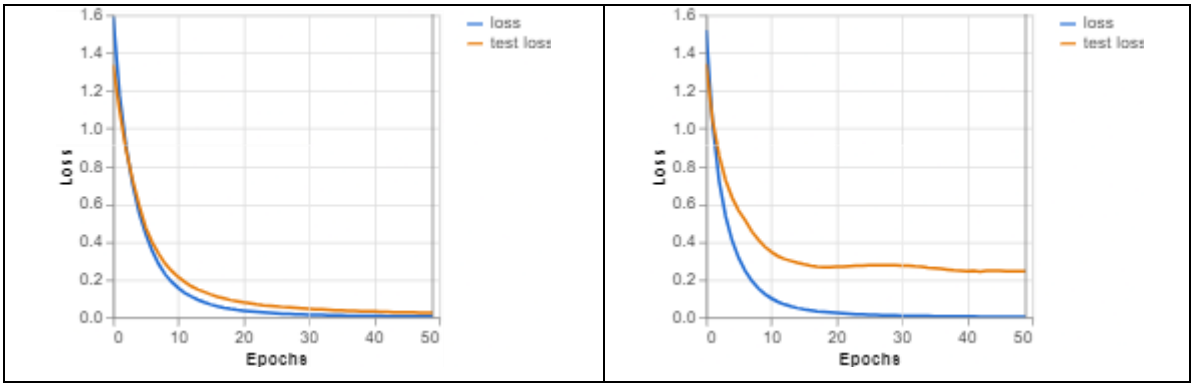
o modelo possui. As linhas azuis e laranjas, demonstram respectivamente a variação de erros conforme aumentam as iterações nos dados de Validação e nos dados de Treino. Nestes gráficos os erros de Validação e de Treino são representados respectivamente por “test loss” e “loss”.

Sendo assim, é possível analisar que conforme as iterações aumentam a quantidade de erros diminui nas últimas iterações, e que no terceiro treinamento, que passou a utilizar 300 imagens, o modelo passou a apresentar uma proporção de erros menor que nos outros treinamentos. Este resultado pode ser explicado em decorrência do aumento no número de imagens de treino e de validação.

Esta tendencia de queda na curva de erros de treino e de validação também ocorreu no quarto treinamento, no entanto este passou a apresentar um ponto de inflexão após o algoritmo passar de 10 iterações. Isso caracteriza que o modelo passou de um ponto ótimo, e passou a operar em um ponto de “*overfitting*”, já que os erros passaram a aumentar nas próximas iterações, onde existe baixo erro de treinamento “loss” e alta quantidade de erros de validação “test loss”.

Gráfico 6 - Perda de treinamento (Azul) e validação (Laranja)

<table>
<tr><td>Treinamento 1</td><td>Treinamento 2</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>Treinamento 3</td><td>Treinamento 4</td></tr>
</table>



Fonte: Dados originais da pesquisa

Este ponto de inflexão no quarto treinamento também pode ser interpretado como o ponto de "*Early Stopping*", já que os erros a partir desta quantidade de iterações passou a aumentar, definindo assim que este é a quantidade ótima de iterações para evitar o "*overfitting*". Portanto, um modelo que possui "*underfitting*" terá alto erro de treinamento e alto erro de validação, enquanto um modelo "overfitting" terá um baixo erro de treinamento "loss", mas um erro de validação alto "test loss", conforme demonstrado no gráfico 6.

