Universidade Federal da Bahia
Instituto de Matemática e Estatística

Programa de Graduação em Ciência da Computação

DETECÇÃO DE ORELHAS EM IMAGENS 2D UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

Ubiratan Correia Barbosa Neto

TRABALHO DE GRADUAÇÃO

Salvador
27 de junho de 2019

UBIRATAN CORREIA BARBOSA NETO

DETECÇÃO DE ORELHAS EM IMAGENS 2D UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

Este Trabalho de Graduação foi apresentado ao Programa de Graduação em Ciência da Computaçãoda Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Maurício Pamplona Segundo

Salvador
27 de junho de 2019

Sistema de Bibliotecas - UFBA

Ubiratan Correia Barbosa Neto.

Detecção de Orelhas em Imagens 2D Utilizando Redes Neurais Convolucionais / Ubiratan Correia Barbosa Neto – Salvador, 2019.

41p.: il.

Orientador: Prof. Dr. Maurício Pamplona Segundo.

Monografia (Graduação) – Universidade Federal da Bahia, Instituto de Matemática e Estatística, 2019.

1. detecção de orelhas. 2. biometria. 3. aprendizado de máquina. 4. redes neurais convolucionais.. I. Maurício Pamplona Segundo. II. Universidade Federal da Bahia. Instituto de Matemática e Estatística. III. Detecção de Orelhas em Imagens 2D Utilizando Redes Neurais Convolucionais.

CDD – XXX.XX

CDU – XXX.XX.XXX

TERMO DE APROVAÇÃO

UBIRATAN CORREIA BARBOSA NETO

DETECÇÃO DE ORELHAS EM IMAGENS 2D UTILIZANDO REDES NEURAIS CONVOLUCIONAIS

Este Trabalho de Graduação foi julgado adequado à obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo Programa de Graduação em Ciência da Computaçãoda Universidade Federal da Bahia.

Salvador, 27 de junho de 2019

Prof. Dr. Maurício Pamplona Segundo
Universidade Federal da Bahia

Prof. Dr. Rubisley de Paula Lemes
Universidade Federal da Bahia

Me. Lucas Amparo Barbosa
Senai Cimatec

AGRADECIMENTOS

Agradeço inicialmente aos meus pais, Maria Olívia e João Evangelista, à minha avó, Olívia da Costa Barbosa, e a todos os meus familiares, por todo o apoio às minhas decisões e escolhas ao longo de toda a minha vida. Agradeço aos meus amigos e minha namorada, Raquel, por sempre estarem presentes até nos meus momentos mais difíceis. O apoio de cada um foi fundamental nesse processo, seja através de conselhos, preocupação, carinho ou dicas. A presença de todos foi influente na minha tomada de decisões e escolhas.

Agradeço à Universidade Federal da Bahia e todo o corpo docente, que me proporcionou a oportunidade de fazer e concluir o curso. Em especial ao meu orientador Maurício Pamplona, que fez com que fosse possível o desenvolvimento deste trabalho, me deu a oportunidade de trabalhar no laboratório IVISION e me apresentou a maratona de programação, uma das coisas mais marcantes do meu período de graduação.

Agradeço aos amigos que conheci desde o início do curso, alguns desde o dia da matrícula no curso. Em especial à Pedro, Bruno, Felipe, Gabriel Dahia, Gabriel Pinho e Victor, que me acompanharam em diversas aulas e atividades extracurriculares, sempre ajudando uns aos outros durante todo o curso, alguns estando próximos até hoje.

Aos colegas e amigos do laboratório IVISION, que sempre criaram um ambiente de pesquisa divertido, compartilhando conhecimento sem se esquecer do bom humor. Foram incontáveis as vezes que permaneci mais tempo do que o planejado no laboratório para que pudesse ajudar, ser ajudado ou apenas jogar conversa fora com cada um.

Aos colegas e amigos do GRUPRO, nosso grupo de programação competitiva. Foram diversas horas de treino, viagens e competições que renderam momentos felizes, tristes e engraçados, porém inesquecíveis. Este grupo foi um dos que mais me motivou a continuar e ultrapassar meus limites, sempre em busca de conhecimento. Agradeço aos professores Rubisley de Paula e, mais uma vez, Maurício Pamplona pela presença e apoio a este grupo. Agradeço também a Roberto e Jonathan, por serem a ponte que me conectou à programação competitiva em geral e por me ensinarem as primeiras coisas que aprendi nessa área.

Agradeço a todos os outros amigos e colegas que conheci durante a graduação, aqueles que só encontrava por poucos minutos ao sair de uma aula, ou nos corredores enquanto ia para o próximo compromisso. Às vezes, encontrar alguém conhecido e trocar algumas palavras eram poucos minutos de alívio e descanso em um dia cheio.

À empresa NVIDIA, pela disponibilização de uma placa de vídeo que foi um componente essencial durante o projeto.

RESUMO

A detecção de orelhas é um estágio importante num sistema de reconhecimento que utiliza essa biometria. Apesar da existência de vantagens da utilização da orelha em relação à faces e impressões digitais, é necessário a existência de métodos de detecção mais eficientes para a composição de um sistema de reconhecimento utilizando orelhas. Trabalhos relacionados a esse tema aparecem em pequena quantidade, e muitos não podem ser utilizados em sistemas de reconhecimento por conta do tempo de execução elevado ou resultados ainda insatisfatórios. Este trabalho propõe um método para detecção de orelhas utilizando redes neurais convolucionais, testado em uma base que contém imagens retiradas de um ambiente não controlado. Utilizamos duas redes neurais: uma para propor regiões com maior probabilidade de serem orelhas e uma para definir se devemos ou não utilizar cada uma das regiões propostas. O processo de detecção leva em torno de 132 milissegundos por imagem em uma máquina pouco robusta. Conseguimos detectar 86.27% das orelhas da base considerando um IOU maior ou igual a 40%, com uma de taxa de falsa detecção de 9.98%. Comparamos nosso método com alguns dos mais recentes encontrados na literatura. Os resultados obtidos ainda não tornam o método viável a ser utilizado por sistemas de reconhecimento, porém superam os resultados e tempo de execução alcançados por alguns dos trabalhos comparados.

Palavras-chave: detecção de orelhas; biometria; aprendizado de máquina; redes neurais convolucionais.

ABSTRACT

Ear detection is an important step in a recognition system that uses this biometry. Despite the advantages of using the ear compared to faces and fingerprints, it is necessary the existence of detection methods more efficient for the composition of an recognition system using ears. Related work appear in small quantity, and many can not be used in recognition systems due to slow execution time or unsatisfactory results. This work proposes an ear detection method using convolutional neural networks, tested on a base containing images taken from an uncontrolled environment. We used two neural networks: one to propose regions that are more likely to be ears and another to define whether or not to use each of the proposed regions. The detection process takes around 132 millisseconds per image on a not so powerful machine. We were able to detect 86.27% of the ears from the base considering an IOU greater or equal to 40%, with a false detection rate of 9.98%. We compared our method with some of the most recent ones. The results still do not make the method feasible to be used on recognition systems, but they surpass the results and execution time achieved by some of the compared works.

