

Universidade Federal do ABC  
Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas  
Trabalho de Graduação em Engenharia de Informação

# **Desenvolvimento e Análise de Viabilidade de Algoritmos de Detecção Facial**

**Gabriel Pitali de Carvalho**

**Santo André**

**Maio de 2020**



Gabriel Pitali de Carvalho

## **Desenvolvimento e Análise de Viabilidade de Algoritmos de Detecção Facial**

**Trabalho de Graduação** apresentado ao concluir a Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Universidade Federal do ABC

Orientador: André Kazuo Takahata

Santo André  
Maio de 2020

# Resumo

Devido aos grandes avanços na área de visão computacional, muitas ferramentas relacionadas ao assunto têm sido criadas, mas torna-se complexo escolher qual a ideal para a necessidade existente e como podem ser aplicadas. Para entender melhor as opções disponíveis e suas aplicações, este trabalho apresenta um estudo sobre o desempenho de dois diferentes algoritmos de detecção facial em imagens e a análise da viabilidade da aplicação dos mesmos em um projeto comercial. Foi estudado o método de Viola-Jones de análise de características e uma *rede convolucional em cascata multi tarefa* (MTCNN, *Multi-Task Cascaded Convolutional Neural Network*) e o desempenho foi medido utilizando conjunto de imagens selecionado manualmente de acordo com uma lista de regras estabelecidas. A análise dos resultados foi feita em função da acurácia obtida e do lucro gerado em uma situação hipotética de uma empresa que necessita validar fotos cadastrais. Ambos os métodos se mostraram lucrativos, sendo que o método *MTCNN* teve o melhor resultado entre os testes. Após a análise, foi possível determinar que é comercialmente interessante a implementação de modelos de visão computacional mas é necessário dedicar esforços para adequação das ferramentas aos objetivos específicos de cada caso.

**Palavras-chaves:** Aprendizado de Máquina. Detecção Facial. Visão Computacional.

# Abstract

Due to the great advances in the field of computer vision, many tools related to this subject have been created, but it has become complex to choose which one is ideal for the existing needs and how they can be applied. In order to understand better the options available and their applications, this paper presents a study on the performance of two different face detection algorithms and the analysis of how feasible is to apply them in a commercial project. The Viola-Jones method of characteristic analysis and a Multi-Task Cascaded Convolutional Neural Network (*MTCNN*) were studied and their performances were measured using a set of manually selected images according to a list of established rules. The analysis of the results was made according to the accuracy obtained and the profit generated in a hypothetical situation of a company that needs to validate registration photos. Both methods proved to be profitable, with the *MTCNN* method having the best result among the tests. After the analysis, it was possible to determine that it is commercially interesting to implement computer vision models, but it is necessary to dedicate efforts to adapt the tools to the specific objectives of each case.

**Keywords:** Machine Learning. Face Detection. Computer Vision.



# Listas de ilustrações

Figura 1 – Representação da imagem integral. . . . .	4
Figura 2 – Exemplos de características retangulares analisadas. . . . .	5
Figura 3 – Características retangulares mais eficientes para detecção facial. . . . .	6
Figura 4 – Diagrama do funcionamento do classificador em cascata. . . . .	6
Figura 5 – Resultados da detecção com variação de parâmetros. . . . .	8
Figura 6 – Resultados de cada etapa do método MTCNN. . . . .	10
Figura 7 – Exemplos de imagens do dataset <i>UTKFace</i> . . . . .	11
Figura 8 – Exemplos de imagens com face válida. . . . .	12
Figura 9 – Exemplos de imagens com face inválida. . . . .	13
Figura 10 – Exemplos de imagens sem nenhuma face. . . . .	13
Figura 11 – Exemplo teórico de uma distribuição normal dos grupos de uma matriz de confusão. . . . .	14
Figura 12 – Diagrama de Venn para as imagens analisadas. . . . .	15
Figura 13 – Espaço ROC. . . . .	16
Figura 14 – Limiar lucrativo exibido no espaço ROC. . . . .	18
Figura 15 – Resultados das classificações sobre o espaço ROC. . . . .	19



# **Lista de tabelas**

Tabela 1 – Grupos observados. . . . .	12
Tabela 2 – Matriz de confusão. . . . .	14
Tabela 3 – Matriz de confusão com probabilidades marginais. . . . .	15
Tabela 4 – Resultados obtidos com os classificadores. . . . .	20
Tabela 5 – Matriz de confusão com os resultados do classificador mais lucrativo. .	21



# Sumário

<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	<b>1</b>
<b>1.1</b>	<b>Objetivos e Motivação . . . . .</b>	<b>1</b>
<b>2</b>	<b>MATERIAIS E MÉTODOS . . . . .</b>	<b>3</b>
<b>2.1</b>	<b>OpenCV e Viola-Jones . . . . .</b>	<b>3</b>
2.1.1	Imagen Integral . . . . .	4
2.1.2	Algoritmo de AdaBoost e Classificador em Cascata . . . . .	5
2.1.3	Parametrização do Algoritmo . . . . .	7
<b>2.2</b>	<b>Aplicações de Redes Neurais Convolucionais . . . . .</b>	<b>7</b>
2.2.1	TensorFlow e Keras . . . . .	8
2.2.2	MTCNN . . . . .	9
<b>2.3</b>	<b>Conjuntos de imagens para teste . . . . .</b>	<b>9</b>
2.3.1	Conjuntos de imagens originais . . . . .	9
2.3.2	Conjunto de imagens selecionadas . . . . .	11
<b>2.4</b>	<b>Metodologia de Análise . . . . .</b>	<b>12</b>
<b>2.5</b>	<b>Análise de Custo . . . . .</b>	<b>16</b>
<b>3</b>	<b>RESULTADOS E DISCUSSÃO . . . . .</b>	<b>19</b>
<b>4</b>	<b>CONCLUSÕES E TRABALHOS FUTUROS . . . . .</b>	<b>23</b>
<b>4.1</b>	<b>Conclusões . . . . .</b>	<b>23</b>
<b>4.2</b>	<b>Trabalhos Futuros . . . . .</b>	<b>23</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>25</b>



# 1 Introdução

Reconhecimento facial é uma tarefa trivial para humanos e há décadas tem sido um desafio para visão computacional e aprendizado de máquina, desde os anos 90 o tema emerge em diferentes conferências e com o aumento do poder computacional dos dias atuais, sua capacidade se expande muito, fazendo com que tal assunto receba enorme atenção, principalmente devido ao seu grande valor comercial e as mais diversas aplicações possíveis, como verificação de identidade, controle de acesso, segurança, investigação de imagens em bancos de dados, vigilância, entretenimento ou realidade virtual [1] [2].

O processo de reconhecimento facial de forma automatizada é separado em quatro principais etapas, conforme detalhado no livro de Li e Jain, primeiramente deve ser feita a *detecção facial*, que consiste em validar e localizar a existência de alguma face na imagem ou vídeo, a segunda etapa consiste no *alinhamento facial*, para que todas faces da base de dados sigam o mesmo padrão, a terceira etapa é a *extração de características* que permite a obtenção de informação efetiva que será útil na distinção das diferentes faces, a quarta e última etapa consiste na *correspondência de características*, onde as características extraídas anteriormente são comparadas com outras já conhecidas para que sejam identificadas.

Aprofundando o estudo da primeira etapa, de *detecção facial*, Hjelmås e Low indica duas diferentes metodologias, a primeira baseada em características e a segunda baseada em imagens, ambas posteriormente podem ser separadas em diversas técnicas mais específicas, como por exemplo a análise de características por constelação ou a análise de imagens com redes neurais, onde cada técnica específica possui seus prós e contras em relação às demais.

## 1.1 Objetivos e Motivação

Este trabalho tem como objetivo encontrar uma forma eficiente de atuar sobre a primeira etapa (*detecção facial*) do processo de reconhecimento facial, avaliando o desempenho qualitativo e quantitativo de diferentes metodologias e ferramentas disponíveis e permitindo a rápida identificação de imagens que não possuem uma face, para satisfazer a necessidade descrita a seguir.