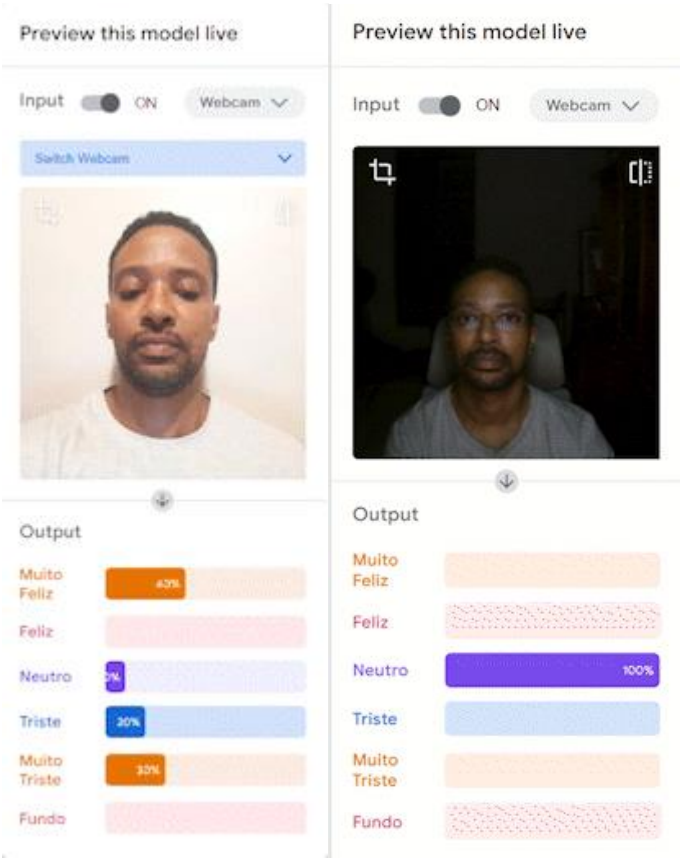
4.2 APLICAÇÃO DO MODELO

Depois de finalizado o treinamento, o modelo de detecção de expressões faciais é capaz de realizar testes/validações em novas amostras de imagens em uma aplicação final. Nesta aplicação, cada nova amostra de imagens capturadas pela câmera, é interpretada por uma rede neural artificial e alimenta o modelo de aprendizado de máquina com capacidade de realizar novas predições. (Feltrin, Fernando p. 49).

Foi utilizado o modelo de aprendizado de máquina resultante do treinamento 3, que obteve a maior acurácia. O sistema final de detecção de expressões faciais que utiliza este modelo, pode ser acessado através da url https://teachablemachine.withgoogle.com/models/HPATHeGnu/. Esta aplicação utilizou como base um modelo com boa capacidade de generalização, no entanto como foi gerada utilizando bases de imagens pessoais, só obtém uma acurácia

satisfatória na detecção de expressões do autor do presente trabalho, não conseguindo generalizar expressões faciais de outros rostos.

Figura 7 - Teste do modelo na plataforma Teachable Machine



Fonte: Dados originais da pesquisa

Como o modelo foi desenvolvido utilizando uma plataforma online que minimiza a possibilidade de erros na etapa de treinamento, era esperado que o modelo gerado minimizasse o viés algorítmico sem aumentar de forma prejudicial a variância na detecção das expressões faciais. Desta forma, o *trade-off* existente entre Viés e Variância algorítmica que os principais algoritmos de detecção de imagens possuem, não deveria prejudicar o desempenho das detecções no terceiro treinamento em um ambiente com iluminação.

Neste sentido, após realização do treinamento e da escolha do melhor modelo

de detecção de expressões faciais, foram realizados diversos testes em ambiente com diferentes níveis de iluminação, em que foram identificados resultados falso-positivos, em expressões faciais que não estavam de acordo com o esperado, evidenciando a incapacidade do modelo em realizar predições com uma acurácia satisfatória em imagens diferentes das utilizadas no treinamento, como em locais com baixa iluminação.

5 CONSIDERAÇÕES FINAIS

Ao treinar o modelo de Aprendizado de Máquina com imagens das diferentes expressões faciais foi observado que a acurácia aumentou gradativamente ao aumentar a quantidade de imagens, sobretudo no terceiro treinamento, que apresentou melhor acurácia.

Este aumento de imagens fez com que cada uma das classes fosse identificada com uma acurácia maior, reduzindo a quantidade de erros e de detecções falso-positivas. No entanto, ficou evidenciado no quarto treinamento que um aumento na quantidade de imagens de treino, não necessariamente aumenta a acurácia e reduz o viés e a variância do modelo de aprendizado de máquina.