Keywords: ear detection; biometry; machine learning; convolutional neural networks.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO

Biometria pode ser definida como o estudo de características físicas e comportamentais de um indivíduo. Uma de suas aplicações é na área de reconhecimento, e uma das mais famosas são as impressões digitais. Porém existem estudos acerca de outros tipos de biometria, como por exemplo faces (Wang e Deng, 2018), que estão se tornando cada vez mais populares. Estudos sobre reconhecimento baseado em orelhas também podem ser encontrados (Emeršič, Struc e Peer, 2016), porém menos numerosos.

Entretanto, a utilização da orelha para o reconhecimento possui vantagens em relação ao uso da face. A orelha sofre poucas mudanças ao longo da vida do indivíduo, e já existem estudos que tentam aplicar esta técnica em recém-nascidos (Tiwari *et al.*, 2015). Em contrapartida, à medida que o tempo passa, a face de um indivíduo sofre diversas mudanças, que comprometem o funcionamento de um sistema de reconhecimento. Tanto é que há estudos que tentam utilizar a informação da orelha para auxiliar métodos baseados em face (Fan, Mu e Yang, 2017). Além disso, é uma biometria ainda menos invasiva, e imagens da orelha podem ser obtidas em diferentes posições. Outro aspecto é a privacidade. O compartilhamento e a captura de imagens das orelhas geram menos desconforto e desaprovação, visto que, diferente das faces, é mais difícil reconhecer indivíduos através dessa informação a olho nu.

Em um sistema de reconhecimento, a detecção da orelha é um dos estágios iniciais e fundamentais, e um detector falho interfere na eficácia do módulo de reconhecimento. No entanto, trabalhos relacionados a esse problema são pouco numerosos. Logo, métodos de detecção mais eficientes são necessários para alavancar estudos sobre reconhecimento baseado em orelha e despertar o interesse de outros pesquisadores pelo assunto.

Porém, a detecção da orelha é mais complicada do que a face devido ao seu tamanho. Outras regiões da face, como o nariz e a região da bochecha, e as mãos podem ser confundidas com uma orelha. Alguns métodos tentam utilizar técnicas para reduzir essa interferência durante a detecção, como é o caso de Zhang e Mu (2017). Eles detectam outras regiões além da orelha na imagem, selecionando apenas as detecções de orelha que estão contidas nessas outras regiões.

A maior parte dos trabalhos nas áreas de reconhecimento de padrões e aprendizado de máquina atualmente se baseiam em redes neurais. Elas se tornaram tão comuns que são utilizadas para a realização das mais variadas tarefas. Alguns exemplos são detectores de objetos em geral (Liu *et al.*, 2018), reconhecimento baseado em face (Wang e Deng, 2018) e detecção de *malwares* (Saxe e Berlin, 2015). Devido a sua crescente popularidade e taxa de sucesso, o uso de redes neurais também foi o caminho escolhido para esse projeto.

Este trabalho, então, propõe um método para detecção de orelhas utilizando redes neurais convolucionais (CNNs). Mostramos os resultados obtidos em uma base de dados que possui imagens com variação de pose, iluminação e quantidade de orelhas presentes, retiradas de um ambiente não controlado. Além disso, comparamos nossos resultados com outros métodos da literatura encontrados durante o processo de projeto e desenvolvimento do nosso detector.

Capítulo 2

REFERENCIAL TEÓRICO

Este capítulo tem como objetivo apresentar, de forma breve, trabalhos relacionados à detecção de orelhas na Seção 2.1, além de discutir sobre dois trabalhos de detecção de objetos em geral que foram usados como referência para a criação do nosso detector na Seção 2.2.

2.1 TRABALHOS RELACIONADOS

Os estudos mais antigos relacionados à detecção de orelha trabalhavam em cima de imagens de rostos de perfil. O trabalho de Chen e Bhanu (2004) extrai informações da forma das orelhas de um conjunto de imagens e as transforma em uma representação de histograma, utilizando-o como modelo para compará-lo a regiões com potencial de serem orelhas. Testando em imagens de perfil, eles obtiveram um resultado de 91.5% de taxa de acerto, calculada como sendo a porcentagem da imagem corretamente classificada como orelha ou como fundo, e 2.52% de taxa de alarme falso, porcentagem de fundo da imagem erroneamente classificada como orelha. Yuan e Mu (2007) assumem que o formato da orelha é semelhante a uma elipse, e realizam detecção de bordas para localizar regiões nesse formato. Eles não realizam uma avaliação adequada, testando apenas utilizando uma *webcam* com suas próprias imagens.

Islam, Bennamoun e Davies (2008) realizam a detecção em imagens de perfil utilizando uma cascata de classificadores *Haar*, similar ao famoso método de Viola-Jones para detecção de faces (Viola e Jones, 2001). Eles reportam um resultado de 100% de taxa de detecção na base da UND[1], porém detalhes acerca de como o cálculo da acurácia foi realizado não foram mostrados. Yuan e Zhang (2009) também utilizam cascata de classificadores *Haar* para detectar orelhas, porém propõem melhorias e reportam resultados em três diferentes bases. Seus piores resultados foram 3% de taxa de falsa rejeição e 3.6% de taxa de falsa aceitação em uma das bases testadas. Novamente, não foram divulgados detalhes sobre as métricas utilizadas. O trabalho de Abaza, Hebert e Harrison (2010) utiliza essa mesma técnica, porém realiza diversas melhorias em relação ao

[1] https://cvrl.nd.edu/projects/data/

tempo de treino e execução. Eles obtêm como resultado 5.31% de taxa de falsa rejeição e 3.5% de taxa de falsa aceitação em 2113 imagens de quatro diferentes bases de dados de imagens de perfil. Esses valores foram calculados em relação ao número de subregiões classificadas pelo método durante a detecção.

Pflug, Winterstein e Busch (2013) utilizam, diferente dos trabalhos anteriormente citados, informações de curvatura na imagem de profundidade junto com detecção de bordas na imagem colorida. Essas informações são combinadas e é atribuído um valor de confiança para cada região escolhida, baseado na similaridade com as características de uma orelha real. É escolhida como orelha a região com maior valor de confiança. Eles reportam um resultado de 99% de taxa de detecção na base da UND-J2[2], este valor sendo a porcentagem de detecções corretas do algoritmo considerando uma sobreposição maior ou igual a 50% em relação à marcação da base. Este valor é calculado através da fórmula $\frac{2*|G \cap R|}{|G|+|R|}$, onde G corresponde à marcação da base e R a região proposta pelo detector. Essa métrica, no entanto, não leva em consideração a detecção de falsos positivos, e nenhuma outra métrica foi utilizada para a avaliação do método.