Atualmente empresas e órgãos públicos possuem a necessidade de manter cadastros pessoais mas existe grande demanda para que estes cadastros sejam feitos de forma totalmente virtual pela população, pois isso evita o deslocamento de pessoas até os pontos de cadastro e torna todo o processo muito mais ágil. Certos cadastros incluem fotos de

identificação e isto traz a necessidade de uma verificação feita por humanos para validar se a mesma consiste em uma foto de face frontal, conforme é necessário para o cadastro.

A validação citada já ocorre nas empresas e órgãos públicos que precisam coletar documentos de forma virtual e é frequentemente feita de forma totalmente manual, onde funcionários tem que verificar cada uma das imagens recebidas e muitas vezes se deparam com fotos sem nenhuma face frontal ou sem condições de serem identificadas (desfocadas, por exemplo), que são rejeitadas para que uma nova imagem seja solicitada. Estas imagens claramente inválidas por estarem em desacordo com o padrão esperado (foto de face frontal), poderiam ser eliminadas por uma filtragem anterior, reduzindo grande parte do trabalho que é feito hoje manualmente.

Portanto este trabalho demonstra como os métodos existentes atualmente para o processo de detecção facial poderiam ser aplicados nos processos expostos, adicionando uma etapa extra antes da validação manual e trazendo redução de custos para as empresas que necessitam de tal validação. Além de testar os métodos existentes, também é proposto um modelo matemático que avalia a viabilidade e lucratividade de cada método a partir dos resultados dos testes.

## 2 Materiais e Métodos

Para obter melhor entendimento sobre ferramentas de visão computacional, em específico para detecção facial, este capítulo descreve a execução de testes feitos utilizando as ferramentas OpenCV e Keras, ambas aplicadas utilizando a linguagem de programação Python. As ferramentas foram utilizadas para análise de um conjunto específico de imagens e os resultados foram validados utilizando as métricas já conhecidas para aprendizado de máquina e também uma metodologia de análise de custo baseada em métricas do mercado.

### 2.1 OpenCV e Viola-Jones

A ferramenta OpenCV, que pode ser encontrada no website Github [5], é uma biblioteca de código aberto focada na solução de problemas utilizando visão computacional em tempo real, desenvolvida pela Intel e posteriormente pela Itseez, com suporte a múltiplas plataformas e uso gratuito sobre a licença de código aberto BSD. A ferramenta apresenta suporte a frameworks de aprendizado profundo, como TensorFlow, Pytorch e Caffe e contempla tanto funções básicas, para aplicações como processamento de imagem, alteração de cor ou resolução, até aplicações avançadas, como detecção facial, identificação de características e biometria [6].

Uma das ferramentas que será utilizada neste trabalho é a função de detecção de faces da ferramenta OpenCV, que utiliza um classificador em cascata baseado características. Esse é um método eficiente para reconhecimento de faces em imagens proposto por Paul Viola and Michael Jones, amplamente conhecido como método Viola-Jones, onde uma função é treinada com muitos exemplos positivos (imagens que contém o objeto a ser detectado) e negativos (imagens que não contém o objeto a ser detectado) e então utilizada para detectar as mesmas características em outras imagens [7].

A detecção de faces utilizando OpenCV consiste em duas etapas principais, a primeira consiste no treinamento do modelo, onde são apresentadas diversas imagens já identificadas para que o modelo encontre padrões positivos e negativos. Após o treinamento, a segunda etapa consiste em utilizar o modelo obtido para identificar, em novas imagens, características semelhantes as vistas nas imagens do treinamento. Neste projeto, será utilizado um modelo fornecido em conjunto com a ferramenta OpenCV, já treinado com diversos exemplos de faces frontais.

O algoritmo Viola-Jones foi publicado em 2001, no paper "Rapid object detection using a boosted cascade of simple features" [8] e é famoso por sua capacidade de detecção de faces com muita velocidade, isso ocorre devido a 3 principais técnicas utilizadas: o

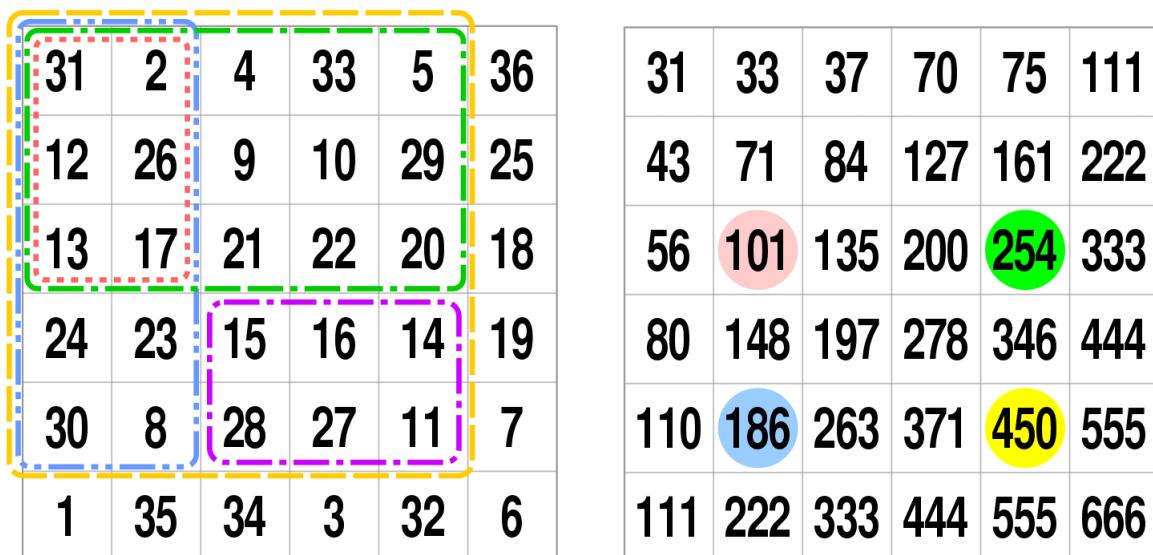
cálculo da imagem integral, o algoritmo *AdaBoost* e o classificador em cascata.

### 2.1.1 Imagem Integral

A primeira etapa do algoritmo Viola-Jones consiste em transformar a imagem original em uma imagem integral, isto é feito calculando o valor de cada ponto como a soma de todos os pontos que estão acima ou a esquerda do mesmo, como ilustrado na Figura 1.

Figura 1 – Representação da imagem integral.

Imagen original (esquerda) e imagen integral (direita). A correspondência das cores demarcadas na imagen com as variáveis da Equação 2.1 se da por A - amarelo, B - verde, C - azul e D - rosa.



Fonte: Wikipedia (2020)

A utilização desta técnica permite calcular facilmente a soma dos valores internos de qualquer retângulo formado entre quatro pontos da imagem, utilizando apenas o valor dos seus cantos, possibilitando assim a análise rápida de diversas partes da imagem.

O cálculo é feito definindo o retângulo a ser analisado e então aplicando a Equação 2.1. Tal equação é utilizada em 2.2 para calcular soma dos valores do retângulo da Figura 1 que inclui os pontos 15, 16, 14, 28, 27 e 11.

$$\text{Soma dos valores do retângulo} = D - (B + C) + A \quad (2.1)$$

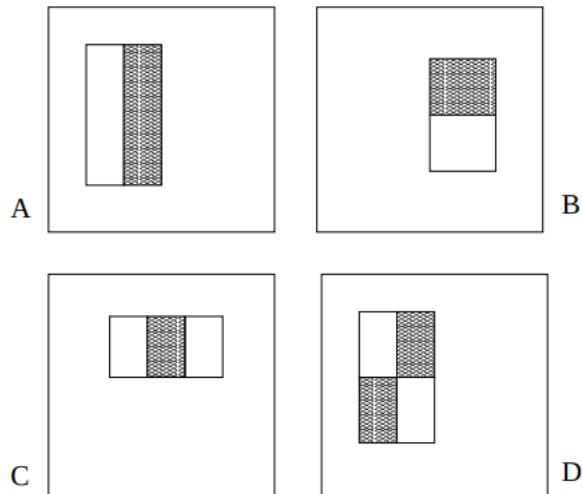
$$15 + 16 + 14 + 28 + 27 + 11 = 101 - (254 + 186) + 450 = 111 \quad (2.2)$$

Com a possibilidade de calcular facilmente a soma dos pixels de um retângulo arbitrário de forma rápida, o algoritmo para detecção pode analisar diversos trechos da

imagem, chamados aqui de características, fazendo a comparação de duas ou mais áreas retangulares predefinidas, como os exemplos ilustrado na Figura 2. O valor final de cada característica é definido pela soma do valor dos pixels sob o retângulo cinza menos a soma do valor dos pixels sob o retângulo branco.