Apesar da boa acurácia apresentada pelo terceiro treinamento, o sistema de detecção de expressões faciais pode ser vulnerável a detecções que sejam realizadas em ambientes com condições diferentes das utilizadas durante o treinamento. Este problema foi detectado durante o quarto treinamento, que utilizou 600 imagens, mas dentre estas imagens constavam algumas com pouca iluminação.

Portanto, a conclusão do trabalho é que, apesar de um aumento na amostra de imagens, o modelo ainda assim possui um viés que prejudica a acurácia de detecções, este viés está relacionado a uma amostra de imagens de treino que não representa todas as condições possíveis de ambientes onde as imagens podem ser captadas, como ambientes com baixa iluminação, expressões faciais que não foram utilizadas durante o treinamento ou alinhamento da face que dificulta a detecção.

Além disso, apesar do modelo resultante do treinamento 3 ter aprendido muito bem com o conjunto de imagens de treino, este modelo ficou superajustado apenas às imagens utilizadas neste treinamento, e apesar de ser altamente preciso, alcançando uma precisão máxima de 1 ele ficou caracterizado pelo problema de *Overfitting*. Neste treinamento, o modelo “aprendeu demais” sobre aspectos das expressões faciais em um ambiente iluminado e “aprendeu de menos” sobre aspectos das expressões faciais em ambientes pouco iluminados.

Tais vulnerabilidades podem ser corrigidas em trabalhos futuros com a

utilização de amostras de imagens coletadas em ambientes com variados tipos de iluminação. Além disso poderia ser feita uma análise comparativa que utilize os resultados de acurácia em modelos treinados com pessoas de diferentes tons de pele, para verificar possíveis problemas de viés racial.

Por fim, estas fragilidades do modelo de Aprendizado de Máquina demonstram que algoritmos de detecção facial devem ser utilizados com cautela, pois o treinamento exige análise minuciosa de cada um dos indicadores de acurácia, e mesmo aqueles que apresentam acurácia satisfatória podem apresentar viés ao detectar as classes de interesse utilizadas durante os treinamentos.

6 REFERÊNCIAS

[1]. AMAZON. **Amazon Machine Learning: Guia do desenvolvedor**. 2022. Disponível em: https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/machinelearning-dg.pdf#model-fit underfitting-vs-overfitting. Acesso em: 07 abr. 2023.

[2]. AVATI, Anand. **Bias-Variance Analysis: Theory and Practice**. Stanford University. California, 2020. Disponível em: http://cs229.stanford.edu/summer2020/BiasVarianceAnalysis.pdf. Acesso em: 07 abr. 2021.

[3]. BAGHERINEZHAD, Hessam; EHSANI, Kiana; FARHADI, Ali; MOTTAGHI, Roozbeh; REDMON, Joseph. **Who Let The Dogs Out? Modeling Dog Behavior From Visual Data**. University of Washington, Allen Institute for AI. Washington, 2018. Disponível em: https://pjreddie.com/media/files/papers/1803.10827.pdf. Acesso em: 07 abr. 2023.

[4]. FÁVERO, Luiz Paulo. **Manual de Análise de Dados. Estatística e Modelagem Multivariada com Excel, SPSS e Stata**. 1 ed. Rio de Janeiro. Elsevier: Campus, 2017.

[5]. FELTRIN, Fernando. **Deep Learning em Python**. Python Nexus. Edição do Kindle. 2023

[6]. GRUS, Joel. Data Science do Zero. **Primeiras Regras com Python**. 2 ed. Rio de Janeiro. Alta Books: O'Reilly, 2016. Disponível em: https://edisciplinas.usp.br/pluginfile.php/5742167/mod_resource/content/1/Data%20Science%20do%20zero%20-%20Primeiras%20regras.pdf. Acesso em: 07 abr. 2023.