Recentemente começaram a surgir métodos baseados em *deep learning* e redes neurais, além de tentarem resolver o problema para imagens capturadas em ambientes não controlados. Emeršič *et al.* (2018) apresentam um método para detecção de orelhas utilizando uma rede *encoder-decoder* que classifica cada *pixel* da imagem como sendo parte de uma orelha ou não. O método é capaz de detectar ambas as orelhas, caso presentes, em uma imagem com uma única pessoa. Os resultados foram obtidos utilizando a base da AWE[3]. É reportada uma acurácia média de 99.21%, esta sendo a taxa de *pixels* classificados corretamente. Outras três métricas são reportadas. A primeira delas é um IOU (*Intersection over Union*) médio de 48.31%, que corresponde à porcentagem de *pixels* classificados corretamente como sendo parte de uma orelha no total de *pixels* da união entre a predição do detector e a marcação da base. A segunda é uma precisão média de 60.83%, porcentagem de *pixels* classificados corretamente dentre todos os *pixels* classificados como parte de uma orelha pelo método. A última corresponde ao *recall*, com um valor médio de 75.86%, que indica a porcentagem de *pixels* de orelha na marcação da base que foram encontrados (classificados corretamente como sendo parte de uma orelha) pelo método.

Wang, Du e Huang (2017) realizam a classificação de diversas subregiões de tamanho fixo da imagem, utilizando uma técnica de janela deslizante. Esta técnica é otimizada utilizando redes convolucionais para realizar a classificação de todas as janelas de uma só vez. Ela é aplicada na mesma imagem em múltiplas escalas. Eles obtém uma acurácia de 98.19%, precisão de 85.97% e *recall* de 83.25%. As métricas foram medidas da mesma forma do trabalho anterior, porém considerando as janelas classificadas e não os *pixels* da imagem. Contudo, não foi especificado o critério para diferenciar uma detecção correta de uma incorreta.

Zhang e Mu (2017) propõem um método de detecção dividido em diversas etapas. Além da região da orelha, a região da cabeça e uma região intermediária incluindo a

[2]https://cvrl.nd.edu/projects/data/#nd-collection-j2

[3]http://awe.fri.uni-lj.si/download

Tabela 2.1 Resumo dos trabalhos relacionados à detecção de orelha.

Autores	Método	Base	Resultados	Observações
Chen e Bhanu (2004)	Representação das orelhas em forma de histograma	Imagens próprias	91.5% de taxa de detecção 2.52% de taxa de alarme falso	Resultado é relacionado à porcentagem da imagem
Yuan e Mu (2007)	Detecção de bordas + rastreamento CAMSHIFT	Imagens próprias	Não especificado	—
Islam, Bennamoun e Davies (2008)	Cascata de Classificadores Haar	UND	100% de taxa de detecção	—
Yuan e Zhang (2009)	Cascata de Classificadores Haar	CAS-PEAL (Gao *et al.*, 2008) UMIST[4] USTB[5]	3% de taxa de falsa rejeição e 3.6% de taxa de falsa aceitação	Resultado na base da CAS-PEAL(pior resultado dentre as três)
Abaza, Hebert e Harrison (2010)	Cascata de Classificadores Haar	UMIST UND WVHTF (Abaza, Hebert e Harrison, 2010) USTB	5.31% de taxa de falsa rejeição e 3.5% de taxa de falsa aceitação	Resultado une as quatro bases
Pflug, Winterstein e Busch (2013)	Extração de informações da orelha em imagens coloridas e de profundidade	UND-J2	99% de taxa de detecção	Detecção é correta se possuir sobreposição de 50% com a marcação
Emeršič *et al.* (2018)	Rede Convolucional Encoder-Decoder	AWE	acurácia de 99.21% IOU de 48.31% precisão de 60.83% *recall* de 75.86%	Resultados medidos a nível de *pixel*
Wang, Du e Huang (2017)	Rede totalmente convolucional (FCN)	AWE	acurácia de 98.19% precisão de 85.97% *recall* de 83.25%	Resultados medidos em relação à quantidade de janelas
Zhang e Mu (2017)	Detecção em múltiplos níveis usando redes neurais convolucionais	WebEar (Zhang e Mu, 2017) UBEAR UND-J2	acurácia de 98.66% precisão de 99.22% *recall* de 98.66%	Resultados em relação ao número de detecções
Ganapathi *et al.* (2018)	\textit{Ensemble} com três modelos de redes neurais	AWE IIT Indore Collection-A	acurácia de 98.20%	Resultado da IIT Indore Collection-A (pior resultado)

orelha também são detectadas. Elas são usadas para filtrar as detecções de orelha, deixando apenas as que se encontram dentro dessas outras duas regiões. São reportados resultados em três bases de dados, sendo um deles a base da UBEAR (Raposo *et al.*, 2011). Eles conseguem obter uma acurácia de 98.66%, precisão de 99.22% e *recall* de 98.66%, calculados em relação ao número de detecções. Novamente, não foi especificado como detecções corretas e incorretas foram diferenciadas.

Todos os estudos mencionados anteriormente, exceto aqueles baseados em cascata de classificadores *Haar*, reportam resultados em imagens que possuem apenas uma orelha presente, ou até duas orelhas, sendo estas a orelha esquerda e direita da mesma pessoa. Ademais, um deles identifica a região ao redor da orelha como um dos passos para a detecção (Zhang e Mu, 2017). Ganapathi *et al.* (2018) utilizam a técnica de *ensemble*, união de vários modelos, para combinar os resultados de três detectores de orelhas. Cada um tem a tarefa de atribuir um valor de probabilidade da imagem dada ser uma orelha ou não. Essa informação é então combinada e é gerada a região da orelha. Eles reportam a acurácia em duas bases, 99.52% na AWE, e 98.20% na IIT Indore Collection-A (Ganapathi *et al.*, 2018). Este valor é calculado em relação ao número de orelhas detectadas, classificando a detecção como correta se o IOU entre a predição e a marcação da base for superior a 80%. Nesse trabalho são mostrados alguns resultados da detecção quando há presença de mais de uma orelha de diferentes pessoas na imagem.

A Tabela 2.1 contém um resumo dos métodos comentados nessa seção.