Figura 2 – Exemplos de características retangulares analisadas.

As áreas demarcadas em cinza e branco compõe uma característica.



Fonte: Viola e Jones (2001)

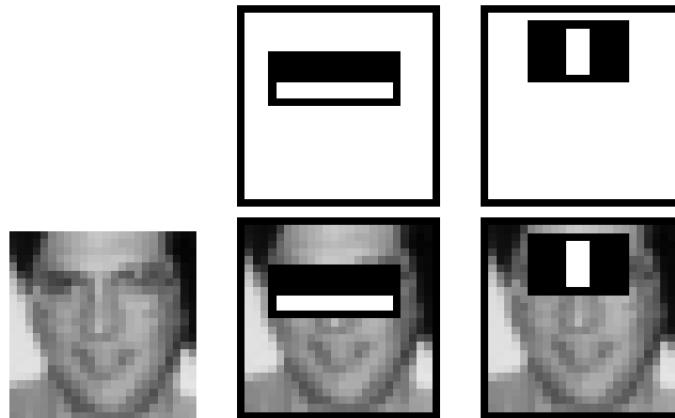
### 2.1.2 Algoritmo de AdaBoost e Classificador em Cascata

As características demonstradas anteriormente, são definidas basicamente como duas ou mais áreas retangulares de qualquer tamanho, tal simplicidade implica na possibilidade da criação de uma enorme variação das mesmas que precisam ser calculadas diversas vezes, para cada parte de imagem e com diferentes tamanhos, isso implica em um alto custo de processamento, para evitar tal problema é utilizado o algoritmo de *AdaBoost* [9], que possibilita grande otimização na demanda de processamento do detector facial e consequente redução do tempo de execução.

No caso do algoritmo de detecção facial Viola-Jones, o algoritmo AdaBoost é utilizado em dois momentos principais, o primeiro deles ocorre na seleção do conjunto de características mais eficientes durante o treino para criação do arranjo ponderado de vários classificadores fracos que combinados adequadamente se tornam um classificador forte [10]. A Figura 3 retrata as melhores características registradas por Viola e Jones (2001), fica claro que as mesmas se destacam por evidenciar as regiões dos olhos e do nariz.

É importante salientar que geralmente uma face não ocupa a maior parte de uma imagem a ser identificada, portanto é necessário encontrar uma forma rápida de descartar os elementos do fundo da mesma e concentrar o poder de processamento nos elementos que

Figura 3 – Características retangulares mais eficientes para detecção facial.

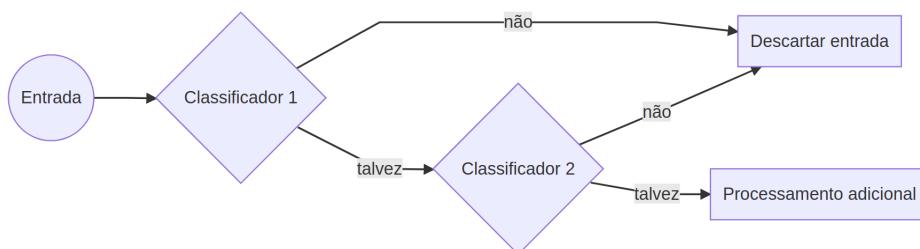


Fonte: Viola e Jones (2001)

têm maior probabilidade de serem reconhecidos como uma face, isso leva a uma formulação para o problema onde ao contrário de encontrar faces, é necessário um algoritmo que descarte as "não faces".

Neste segundo momento, o AdaBoost apresenta uma boa solução, esta consiste na utilização do arranjo de classificadores já ponderados anteriormente, aplicados de forma sequencial, iniciando do mais fraco para o mais forte, criando o que é chamado de classificador em cascata, conforme ilustrado na Figura 4, permitindo que imagens que certamente não possuem faces sejam rapidamente descartadas logo nas primeiras iterações, enquanto imagens com possíveis faces são classificadas por toda cascata, trazendo um elevado nível de confiança ao resultado.

Figura 4 – Diagrama do funcionamento do classificador em cascata.



Fonte: Própria (2020)

Um classificador comum, com um único estágio normalmente aceitaria muitos casos de falso negativo, para reduzir a taxa de falsos positivos e de descarte de imagens relevantes, mas no classificador em cascata, falsos positivos nos primeiros estágios não são um problema, pois serão analisados em outros diversos estágios e provavelmente eliminados.

A utilização desse modelo combinada com o algoritmo *AdaBoost*, possibilita a análise das características mais eficientes logo no início e consequentemente o descarte

muito mais rápido dos casos negativos nos primeiros estágios.

### 2.1.3 Parametrização do Algoritmo

O procedimento de detecção facial utilizando o OpenCV permite ainda ajuste de parâmetros para a execução do algoritmo de Viola-Jones.

O primeiro deles é chamado fator de escala que é o fator pelo qual as dimensões da imagem serão multiplicadas na tentativa de encontrar faces de diferentes tamanhos, pois o modelo é preparado para reconhecer faces de um tamanho específico, então durante a detecção a imagem é redimensionada e reanalisa diversas vezes para que faces de diferentes tamanhos possam ser detectadas. Para ajuste deste parâmetro, deve-se considerar que quanto mais próximo de 1 o seu valor, maior a chance de encontrar faces mas a execução do detector também se torna mais custosa em termos de processamento.

O segundo parâmetro é o número mínimo de vizinhos, que especifica quantos vizinhos cada retângulo candidato deve ter para retê-lo. Em mais detalhes, o detector analisa imagens utilizando um método de janelas retangulares que selecionam a partes da imagem, cada parte tem seus padrões comparados com os padrões de uma face e então é definido se esta janela contém ou não uma face, mas devido as várias iterações do detector sobre a mesma imagem durante a análise, uma mesma face pode ser detectada em mais de uma janela, tais janelas são consideradas janelas vizinhas, pois foram duas detecções que ocorreram em diferentes varreduras mas estão sobre uma mesma área da imagem. Portanto para melhorar a precisão do detector e também filtrar detecções duplicadas, o parâmetro de número mínimo de vizinhos define quantas janelas vizinhas devem existir para considerar aquela parte da imagem como uma face. Enfim, para ajuste deste parâmetro, deve-se considerar que quanto mais próximo de zero o seu valor, maior a chance de detectar faces, mas consequentemente maior a chance de falsos positivos.

Por exemplo, caso o parâmetro de número mínimo de vizinhos seja igual a zero, qualquer janela que detectar uma face, em qualquer uma das varreduras, será considerada como uma face no resultado final. Já com o parâmetro igual a 4, é necessário que a mesma região da imagem seja considerada face por 4 janelas de varreduras diferentes para ser considerada como uma única face no resultado final. Esse comportamento pode ser observado nos resultados apresentados na Figura 5.

## 2.2 Aplicações de Redes Neurais Convolucionais

Uma das ferramentas que tem ganhado maior destaque na área de visão computacional recentemente são as redes neurais profundas, principalmente os algoritmos de redes neurais convolucionais, devido a sua poderosa capacidade de detectar padrões mais complexos. [11] Uma rede neural convolucional funciona aplicando filtros com variados

Figura 5 – Resultados da detecção com variação de parâmetros.

Utilizando o parâmetro de número mínimo de vizinhos igual a 0 (esquerda), 2 (centro) e 4 (direita).



Fonte: Própria (2020)

pesos as imagens de entrada, possibilitando assim a detecção de padrões específicos. A principal diferença para a abordagem do método de Viola-Jones é que aqui os filtros e seus pesos são inicialmente aleatórios e então automaticamente aprimorados durante o treinamento da rede.