[7]. HASTIE, T., TIBSHIRANI, R., Friedman, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction.** 2 Ed. California. Springer. 2013.

[8]. IANDOLA, Forrest N.; HAN, Song. **Squeezenet: Alexnet-Level Accuracy With 50x Fewer Parameters And <0.5mb Model Size.** Stanford University. 2017. Disponível em: https://arxiv.org/pdf/1602.07360.pdf

[9]. S LI, Stan Z.; JAIN, Anil K. **Handbook of Face Recognition**. 2 ed. Londres. Springer, 2011.

[10]. NG, Andrew. Machine Learning Yearning. **Technical Strategy for AI Engineers, In the Era of Deep Learning**. Deeplearning.ai. 2018.

[11]. OLESZAK, Michal. 2019. **Regularization: Ridge, Lasso and Elastic Net**. Disponível em: https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net. Acesso em: 07 abr. 2023.

[12]. SAES, Keylla Ramos. **Abordagem para Integração automática de dados estruturados e não estruturados em um contexto Big Data**. Dissertação (Mestrado em Sistemas de Informação) – Universidade de São Paulo, São Paulo, 2019. Disponível em: https://teses.usp.br/teses/disponiveis/100/100131/tde-16012019-212403/publico/Dissertacao_Keylla_Saes_Versao_Final_Pos_Banca_2019_01_15.pdf. Acesso em: 07 abr. 2023.

[13]. SALOMON, Gabriel. **Deep Learning Aplicado a Robótica**. UFPR. Disponível em: https://www.inf.ufpr.br/gsaniceto/rosie/rosie-deep.pdf. Acesso em: 07 abr. 2023.

[14]. SKANSI, Sandro. **Introduction to Deep Learning. From Logical Calculus to Artificial Intelligence**. 1 ed. Londres. Springer. 2018.

[15]. RUSSEL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 3 ed. rev. Rio de Janeiro. Elsevier: Campus, 2013.

5.1 APÊNDICE A – CÓDIGO EM JAVASCRIPT DO MODELO

```
<div>Teachable Machine Image Model</div>
<button type="button" onclick="init()">Start</button>
<div id="webcam-container"></div>
<div id="label-container"></div>
<script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/dist/teachablemach
ine-image.min.js"></script>
<script type="text/javascript">

    // link para o seu modelo fornecido pelo painel de exportação Teachable Machine
    const URL = "https://teachablemachine.withgoogle.com/models/HPATHeGnu/";

    let model, webcam, labelContainer, maxPredictions;

    // Carrega o modelo de imagem e configura a webcam
    async function init() {
        const modelURL = URL + "model.json";
        const metadataURL = URL + "metadata.json";

        // carregar o modelo e os metadados
        model = await tmImage.load(modelURL, metadataURL);
        maxPredictions = model.getTotalClasses();

        // Função de configuração da webcam
        const flip = true; // whether to flip the webcam
        webcam = new tmImage.Webcam(200, 200, flip); // width, height, flip
        await webcam.setup(); // request access to the webcam
        await webcam.play();
        window.requestAnimationFrame(loop);

        // acrescenta elementos ao conteiner
        document.getElementById("webcam-container").appendChild(webcam.canvas);
        labelContainer = document.getElementById("label-container");
        for (let i = 0; i < maxPredictions; i++) { // and class labels
            labelContainer.appendChild(document.createElement("div"));
        }
    }

    async function loop() {
        webcam.update(); // update the webcam frame
        await predict();
        window.requestAnimationFrame(loop);
    }

    // execute a webcam através do modelo
    async function predict() {
        // Predição pode incluir uma imagem, vídeo ou elemento html de tela
        const prediction = await model.predict(webcam.canvas);
        for (let i = 0; i < maxPredictions; i++) {
            const classPrediction =
                prediction[i].className + ": " +
prediction[i].probability.toFixed(2);
            labelContainer.childNodes[i].innerHTML = classPrediction;
        }
    }
</script>
```