2.2 YOLO E FASTER R-CNN

Duas redes de detecção de objetos em geral também foram estudadas e utilizadas como referência durante o processo de construção do nosso detector. A YOLO (Redmon *et al.*, 2016) e a Faster R-CNN (Ren *et al.*, 2017) realizam essa tarefa muito bem de formas distintas. A Faster R-CNN utiliza uma *Region Proposal Network* (RPN) para propor partes da imagem que possam conter os objetos buscados e, acoplada a essa rede, realiza

[4]https://www.sheffield.ac.uk/eee/research/iel/research/face

[5]http://www1.ustb.edu.cn/resb/en/visit/visit.htm

a classificação do objeto presente na região e a regressão de uma caixa retangular delimitando o objeto. Todas essas etapas podem ser realizadas a 17 quadros por segundo, dependendo da arquitetura da rede utilizada internamente, em uma máquina com placa gráfica Nvidia Tesla K40. A YOLO, por sua vez, divide a imagem de entrada em seções, cada uma possuindo uma quantidade fixa de preditores. Cada preditor tem a responsabilidade de detectar os objetos cujo centro está contido em sua seção, além de classificar o objeto. Ela possui esse nome pois realiza todo o processamento passando a imagem uma única vez pela rede (YOLO, *You Only Look Once*). A YOLO consegue alcançar uma taxa de 45 quadros por segundo em uma máquina com placa gráfica Nvidia GeForce GTX Titan X. Uma versão alternativa e simplificada da mesma rede, a *Fast* YOLO, que alcança uma taxa de 150 quadros por segundo também é apresentada no mesmo trabalho. Versões mais recentes, como a YOLOv2 e a YOLO9000 (Redmon e Farhadi, 2017), e a YOLOv3 (Redmon e Farhadi, 2018), resolvem alguns problemas da versão original e melhoram seus resultados. Porém, para a compreensão deste trabalho, é necessário apenas o conhecimento da estrutura básica da rede original.

Capítulo 3

VERSÕES PRELIMINARES

O desenvolvimento do método ocorreu de forma gradativa, partindo de um detector mais simples e realizando mudanças a fim de aperfeiçoá-lo. Neste capítulo abordaremos o processo de construção das versões preliminares do detector, contendo uma breve descrição e abrangendo suas características e limitações. A versão final será apresentada no Capítulo 4, contará com mais detalhes e será o foco até o fim deste documento.

3.1 BASE DE DADOS

A base de dados utilizada no trabalho é a *In-the-wild Ear Database*[1]. Algumas imagens dessa base podem ser vistas na Figura 3.1. Ela possui 2663 imagens divididas em duas coleções. A Coleção A contém 605 imagens retiradas do Google Imagens com marcações manuais de 55 pontos em cada uma delas. Estas marcações são referentes à locais específicos na orelha e possuem maior importância para métodos de reconhecimento. Porém, pudemos extrair a região da orelha a partir desses pontos, encontrando o menor retângulo que os contém. A Coleção A ainda é subdividida em 2 pastas. A primeira possui 500 imagens e a segunda possui 105, e foram utilizadas por nós como treino e validação, respectivamente. A Coleção B possui 2058 imagens de 231 celebridades, e conta apenas com a marcação de 4 pontos delimitando a região de uma orelha da imagem num formato retangular. As imagens foram selecionadas manualmente de forma que as orelhas estivessem visíveis e que sua região pudesse ser encontrada por um simples detector baseado em *Histogram of Oriented Gradients* (Dalal e Triggs, 2005). As imagens da base possuem diferentes resoluções, além de variedade de pose e iluminação. Por mais que exista apenas uma orelha marcada por imagem, uma grande porção da base é composta por imagens com mais de uma orelha presente. Todos esses fatores tornam-a uma base adequada para nosso trabalho, que tem como objetivo final detectar orelhas em condições irrestritas. Problemas com a base, como a ausência de marcações citada acima, foram resolvidos no decorrer do trabalho quando houve necessidade, e serão abordados nos respectivos capítulos.

[1] https://ibug.doc.ic.ac.uk/resources/ibug-ears/

Figura 3.1 Algumas imagens da base de dados utilizada no trabalho.

3.2 VERSÃO INICIAL - DETECÇÃO DE UMA ORELHA NUMA FACE

Nessa etapa nos preocupamos em confeccionar um detector de orelhas capaz de encontrar uma orelha em uma face. Para a localização e recorte da face na imagem durante o treino, utilizamos as marcações das orelhas para todas as imagens de treino e validação. A partir das marcações, aumentamos o tamanho da região nas quatro direções (cima, baixo, direita e esquerda) a fim de obter a face inteira dentro dela, porém tentando evitar a presença de muita informação de fundo. Estes valores de aumento são fixos, porém levam em consideração o tamanho da região da orelha em cada imagem. Evitamos utilizar um detector de faces para realizar o recorte pois podem acarretar em erros e afetar negativamente o treino. Uma ilustração do procedimento pode ser vista na Figura 3.2 e alguns exemplos aparecem na Figura 3.3.

3.2.1 Arquitetura da Rede e Treinamento

Em posse das imagens recortadas, treinamos uma rede neural convolucional para realizar o nosso objetivo descrito anteriormente. A rede recebe como entrada imagens 128×128, se fazendo necessário o redimensionamento de todas as imagens de face. A saída consiste de 4 valores $[x, y, w, h]$ por imagem, indicando a região proposta pela rede para a localização da orelha. A região possui um formato retangular, comumente utilizado em

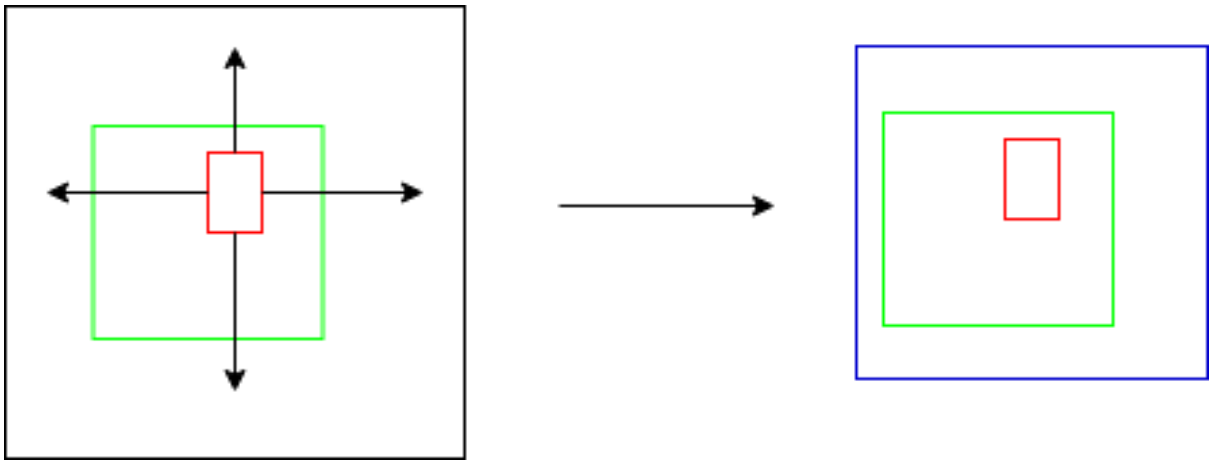

Figura 3.2 Ilustração do processo de recorte. O retângulo preto corresponde à imagem, o verde à face e o vermelho à orelha. A partir da orelha são utilizados valores fixos de aumento, e então recortamos a região da imagem. O retângulo azul corresponde à região recortada da imagem.