Os filtros aplicados através da operação de convolução, permitem detectar desde padrões simples, como linhas e curvas, mas com o treinamento e a combinação de uma grande quantidade de filtros, é possível detectar características de maior complexidade, como objetos, animais ou faces. Além disso, a operação de convolução ainda pode ser aplicada de forma que o tamanho da imagem resultante seja menor que o tamanho da imagem de entrada, mas mantendo as características mais importantes, isso faz com que a necessidade de poder computacional para o restante da análise seja reduzida [12].

Além da etapa de convolução com filtros, outras duas etapas igualmente importante são aplicadas nas imagens na rede neural. A primeira delas é a aplicação de uma função de ativação, que costuma ser não linear como a ReLU, que transforma todos valores resultantes negativos em zero sem alterar os valores positivos [13]. A segunda etapa aplicada sobre a imagem é a de agrupamento, que tem a função de reduzir o tamanho a imagem, com o objetivo de agilizar o treinamento e obter invariância ao deslocamento das features, tal redução é feita agrupando pixels próximos utilizando funções como o valor máximo ou médio entre os pixel da entrada [14].

### 2.2.1 TensorFlow e Keras

Para este trabalho foram utilizadas três ferramentas que facilitam a aplicação da rede neural convolucional. A primeira delas é chamada TensorFlow, uma plataforma de código aberto que disponibiliza ferramentas de aprendizado de máquina. A segunda

ferramenta, Keras, disponibiliza uma interface em python para utilização de plataformas como TensorFlow, facilitando bastante a utilização da mesma para o usuário final.

Adentrando a área de detecção facial especificamente, um método popular é o de *Rede Neural Convolucional Multi Tarefa em Cascata* (MTCNN, *Multi-Task Cascaded Convolutional Neural Network*), pois consegue atingir resultados bons não apenas para detecção de faces, mas também de partes da face, como olhos e boca. [15]

### 2.2.2 MTCNN

No método MTCNN, a estrutura em cascata consiste em três redes, no pré-processamento a imagem é redimensionada para uma variedade de tamanhos diferentes (chamada de pirâmide de imagens), depois a primeira rede (Rede de propostas) propõe regiões candidatas a conterem faces, então a segunda rede (Rede de aperfeiçoamento) filtra as regiões específicas que delimitam as posições das faces detectadas, por fim, a terceira rede (Rede de saída) propõe pontos de referência faciais como olhos e boca. Este método se assemelha bastante ao classificador em cascata do método de Viola-Jones, onde são utilizadas ferramentas mais simples para um filtragem inicial e então a complexidade das análises aumenta nas regiões onde possivelmente pode ser encontrada uma face [15]. Os resultados de cada uma das etapas podem ser visualizados na Figura 6.

Neste trabalho, foi utilizada uma implementação do método MTCNN que utiliza Python, TensorFlow e Keras e está disponível no github [16], esta implementação possui um modelo já treinado que a partir de uma imagem de entrada, fornece a posição das faces detectadas, assim como a confiança de que a mesma é realmente uma face e a posição dos pontos de referência (olhos, nariz e boca) da mesma. Para análise, todas as faces detectadas foram consideradas, independente da confiança na detecção, pois assim os resultados melhor satisfazem a necessidade deste projeto.

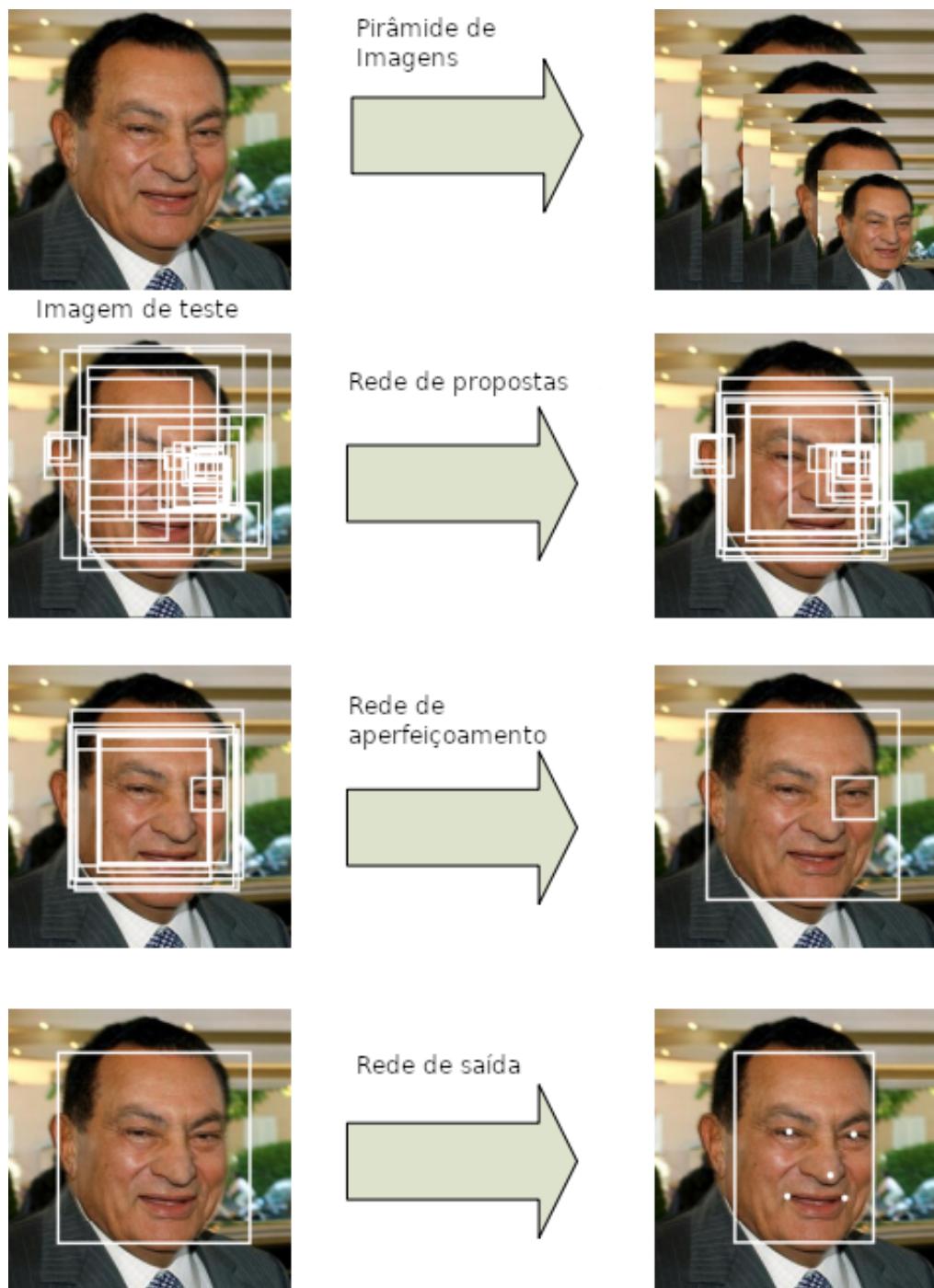
## 2.3 Conjuntos de imagens para teste

Para analisar a eficiência da implementação do algoritmo *Viola-Jones* na biblioteca *OpenCV*, assim como o seu modelo previamente treinado, foram selecionadas manualmente 500 imagens retiradas de dois conjuntos de imagens disponíveis na internet.

### 2.3.1 Conjuntos de imagens originais

O primeiro conjunto de imagens, nomeado *UTKFace*, disponível em [17], consiste em um conjunto de mais de 20 mil imagens, com uma única face em cada, de diversas pessoas entre 0 e 116 anos de idade, catalogadas de acordo com idade, raça e sexo. Alguns exemplos das imagens contidas no dataset podem ser vistos na Figura 7.

Figura 6 – Resultados de cada etapa do método MTCNN.



Fonte: Zhang et al. 2016

Figura 7 – Exemplos de imagens do dataset *UTKFace*.



Fonte: *UTKFace dataset* [17]

O segundo conjunto de imagens, nomeado *Hotels-50k* [18], consistia originalmente em mais de 50 mil fotografias de diversos quartos de hotel vazios, este foi utilizado pois imagens com características semelhantes a estas são muitas vezes submetidas erroneamente em cadastros pessoais, ao invés de uma imagem da face a ser cadastrada.

### 2.3.2 Conjunto de imagens selecionadas

Para os testes, foi selecionado manualmente um conjunto com 400 positivas (possuíam uma face válida) e 100 imagens negativas (não possuíam uma face válida). Tal proporção representa a hipótese de que cerca de 20% de imagens enviadas para cadastros na verdade não possuem uma face.

Por considerar que as imagens avaliadas pelo detector facial estudado neste trabalho serão utilizadas para cadastros pessoais e portanto é necessário que demonstrem que houve o consentimento de quem estava sendo fotografado e que seja possível identificar com clareza a pessoa da foto, as 400 imagens positivas (exemplos na Figura 8) foram selecionadas de forma que cumprissem os critérios abaixo.

- A imagem deve conter uma única face frontal;
- Olhos, nariz e boca devem estar visíveis;
- A pessoa fotografada deve ter entre 18 e 60 anos;
- A pessoa fotografada deve estar olhando para a câmera;
- A imagem deve ter boa definição e foco;
- A face deve estar centralizada e próxima da câmera.

Figura 8 – Exemplos de imagens com face válida.



Fonte: *UTKFace dataset* [17]

É preciso considerar também que existem imagens que contém faces mas que não são válidas pois não cumprem os critérios definidos, portanto essas devem ser consideradas negativas. Para maior realidade nos testes feitos, o grupo de imagens negativas foi composto por 30 imagens que apresentavam faces inválidas (exemplos na Figura 9) e outras 70 imagens selecionadas aleatoriamente do conjunto *Hotels-50k* (exemplos na Figura 10).

## 2.4 Metodologia de Análise

Para melhor analisar os resultados dos testes, foi necessário especificar com clareza como poderiam ser agrupadas as imagens, dada a sua origem e o resultado observado no teste, para isso foram utilizadas as definições da Tabela 1.

Tabela 1 – Grupos observados.

Grupo	Descrição	Quantidade
$A$	Imagens que contêm uma face válida	400
$\bar{A}$	Imagens que não contêm uma face válida	100
$B$	Imagens onde o algoritmo identificou uma ou mais faces	Variável
$\bar{B}$	Imagens onde o algoritmo não identificou nenhuma face	Variável

Definidos os grupos, pode-se utilizar a matriz de confusão da Tabela 2 para facilitar a análise da relação entre os grupos definidos anteriormente. Na Tabela 2 são observados os grupos *a - verdadeiro positivo* (*TP, true positive*) onde o algoritmo identifica corretamente

Figura 9 – Exemplos de imagens com face inválida.



Fonte: *UTKFace dataset* [17]

Figura 10 – Exemplos de imagens sem nenhuma face.



Fonte: *Hotels-50k dataset* [18]

uma face em cada uma das imagens que realmente contém uma face,  $b$  - *falso negativo* (FN, *false negative*) onde o algoritmo erroneamente não reconheceu nenhuma face, apesar das imagens conterem uma face cada,  $c$  - *falso positivo* (FP, *false positive*) onde o algoritmo erroneamente identificou ao menos uma face, mesmo as imagens não contendo nenhuma e  $d$  - *verdadeiro negativo* (TN, *true negative*) onde o algoritmo identificou corretamente que não existia nenhuma face nas imagens. É importante destacar que os quatro grupos destacados na matriz de confusão são mutuamente excludentes [19].

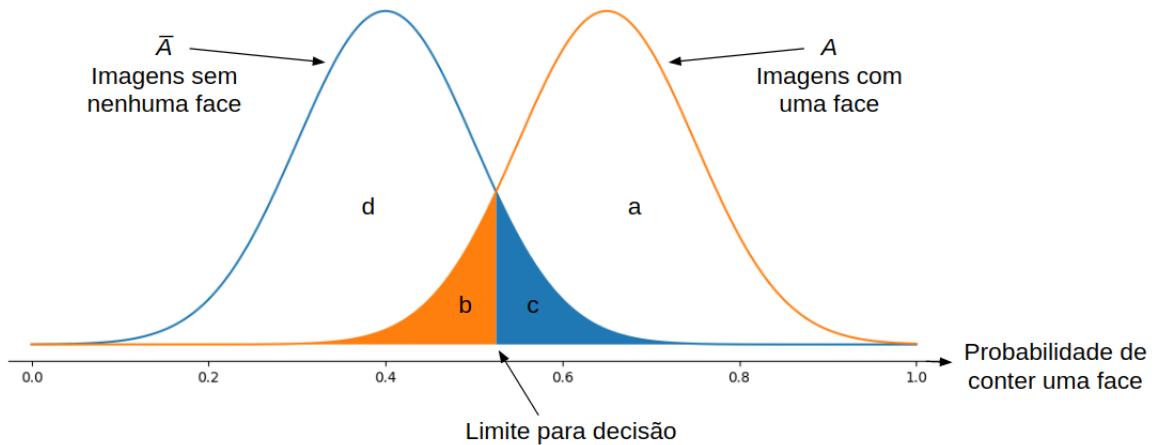
Tabela 2 – Matriz de confusão.

	$B$	$\bar{B}$
$A$	$a$ (verdadeiro positivo)	$b$ (falso negativo)
$\bar{A}$	$c$ (falso positivo)	$d$ (verdadeiro negativo)

Para melhor entender os agrupamentos da matriz de confusão, pode-se observar na Figura 11 as distribuições que representam a quantidade de imagens dos grupos  $A$  e  $\bar{A}$  dada a sua probabilidade de conter uma face.

Figura 11 – Exemplo teórico de uma distribuição normal dos grupos de uma matriz de confusão.

Aqui deve-se supor o uso de apenas uma caracterísca, mostrada de modo ilustrativo.



Fonte: Própria (2020)

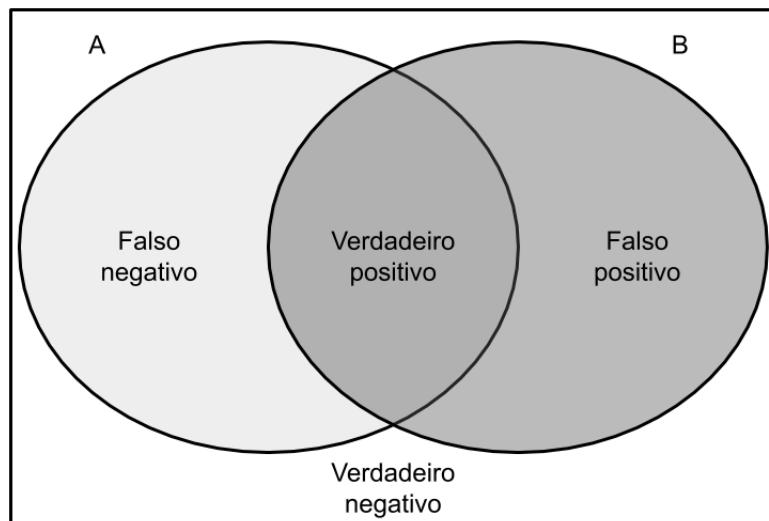
Nas distribuições são destacados os grupos  $b$  (falso negativo) e  $c$  (falso positivo) e fica evidente que, devido a sobreposição das distribuições  $A$  e  $\bar{A}$ , é necessário definir uma limite para decisão, que pode ser ajustado conforme a necessidade, mas que independente do seu ajuste, sempre existirá um grupo categorizado de forma incorreta.

A matriz de confusão também pode ser escrita em termos probabilísticos (Tabela 3), incluindo as probabilidades marginais ou pode ser visualizada no diagrama de Venn correspondente (Figura 12).

Tabela 3 – Matriz de confusão com probabilidades marginais.

	$B$	$\bar{B}$	Soma
$A$	$P(A \cap B)$	$P(A \cap \bar{B})$	$P(A)$
$\bar{A}$	$P(\bar{A} \cap B)$	$P(\bar{A} \cap \bar{B})$	$P(\bar{A})$
Soma	$P(B)$	$P(\bar{B})$	1

Figura 12 – Diagrama de Venn para as imagens analisadas.