(a) Imagem 1 original (b) Imagem 2 original (c) Imagem 3 original

(d) Imagem 1 recortada (e) Imagem 2 recortada (f) Imagem 3 recortada

Figura 3.3 Resultado do processo de recorte da face na imagem. A primeira linha contém as imagens originais. A segunda contém as imagens recortadas.

métodos de detecção de objetos. É conhecida pelo nome *bounding box*, a menor região que engloba o objeto de interesse. Os valores x, y correspondem à coordenada do canto superior esquerdo da região, enquanto os valores w, h correspondem à largura e altura, respectivamente. Estes valores estão normalizados entre $[0, 1]$. Para obtermos os valores na escala desejada, fazemos a seguinte conversão: sejam W e H a largura e altura da imagem, respectivamente, e $[x, y, w, h]$ a saída da rede, podemos obter as coordenadas (x_l, y_l) e (x_r, y_r), o canto superior esquerdo e o canto inferior direito da região retornada pela nossa rede neural, respectivamente, através das Equações 3.1, 3.2, 3.3 e 3.4. No nosso caso atualmente, $H = 128$ e $W = 128$.

$$x_l = x * W \tag{3.1}$$
$$x_r = x_l + w * W \tag{3.2}$$
$$y_l = y * H \tag{3.3}$$
$$y_r = y_l + h * H \tag{3.4}$$

A Tabela 3.1 detalha a arquitetura da rede utilizada nessa versão. O tamanho do *batch* escolhido foi 16 e a rede foi treinada com a *L2 Loss*, também chamada de *Mean Squared Error*. Ela é calculada da seguinte forma: sejam $[\hat{x}, \hat{y}, \hat{w}, \hat{h}]$ os valores de saída da rede, $[x, y, w, h]$ os valores esperados (a marcação da orelha), e N o tamanho do *batch*, o *loss* $\mathcal{L}$ é definido através da Equação 3.5.

$$\mathcal{L} = \frac{\sum_{i=1}^{N}(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2 + (w_i - \hat{w_i})^2 + (h_i - \hat{h_i})^2}{N} \tag{3.5}$$

3.2.2 Resultados e Discussão

Durante a inferência utilizamos um detector de faces pré treinado para realizar o recorte nas imagens, visto que queremos simular nesta etapa o funcionamento do detector de forma geral. O detector de faces utilizado se encontra na biblioteca *cvlib*[2], de código aberto e com licença MIT. Para avaliarmos nosso método durante a fase inicial calculamos apenas o IOU, pois nosso objetivo era apenas ter uma noção do progresso que estávamos fazendo enquanto caminhávamos para a versão final do detector. O IOU (*Intersection over Union*) é uma métrica utilizada frequentemente em avaliação de performance em trabalhos relacionados à detecção de objetos. Sejam B_1 e B_2 a região marcada e a predição do detector, respectivamente, o IOU(B_1, B_2) pode ser obtido através da fórmula:

$$IOU(B_1, B_2) = \frac{Inter(B_1, B_2)}{Union(B_1, B_2)} \tag{3.6}$$

onde $Inter(B_1, B_2)$ é a área de interseção entre as duas regiões, e $Union(B_1, B_2)$ é a área de união. O IOU faz com que tanto detecções em falta, causadas por regiões menores que

[2]https://www.cvlib.net/

Tabela 3.1 Arquitetura da Versão Inicial da Rede

	Entrada	Filtros	*Strides*	Saída
Convolução + *ReLU*	128x128x3	5x5x3x64	1	128x128x64
Max Pooling	128x128x64	2x2	2	64x64x64
Convolução + *ReLU*	64x64x64	3x3x64x128	1	64x64x128
Max Pooling	64x64x128	2x2	2	32x32x128
$2x$(Convolução + *ReLU*)	32x32x128	3x3x128x256	1	32x32x256
Max Pooling	32x32x256	2x2	2	16x16x256
$3x$(Convolução + *ReLU*)	16x16x256	3x3x256x512	1	16x16x512
Max Pooling	16x16x512	2x2	2	8x8x512
$5x$(Convolução + *ReLU*)	8x8x512	2x2x512x1024	1	8x8x1024
Max Pooling	8x8x1024	2x2	2	4x4x1024
Flattening	4x4x1024	—	—	16384
Fully Connected + ReLU	16384	—	—	512
Fully Connected + ReLU	512	—	—	1024
Dropout 10%	1024	—	—	1024
Fully Connected + Sigmoid	1024	—	—	4

a orelha, e detecções em excesso, causadas por regiões que cobrem mais do que a orelha, influenciem negativamente o resultado.

Conseguimos realizar a tarefa descrita com média de 70% de IOU entre todas as imagens da base de teste. A Figura 3.4 mostra alguns resultados do nosso algoritmo, incluindo detecções bem sucedidas com diferentes porcentagens de IOU e casos de falha. Falhas durante a detecção da face afetaram negativamente o resultado, e ocorrem de duas formas. O primeiro tipo ocorre por conta do detector de face não ter nenhuma predição para a imagem, acarretando em nenhuma orelha ser marcada. Quando isso ocorre, descartamos a imagem e seguimos para a próxima, porém atribuímos o valor de IOU igual a 0% para esses casos. 12 imagens não tiveram uma face detectada. O segundo tipo ocorre quando o detector encontra uma face e nós detectamos a sua orelha, porém a marcação da base não corresponde à face detectada pelo nosso método. Este último caso é mostrado na Figura 3.4(f). Um valor de IOU igual a 0% também é atribuído nesses casos, mesmo que o detector tenha realizado bem a sua tarefa. Erros desse tipo não foram contabilizados, porém ocorreram também em pequena quantidade e não achamos necessário tratá-los durante essa etapa do desenvolvimento. Em posse desse resultado, consideramos que podíamos dar prosseguimento para uma versão melhorada do detector.

3.3 VERSÃO INTERMEDIÁRIA - DETECÇÃO DE UMA ORELHA NUMA IMAGEM

Durante esta versão buscamos remover a necessidade da localização da face no nosso detector de orelhas. Continuamos a utilizar imagens 128×128 como entrada, porém não mais imagens de faces recortadas, e sim a imagem original redimensionada. A rede retorna como saída a região da orelha no formato especificado na versão anterior e também um

Figura 3.4 Resultado da versão inicial do detector para algumas imagens. A primeira fileira contém casos de sucesso acompanhado do valor de IOU. A segunda fileira contém alguns casos onde o detector falha. A região verde corresponde à marcação da base, e a vermelha à saída do detector.

novo valor, que chamaremos de valor de confiança. Este valor representa a probabilidade da região detectada pela rede ser realmente uma orelha. O objetivo dessa versão foi produzir um detector que realize bem a sua tarefa onde existe a restrição de uma orelha por imagem.