Fonte: Própria (2020)

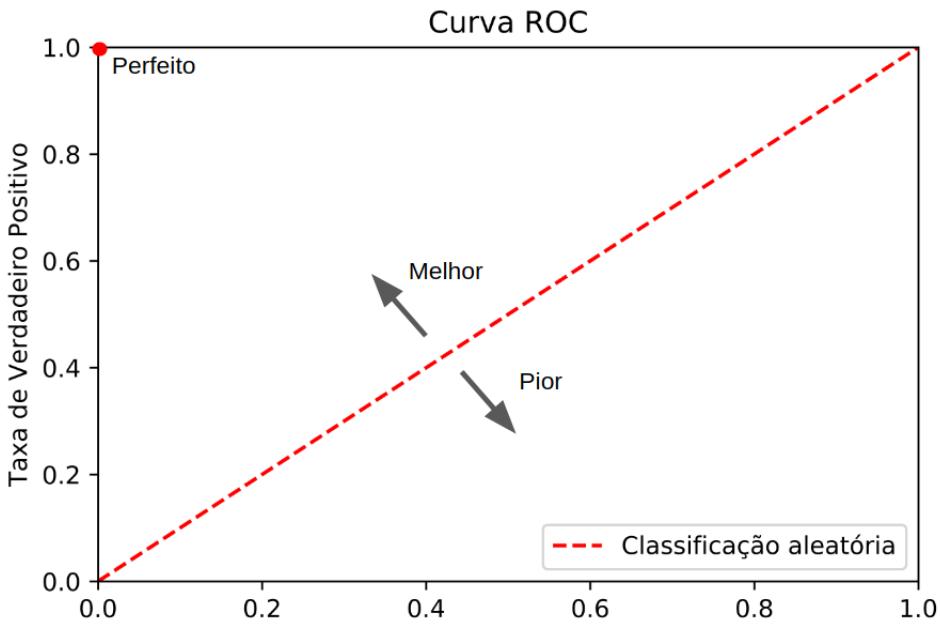
Podem também ser calculadas as medidas tradicionais de sensitividade (a probabilidade condicional do algoritmo identificar ao menos uma face dado que a imagem contém uma face) e especificidade (a probabilidade condicional do algoritmo não identificar nenhuma face em uma imagem que realmente não contém nenhuma face) conforme as Equações 2.3 e 2.4.

$$\text{sensitividade, } P(B|A) = P(A \cap B)/(P(A \cap B) + P(A \cap \bar{B})) = a/(a + b) \quad (2.3)$$

$$\text{especificidade, } P(\bar{B}|\bar{A}) = P(\bar{A} \cap \bar{B})/(P(\bar{A} \cap \bar{B}) + P(\bar{A} \cap B)) = d/(d + c) \quad (2.4)$$

Por fim, a partir da sensitividade e especificidade é possível exibir o resultado do classificador no espaço ROC (*Receiver Operating Characteristic*, traduzido literalmente como Característica operacional do receptor), que consiste em um plano onde o eixo horizontal mede a taxa de falsos positivos (1 - especificidade) e o eixo vertical mede a taxa de verdadeiros positivos (sensitividade), isso permite comparar facilmente diferentes classificadores, pois quanto mais próximo do canto esquerdo superior, melhor a classificação.

Figura 13 – Espaço ROC.



Fonte: Própria (2020)

## 2.5 Análise de Custo

Para entender o valor da utilização de um modelo de reconhecimento facial para filtragem de imagens inválidas, pode-se utilizar o exemplo de uma empresa de cartões de crédito que precisa analisar fotos de clientes antes de fornecer um cartão para os mesmos. Considerando como R o valor presente líquido [20], ou seja, a receita total que este cliente traz para empresa, e H como o custo de trabalho humano para a análise de uma imagem, podemos calcular a receita da empresa antes e depois da aplicação de um modelo de detecção facial. Para as análises feitas neste trabalho, foi utilizado o valor presente líquido de R\$250,00 e o custo de análise de R\$ 1,00 [21].

Antes da aplicação do modelo, todas as fotos enviadas deveriam ser analisadas por um humano, gerando o custo de análise manual H, e apenas as fotos que continham uma face trariam a receita R para empresa, relacionando isso aos valores encontrados na matriz de confusão do modelo, pode-se obter o *ganho inicial* ( $g_0$ ) na Equação 2.5. Nesta equação tem-se o *Evento A* =  $(TP + FN)$ , que representa os casos onde uma foto que realmente contém uma face, multiplicado pela receita de cada cliente R, subtraídos do custo de análise H multiplicado pela quantidade total de fotos recebidas, que equivale a soma de todas as taxas da matriz de confusão.

$$g_0 = (TP + FN) \times R - (TP + FP + FN + TN) \times H \quad (2.5)$$

Após a aplicação do modelo, apenas as fotos que foram classificadas pelo modelo como faces serão analisadas por humanos, gerando custo de análise manual  $H$  e apenas as fotos classificadas pelo modelo como face e que realmente contenham uma face se tornarão clientes reais, trazendo receita  $R$  para a empresa, com isso, pode-se obter o *ganho final* ( $g_1$ ) na Equação 2.6.

$$g_1 = TP \times R - (TP + FP) \times H \quad (2.6)$$

Para avaliar a receita que o modelo traz para a empresa, pode-se calcular a diferença entre a receita antes e depois do modelo, obtendo a Equação 2.7.

$$g_1 - g_0 = (FN + TN) \times H - FN \times R \quad (2.7)$$

Considerando que para o modelo trazer valor para a empresa, é necessário que a receita após sua aplicação seja maior que a receita antes da sua aplicação, obtendo a Equação 2.8.

$$g_1 - g_0 > 0 \quad (2.8)$$

$$(FN + TN) \times H - FN \times R > 0 \quad (2.9)$$

A equação obtida ainda pode ser reescrita em função da *tasa de verdadeiros positivos* (TPR, *true positive rate*), que define quantos resultados verdadeiros positivos ocorrem entre todas as amostras positivas disponíveis, e da taxa de falsos positivos (FPR, *false positive rate*), que define quantos resultados falsos positivos ocorrem entre todas as amostras negativas disponíveis durante o teste. Com esses valores é possível traçar a equação sobre o espaço ROC e visualizar de forma simples quais classificadores satisfazem o critério necessário para obtenção de receita. Para isso são consideradas a quantidade de imagens positivas ( $P = 0.8$ ) e negativas ( $N = 0.2$ ) definidas anteriormente e as relações 2.10 e 2.11, obtendo por fim a Equação 2.12 que pode ser exibida no espaço ROC em 14.

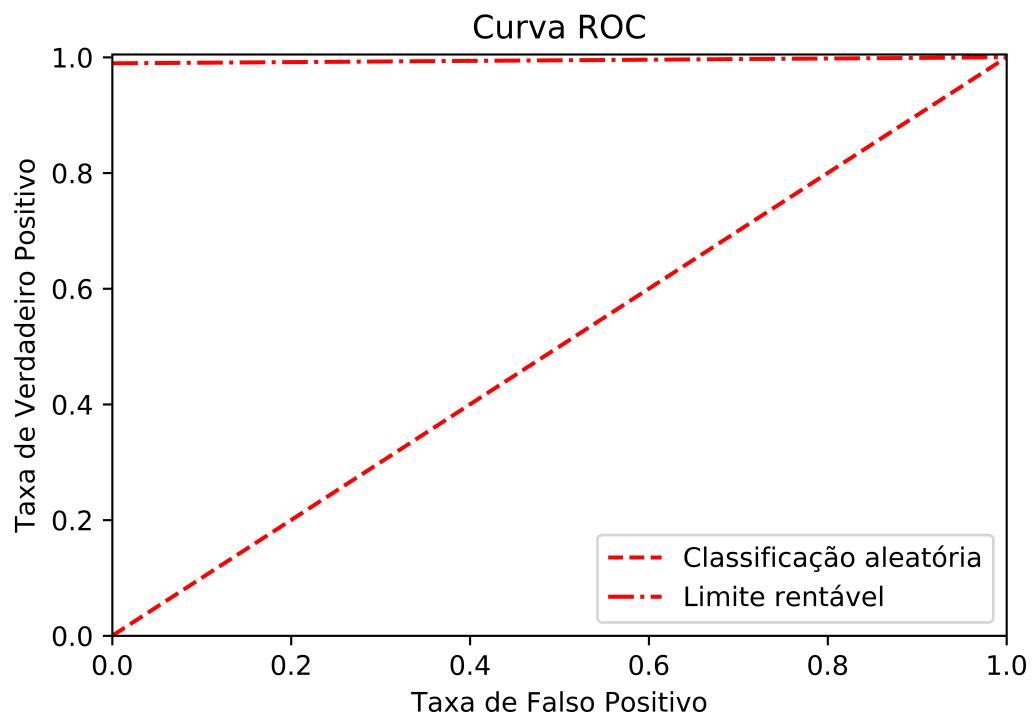
$$FN = P \times (1 - TPR) \quad (2.10)$$