3.3.1 Arquitetura da Rede e Treinamento

Assim como na versão anterior, a rede recebe como entrada imagens 128×128, porém a saída agora consiste de 5 valores no formato $[x, y, w, h, c]$, os 4 primeiros sendo os valores já conhecidos anteriormente, e o último o valor de confiança. Este novo valor também está contido no intervalo $[0, 1]$. A arquitetura da rede conta com algumas diferenças em relação à da versão anterior e pode ser vista na Tabela 3.2. Em relação ao treinamento, o tamanho do *batch* permaneceu como 16, e o *Mean Squared Error* continuou sendo utilizado como *loss*. A adição do valor de confiança causou mudanças na forma como calculávamos o *loss*. Primeiramente, necessitamos de um valor que represente o valor de confiança esperado. Escolhemos utilizar o IOU entre a saída da rede a cada passo durante o treino e as marcações originais da orelha. Dessa forma, podemos ensinar a rede

a dar probabilidades altas para boas detecções e probabilidades baixas caso não consiga detectar uma orelha muito bem. Portanto, sejam $[\hat{x}, \hat{y}, \hat{w}, \hat{h}, \hat{c}]$ os valores de saída da rede, $[x, y, w, h]$ os valores esperados, e N o tamanho do *batch*, definimos o valor de confiança esperado c e novo *loss* $\mathcal{L}$ através das Equações 3.7 e 3.8.

$$c_i = IOU([x_i, y_i, w_i, h_i], [\hat{x_i}, \hat{y_i}, \hat{w_i}, \hat{h_i}]), \forall i, 1 \leq i \leq N \tag{3.7}$$

$$\mathcal{L} = \frac{\sum_{i=1}^{N}(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2 + (w_i - \hat{w_i})^2 + (h_i - \hat{h_i})^2}{N} + \alpha \frac{\sum_{i=1}^{N}(c_i - \hat{c_i})^2}{N} \tag{3.8}$$

Um parâmetro α é utilizado para regular a importância do valor de confiança no *loss*. Ele é necessário para balancear a importância de detecções corretas e valores de confiança adequados. Sem esse controle, a rede tende a encontrar uma solução onde o valor de confiança é sempre zero e ela erra totalmente as detecções. Em nossos experimentos, atribuimos o valor 0.01 a esse parâmetro.

Tabela 3.2 Arquitetura da Versão Intermediária da Rede

	Entrada	Filtros	*Strides*	Saída
Convolução + *ReLU*	128x128x3	5x5x3x32	1	128x128x32
Max Pooling	128x128x32	2x2	2	64x64x32
Convolução + *ReLU*	64x64x32	3x3x32x64	1	64x64x64
Max Pooling	64x64x64	2x2	2	32x32x64
$3x$(Convolução + *ReLU*)	32x32x64	3x3x64x128	1	32x32x128
Max Pooling	32x32x128	2x2	2	16x16x128
$3x$(Convolução + *ReLU*)	16x16x128	3x3x128x256	1	16x16x256
Max Pooling	16x16x256	2x2	2	8x8x256
$5x$(Convolução + *ReLU*)	8x8x256	2x2x256x512	1	8x8x512
Max Pooling	8x8x512	2x2	2	4x4x512
Flattening	4x4x512	—	—	8192
Fully Connected + ReLU	8192	—	—	1024
Fully Connected + ReLU	1024	—	—	2048
Dropout 10%	2048	—	—	2048
Fully Connected + Sigmoid	2048	—	—	5

3.3.2 Resultados e Discussão

Para realizar a inferência na base de teste, passamos para a rede as imagens redimensionadas para 128×128, dessa vez sem recorte de face. Da mesma forma que na fase anterior, calculamos apenas o IOU entre a detecção e a marcação original. Foram descartadas todas as detecções com valor de confiança abaixo de 0.3, mais precisamente 98 detecções foram descartadas. Porém, contabilizamos o IOU delas como 0% para que o resultado seja comparável ao nosso resultado anterior. Esta versão do detector alcançou 58% de IOU médio. Como podemos ver, houve uma queda do resultado da versão inicial para

a intermediária, que pode ser explicada pela natureza da nossa base de dados. Como nosso detector possui capacidade de encontrar apenas uma orelha na imagem, seu funcionamento não pode ser garantido na presença de mais de uma orelha. Casos onde mais de uma orelha estão presentes na imagem são comuns, e não impactaram negativamente a versão anterior tanto quanto essa por conta do recorte da região da face. A Figura 3.5 contém alguns resultados de nossa detecção, incluindo sucessos e falhas. Utilizando o aprendizado obtido durante ambas as fases inicial e intermediária, focamos em desenvolver a versão final, com o objetivo de eliminar as limitações destas versões e obter melhores resultados. Também identificamos a necessidade de complementar as marcações da nossa base de dados.

(a) 96.26% (b) 71.02% (c) 58.70%

(d) Múltiplas orelhas (e) Múltiplas orelhas (f) Baixo valor de confiança

Figura 3.5 Resultado da versão intermediária do detector, contendo casos de sucesso e falha na detecção. Para os casos de sucesso o IOU está presente. A região verde corresponde à marcação da base, e a vermelha à saída do detector.

Capítulo

4

MÉTODO PROPOSTO - DETECÇÃO DE MÚLTIPLAS ORELHAS POR IMAGEM

4.1 MARCAÇÕES NA BASE DE DADOS

No capítulo anterior foi citado um dos problemas na base de dados para a nossa tarefa: existe apenas uma marcação de orelha por imagem, e muitas imagens contém mais de uma orelha. Nas versões anteriores não foi dada muita importância a isso, visto que estávamos detectando apenas uma única orelha, e estávamos apenas interessados em saber se nosso detector era funcional. Como agora temos como objetivo detectar todas as orelhas da imagem, precisamos das marcações de todas as orelhas presentes. Para tanto, marcamos manualmente as orelhas restantes nas bases de treino, validação e teste. Durante o processo, tomou-se o cuidado de marcá-las de forma que fossem tão fieis quanto as marcações já existentes. A Figura 4.1 mostra um exemplo desse procedimento. Com este problema resolvido, seguimos com o aprimoramento do detector proposto.

4.2 DESCRIÇÃO GERAL DO MÉTODO

Na versão intermediária, como queríamos detectar no máximo uma orelha por imagem, nossa rede atribuía um valor de probabilidade indicando a presença ou não de uma orelha, e caso esse valor fosse maior que um certo limiar, utilizávamos a região encontrada como nossa detecção. Para que seja possível detectar múltiplas orelhas, podemos dividir a nossa imagem em diversas partes e agir de forma similar em cada uma delas independentemente.

No entanto, realizar essa divisão explicitamente acarretaria em um maior custo computacional, visto que precisaríamos realizar um método similar ao da versão anterior para cada uma das partes. Outro problema dessa estratégia é a possibilidade de uma única orelha estar localizada na fronteira de duas ou mais regiões, impossibilitando ou ao menos tornando mais difícil a detecção. Decidimos então realizar essa divisão implicitamente e, assim como foi feito na YOLO, dividimos a nossa imagem em seções. Porém, atribuímos apenas um preditor por seção. Dividimos a tarefa de detectar orelhas entre todos os preditores, ficando responsáveis pela orelha cujo centro está contido em sua seção. O centro

Figura 4.1 Alguns exemplos do processo de marcação em uma imagem da base de treino. A primeira linha contém as imagens com a marcação original. A segunda contém as imagens após a marcação manual.

da orelha é definido como sendo o ponto central da sua *bounding box*. A justificativa para a escolha de apenas um preditor por seção tem relação com a tarefa que deve ser realizada pelo detector. A YOLO deve lidar com uma grande quantidade de objetos de diferentes classes. Ao detectar diversos objetos, a chance de uma única seção ser responsável pela detecção de mais de um deles aumenta. Estes objetos também possuem diferentes tamanhos e proporções, o que torna cada um de seus preditores, durante a etapa de treino, mais especializados em certos tipos de objetos do que em outros. No nosso caso estamos apenas interessados em detectar orelhas, e a probabilidade de que múltiplos objetos apareçam em uma única seção é muito inferior se comparado à YOLO, além de todas elas possuírem proporções similares, variando apenas em escala.

A Figura 4.2 apresenta o *pipeline* do nosso método de detecção. É possível notar pelo diagrama que o nosso detector possui mais diferenças em relação à YOLO. O nosso classificador e o preditor são treinados separadamente e são independentes. Portanto, possuímos uma rede para realizar a predição da suposta localização da orelha em cada seção da forma descrita acima, e uma outra que classifica cada seção da imagem como um centro de orelha ou não. Note que a tarefa do classificador é atuar da mesma forma que o nosso valor de confiança da versão intermediária, sendo apenas treinado de outra forma e desacoplado do preditor. A nossa abordagem compartilha semelhanças com o

método de detecção da *Faster* R-CNN por utilizar um propositor e um classificador para realizar a tarefa. Contudo, a *Faster* R-CNN classifica cada região proposta, enquanto o nosso classificador nos diz apenas se devemos utilizar ou não a região proposta pelo preditor dessa mesma seção, sem nenhum conhecimento da região que o preditor propôs.

Durante o processo de treino e teste do nosso detector, utilizamos imagens de tamanho 160×160, portanto todas as imagens foram inicialmente redimensionadas. O tamanho foi alterado com o objetivo de lidar com orelhas pequenas em uma resolução um pouco maior, porém com poucas perdas no tempo de execução em relação às versões anteriores. Cada imagem possui 100 detectores e classificadores, sendo 10 por linha e 10 por coluna. Isso significa que cada detector e classificador é responsável por uma seção de tamanho 16×16 da imagem. Utilizaremos estes valores como constantes em cálculos futuros, ao invés de representar o tamanho da imagem e das seções por variáveis.

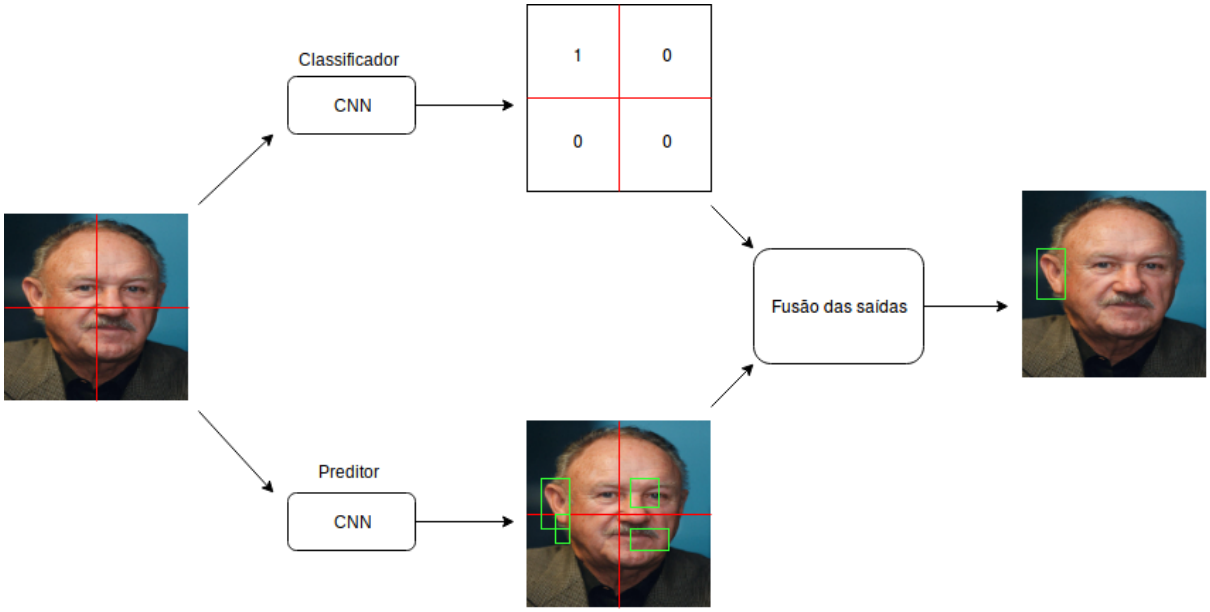

Figura 4.2 Diagrama ilustrativo do *pipeline* do nosso detector de orelhas. O preditor de cada seção encontra a região mais provável de ser uma orelha, enquanto o classificador de cada seção determina com base na presença ou não do centro de uma orelha se a predição daquela seção deverá ser usada. A saída das redes é fundida e as detecções realizadas são retornadas.

4.3 CLASSIFICADOR: ARQUITETURA E TREINAMENTO

A arquitetura da rede é composta por poucas camadas convolucionais e de *pooling*, seguidas de camadas *fully connected*. A rede recebe imagens de tamanho 160×160 e retorna 100 valores entre [0,1] para cada imagem, que indica a probabilidade de existir um centro de orelha em cada uma das seções. Assim como na versão intermediária, chamaremos estas probabilidades de valores de confiança. A Tabela 4.1 contém mais detalhes da arquitetura da rede.