$$TN = N \times (1 - FPR) \quad (2.11)$$

$$TPR = 1 + \left( \frac{N}{P \times \left( \frac{R}{H} - 1 \right)} \times (FPR - 1) \right) \quad (2.12)$$

Assim, é possível avaliar em função da matriz de confusão do modelo, do custo de análise manual de um documento, do valor presente líquido de um cliente e da proporção média de imagens positivas e negativas, se a aplicação deste modelo é viável ou não.

Figura 14 – Limiar lucrativo exibido no espaço ROC.



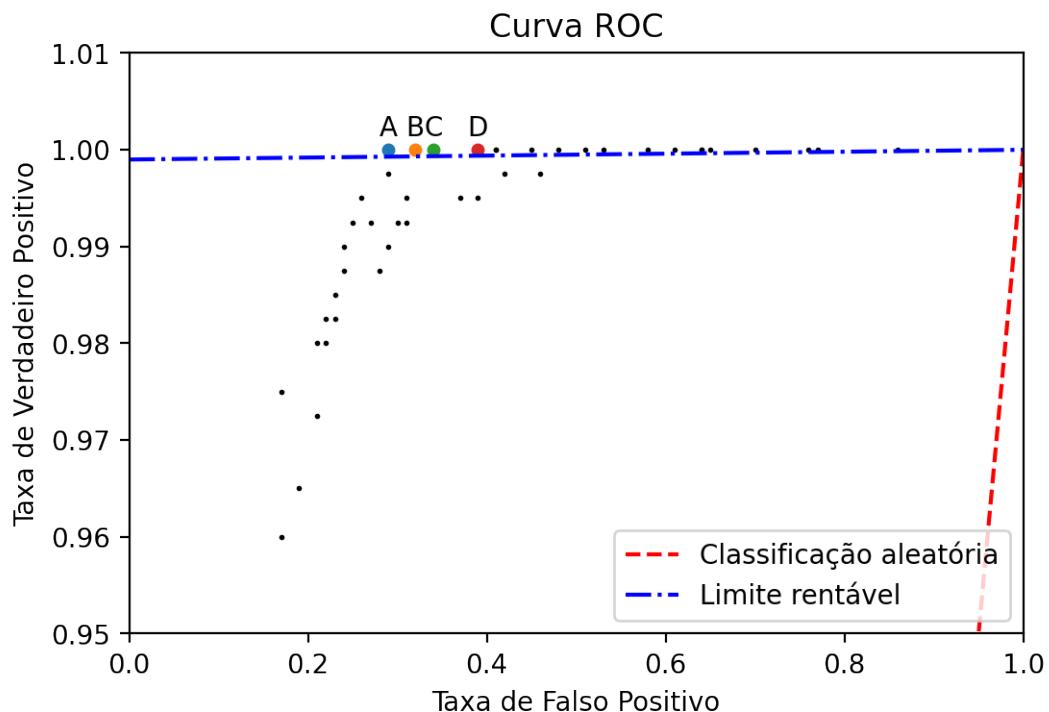
Fonte: Própria (2020)

### 3 Resultados e Discussão

Após entender o funcionamento dos algoritmos e definir a metodologia para análise, foram executadas diversas iterações de classificação sobre o conjunto de imagens, variando os parâmetros de fator de escala (*FE*) para e número mínimo de vizinhos (*MV*), explicados anteriormente, e cada um dos resultados obtidos pode ser exibido como um ponto no espaço ROC obtendo a Figura 15, para que todos sejam facilmente comparados e também se torna simples observar se algum dos resultados está posicionado acima o limite lucrativo traçado.

Figura 15 – Resultados das classificações sobre o espaço ROC.

Os 4 resultados mais lucrativos são identificados por pontos coloridos e o restante por pontos pretos. Os parâmetros utilizados em cada resultado são listados na Tabela 4.



Fonte: Própria (2020)

Analizando os resultados de sensitividade, especificidade e acurácia de cada um dos classificadores na Tabela 4 e a matriz de confusão do resultado que mais lucrativo na Tabela 5, é possível obter algumas conclusões interessantes.

Comparando tal resultado com a análise da equação que define o limiar lucrativo da classificação, é possível perceber que cada imagem positiva que é classificada como negativa gera um prejuízo muito elevado, que precisa ser compensado com a classificação

Tabela 4 – Resultados obtidos com os classificadores.

<sup>1</sup>Lucro refere-se ao lucro obtido por imagem analisada.

Letra	Método	FE	MV	Sensitividade	Especificidade	Acurácia	Lucro <sup>1</sup>
A	Keras	-	-	100,00%	71,00%	94,20%	R\$ 0,142
B	OpenCV	1,10	4	100,00%	68,00%	93,60%	R\$ 0,136
C	OpenCV	1,30	1	100,00%	66,00%	93,20%	R\$ 0,132
D	OpenCV	1,10	3	100,00%	61,00%	92,20%	R\$ 0,122
E	OpenCV	1,25	1	100,00%	59,00%	91,80%	R\$ 0,118
-	OpenCV	1,05	5	100,00%	59,00%	91,80%	R\$ 0,118
-	OpenCV	1,05	4	100,00%	55,00%	91,00%	R\$ 0,110
-	OpenCV	1,10	2	100,00%	52,00%	90,40%	R\$ 0,104
-	OpenCV	1,15	1	100,00%	49,00%	89,80%	R\$ 0,098
-	OpenCV	1,05	3	100,00%	47,00%	89,40%	R\$ 0,094
-	OpenCV	1,10	1	100,00%	42,00%	88,40%	R\$ 0,084
-	OpenCV	1,05	2	100,00%	39,00%	87,80%	R\$ 0,078
-	OpenCV	1,30	0	100,00%	36,00%	87,20%	R\$ 0,072
-	OpenCV	1,25	0	100,00%	36,00%	87,20%	R\$ 0,072
-	OpenCV	1,20	0	100,00%	35,00%	87,00%	R\$ 0,070
-	OpenCV	1,05	1	100,00%	30,00%	86,00%	R\$ 0,060
-	OpenCV	1,15	0	100,00%	24,00%	84,80%	R\$ 0,048
-	OpenCV	1,10	0	100,00%	23,00%	84,60%	R\$ 0,046
-	OpenCV	1,05	0	100,00%	14,00%	82,80%	R\$ 0,028
-	OpenCV	1,10	5	99,75%	71,00%	94,00%	-R\$ 0,356
-	OpenCV	1,15	2	99,75%	58,00%	91,40%	-R\$ 0,382
-	OpenCV	1,20	1	99,75%	54,00%	90,60%	-R\$ 0,390
-	OpenCV	1,10	6	99,50%	74,00%	94,40%	-R\$ 0,848
-	OpenCV	1,20	2	99,50%	69,00%	93,40%	-R\$ 0,858
-	OpenCV	1,15	3	99,50%	63,00%	92,20%	-R\$ 0,870
-	OpenCV	1,05	6	99,50%	61,00%	91,80%	-R\$ 0,874
-	OpenCV	1,30	2	99,25%	75,00%	94,40%	-R\$ 1,344
-	OpenCV	1,15	5	99,25%	73,00%	94,00%	-R\$ 1,348
-	OpenCV	1,15	4	99,25%	70,00%	93,40%	-R\$ 1,354
-	OpenCV	1,25	2	99,25%	69,00%	93,20%	-R\$ 1,356
-	OpenCV	1,15	6	99,00%	76,00%	94,40%	-R\$ 1,840
-	OpenCV	1,20	3	99,00%	71,00%	93,40%	-R\$ 1,850
-	OpenCV	1,20	4	98,75%	76,00%	94,20%	-R\$ 2,338
-	OpenCV	1,25	3	98,75%	72,00%	93,40%	-R\$ 2,346
-	OpenCV	1,30	3	98,50%	77,00%	94,20%	-R\$ 2,834
-	OpenCV	1,20	5	98,25%	78,00%	94,20%	-R\$ 3,330
-	OpenCV	1,25	4	98,25%	77,00%	94,00%	-R\$ 3,332
-	OpenCV	1,25	5	98,00%	79,00%	94,20%	-R\$ 3,826
-	OpenCV	1,20	6	98,00%	78,00%	94,00%	-R\$ 3,828
-	OpenCV	1,25	6	97,50%	83,00%	94,60%	-R\$ 4,814
-	OpenCV	1,30	4	97,25%	79,00%	93,60%	-R\$ 5,320
-	OpenCV	1,30	5	96,50%	81,00%	93,40%	-R\$ 6,810
-	OpenCV	1,30	6	96,00%	83,00%	93,40%	-R\$ 7,802

Tabela 5 – Matriz de confusão com os resultados do classificador mais lucrativo.

	$B$	$\bar{B}$	Soma
$A$	80.0%	00.0%	80%
$\bar{A}$	05.8%	14.2%	20%
Soma	85.8%	14.2%	100%

correta de uma grande quantidade de imagens negativas. Para o classificador estudado, percebe-se que todos os cenários lucrativos apresentam sensitividade igual a 100%, ou seja, todas imagens positivas foram classificadas corretamente.

É curioso notar que o lucro não é diretamente proporcional a acurácia do detector, destacando que a configuração de maior acurácia apresentou prejuízo, tal fato também ocorre devido ao prejuízo elevado de classificar uma imagem positiva de forma incorreta.



# 4 Conclusões e Trabalhos Futuros

## 4.1 Conclusões

Relembrando o objetivo do projeto, que consiste em negar rapidamente imagens que certamente não possuem nenhuma face, e observando os resultados obtidos e analisados, pode-se concluir que tanto o algoritmo de Viola-Jones implementado na biblioteca OpenCV quando o Keras poderiam ser utilizado de forma satisfatória para solucionar tal problema, mesmo utilizando um modelo já treinado, desde que parametrizados da forma correta, permitindo a obtenção lucros interessantes.

Durante o desenvolvimento do projeto foi possível perceber que o ajuste fino de um modelo tem grande impacto no seu resultado final, podendo ser considerada a parte mais importante para um projeto de implementação de um sistema de detecção facial.

Além disso, os testes apenas apresentaram resultados positivos após a seleção correta do conjunto de imagens, o que é um forte indicativo de que o modelo precisa ser preparado especificamente para as necessidades existentes.

## 4.2 Trabalhos Futuros

Para dar continuidade ao desenvolvimento desse projeto, podem ser estudadas outras metodologias de detecção facial ou evoluções da mesma, adicionando técnicas como corte e rotação das imagens originais.

Além disso, dada a necessidade apresentada, onde as imagens devem cumprir algumas características específicas além de apenas conter uma face, torna-se interessante a busca por um conjunto maior de imagens que permitiria o treinamento de um modelo específico para tal problema, que deve apresentar resultados ainda melhores.



# Referências

- 1 PARMAR, D.; MEHTA, B. Face recognition methods & applications. *International Journal of Computer Technology and Applications*, v. 4, p. 84–86, 01 2014. Citado na página 1.
- 2 ZHAO, W. et al. Face recognition: A literature survey. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 35, n. 4, p. 399–458, dez. 2003. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/954339.954342>>. Citado na página 1.
- 3 LI, S. Z.; JAIN, A. K. *Handbook of Face Recognition*. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2011. ISBN 085729931X, 9780857299314. Citado na página 1.
- 4 HJELMÅS, E.; LOW, B. K. Face detection: A survey. *Computer Vision and Image Understanding*, v. 83, n. 3, p. 236 – 274, 2001. ISSN 1077-3142. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S107731420190921X>>. Citado na página 1.
- 5 ITSEEZ. *Open Source Computer Vision Library*. 2015. <<https://github.com/itseez/opencv>>. [Acessado em 31 de Julho de 2019]. Citado na página 3.
- 6 WIKIPEDIA. *OpenCV — Wikipedia, The Free Encyclopedia*. 2019. <<http://en.wikipedia.org/w/index.php?title=OpenCV&oldid=906212825>>. [Acessado em 31 de Julho de 2019]. Citado na página 3.
- 7 ITSEEZ. *The OpenCV Reference Manual*. 2.4.9.0. ed. [S.l.], 2014. Citado na página 3.
- 8 VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: . [S.l.: s.n.], 2001. v. 1, p. I–511. ISBN 0-7695-1272-0. Citado 3 vezes nas páginas 3, 5 e 6.
- 9 FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, Academic Press, Inc., Orlando, FL, USA, v. 55, n. 1, p. 119–139, ago. 1997. ISSN 0022-0000. Disponível em: <<http://dx.doi.org/10.1006/jcss.1997.1504>>. Citado na página 5.
- 10 SANTANA, L. M. Queiroz de; ROCHA, F. Processo de detecção facial utilizando viola;jones. *Interfaces Científicas - Exatas e Tecnológicas*, v. 1, 02 2015. Citado na página 5.
- 11 GU, J. et al. Recent advances in convolutional neural networks. *Pattern Recognition*, Elsevier, v. 77, p. 354–377, 2018. Citado na página 7.
- 12 LAWRENCE, S. et al. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, IEEE, v. 8, n. 1, p. 98–113, 1997. Citado na página 8.
- 13 IDE, H.; KURITA, T. Improvement of learning for cnn with relu activation by sparse regularization. In: IEEE. *2017 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2017. p. 2684–2691. Citado na página 8.

- 14 VARGAS, A. C. G.; PAES, A.; VASCONCELOS, C. N. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. In: *Proceedings of the XXIX Conference on Graphics, Patterns and Images*. [S.l.: s.n.], 2016. v. 1, n. 4. Citado na página 8.
- 15 ZHANG, K. et al. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, v. 23, n. 10, p. 1499–1503, Oct 2016. ISSN 1070-9908. Citado 2 vezes nas páginas 9 e 10.
- 16 CENTENO, I. de P. *MTCNN face detection implementation for TensorFlow*. 2020. <<https://github.com/ipazc/mtcnn>>. [Acessado em 15 de Fevereiro de 2020]. Citado na página 9.
- 17 IMAGING, A.; PROCESSING, C. I. *UTK Face Dataset*. [Acessado em 04 de Agosto de 2019]. Disponível em: <<https://susanqq.github.io/UTKFace/>>. Citado 4 vezes nas páginas 9, 11, 12 e 13.
- 18 STYLIANOU, A. et al. Hotels-50k: A global hotel recognition dataset. *CoRR*, abs/1901.11397, 2019. Disponível em: <<http://arxiv.org/abs/1901.11397>>. Citado 2 vezes nas páginas 11 e 13.
- 19 DOUGHERTY, G. *Pattern Recognition and Classification: An Introduction*. [S.l.]: Springer Publishing Company, Incorporated, 2012. ISBN 1461453224, 9781461453222. Citado na página 14.
- 20 ZIZLAVSKY, O. Net present value approach: Method for economic assessment of innovation projects. *Procedia - Social and Behavioral Sciences*, v. 156, 11 2014. Citado na página 16.
- 21 HUSSAIN, K. Valuation of a bank credit-card portfolio. *Journal of American Academy of Business*. v10, p. 29–35, 2007. Citado na página 16.