Tabela 4.1 Arquitetura do Classificador

	Entrada	Filtros	*Strides*	Saída
Convolução + *ReLU*	160x160x3	5x5x3x32	1	160x160x32
Max Pooling	160x160x32	2x2	2	80x80x32
Convolução + *ReLU*	80x80x32	3x3x32x64	1	80x80x64
Max Pooling	80x80x64	2x2	2	40x40x64
Convolução + *ReLU*	40x40x64	3x3x64x128	1	40x40x128
Max Pooling	40x40x128	2x2	2	20x20x128
Convolução + *ReLU*	20x20x128	3x3x128x256	1	20x20x256
Max Pooling	20x20x256	2x2	2	10x10x256
Convolução + *ReLU*	10x10x256	3x3x256x512	1	10x10x512
Max Pooling	10x10x512	2x2	2	5x5x512
Flattening	5x5x512	—	—	12800
Fully Connected + ReLU	12800	—	—	512
Dropout 10%	512	—	—	512
Fully Connected + Sigmoid	512	—	—	100
Reshape	100	—	—	10x10

Durante o treinamento o tamanho do *batch* utilizado foi $N = 8$. É comum ver a utilização da função $softmax$ para o *loss* em tarefas de classificação, porém não a utilizamos. A razão para isso está no desbalanceamento entre as classes que a rede deve aprender a diferenciar. Como cada imagem possui 100 seções e no mínimo uma orelha, isso significa que, no pior caso, apenas uma seção é classificada como centro de orelha. Para lidar com o desbalanceamento, utilizamos o *focal loss* (Lin *et al.*, 2017), que atribui pesos dinamicamente para cada uma das seções, dando maior ênfase às seções que não estão sendo classificadas corretamente. Logo, seja N o tamanho do *batch*, $focal\_loss(a, b)$ a função que calcula a contribuição de cada saída a partir do valor real e do valor de saída, $\hat{c_{k,i,j}}$ e $c_{k,i,j}$ a saída da rede e o valor real da seção (i, j) da imagem k, o *loss* $\mathcal{L}$ é definido pelas Equações 4.1 e 4.2.

$$focal\_loss(\hat{p}, p) = \begin{cases} -(1-\hat{p})^{\gamma}\log(\hat{p}) & \text{se } p = 1 \\ -(1-(1-\hat{p}))^{\gamma}\log(1-\hat{p}) & \text{caso contrário} \end{cases} \tag{4.1}$$

$$\mathcal{L} = \frac{\sum_{k=1}^{N}\sum_{i=1}^{10}\sum_{j=1}^{10} focal\_loss(\hat{c_{k,i,j}}, c_{k,i,j})}{N} \tag{4.2}$$

O parâmetro γ visto na definição do *focal loss* é utilizado para regular o peso de cada saída no *loss*. Escolhemos o valor $\gamma = 1$ para os nossos experimentos.

A rede foi treinada por 6000 épocas, utilizando o otimizador *Adam* (Kingma e Ba, 2015), com *learning rate* de 0.0001, e valores de $\beta_1 = 0.9$ e $\beta_2 = 0.999$. Os dois últimos são parâmetros de decaimento de momento do otimizador, e utilizamos os valores padrão recomendados. Selecionamos, no final do treino, o modelo com maior acurácia no conjunto de validação. Uma época corresponde ao uso de todas as imagens de treino uma

única vez. Ao final de uma época, embaralhamos a lista de imagens para termos novas combinações de *batches*. As imagens utilizadas para o treino do classificador, e também do preditor, passam por diversas transformações com o objetivo de melhorar a generalização de ambas as redes. Esse processo será explicado na Seção 4.5. Calculamos a acurácia na validação utilizando a fórmula $\frac{VP}{VP+FP+FN}$, onde VP corresponde à quantidade de seções corretamente classificadas, FP à quantidade de seções classificadas como verdadeiro mas que deveriam ser falsas e FN à quantidade de seções classificadas como falso mas que deveriam ser verdadeiras. Os valores de probabilidades da saída da rede maiores ou iguais a 0.5 eram considerados como verdadeiro, enquanto os menores eram considerados como falso. Este limiar foi utilizado somente durante a validação do modelo durante o treino. Escolhemos este valor por ser difícil determinar o melhor limiar durante o treinamento da rede. A escolha do melhor limiar foi realizada durante a experimentação e será mostrado mais a frente. A métrica utilizada possui a mesma propriedade do IOU, punindo as duas formas de classificação incorreta e julgamos ser uma boa métrica para a escolha do modelo final.

4.4 PREDITOR: ARQUITETURA E TREINAMENTO

O nosso preditor, por sua vez, compartilha algumas semelhanças com a arquitetura da versão intermediária. Ele recebe, assim como o classificador, imagens de tamanho 160×160 e retorna, para cada uma delas, 100 predições no formato $[x, y, w, h]$, definido anteriormente, uma por seção, indicando a região onde o preditor acredita que exista uma orelha cujo centro está contido em sua seção. Mostramos a arquitetura na Tabela 4.2.

Tabela 4.2 Arquitetura do Preditor

	Entrada	Filtros	*Strides*	Saída
Convolução + *ReLU*	160x160x3	3x3x3x32	1	160x160x32
Max Pooling	160x160x32	2x2	2	80x80x32
$2x$(Convolução + *ReLU*)	80x80x32	3x3x32x64	1	80x80x64
Max Pooling	80x80x64	2x2	2	40x40x64
$3x$(Convolução + *ReLU*)	40x40x64	3x3x64x128	1	40x40x128
Max Pooling	40x40x128	2x2	2	20x20x128
$3x$(Convolução + *ReLU*)	20x20x128	3x3x128x256	1	20x20x256
Max Pooling	20x20x256	2x2	2	10x10x256
$5x$(Convolução + *ReLU*)	10x10x256	3x3x256x512	1	10x10x512
Max Pooling	10x10x512	2x2	2	5x5x512
Flattening	5x5x512	—	—	12800
Fully Connected + ReLU	12800	—	—	1024
Dropout 10%	1024	—	—	1024
Fully Connected + Sigmoid	1024	—	—	400
Reshape	400	—	—	10x10x4

Utilizamos $N = 8$ novamente como o tamanho do *batch*, e o cálculo do *loss* é derivado da fórmula usada na versão intermediária. Sejam $[\hat{x_{k,i,j}}, \hat{y_{k,i,j}}, \hat{w_{k,i,j}}, \hat{h_{k,i,j}}]$ e

$[x_{k,i,j}, y_{k,i,j}, w_{k,i,j}h_{k,i,j}]$ os valores de saída da rede e os valores esperados para a seção (i, j) da imagem k, e N o tamanho do *batch*, o *loss* $\mathcal{L}$ é definido pelas equações 4.3, 4.4 e 4.5.

$$coord\_loss = \frac{\sum_{k=1}^{N}\sum_{i=1}^{10}\sum_{j=1}^{10}(x_{k,i,j} - \hat{x_{k,i,j}})^2 + (y_{k,i,j} - \hat{y_{k,i,j}})^2}{N} \tag{4.3}$$

$$dimension\_loss = \frac{\sum_{k=1}^{N}\sum_{i=1}^{10}\sum_{j=1}^{10}(\sqrt{w_{k,i,j}} - \sqrt{\hat{w_{k,i,j}}})^2 + (\sqrt{h_{k,i,j}} - \sqrt{\hat{h_{k,i,j}}})^2}{N} \tag{4.4}$$

$$\mathcal{L} = coord\_loss + dimension\_loss \tag{4.5}$$

Separamos em duas partes apenas para melhor visualização. Somamos a contribuição de cada seção no *loss* e dividimos pelo número de imagens do *batch*. A diferença aqui está no *dimension_loss*. O artifício foi retirado da função de *loss* da YOLO, que utiliza a raiz quadrada para aumentar a contribuição dos erros em detecções de regiões menores, e ela está presente aqui com o mesmo objetivo.

Treinamos o preditor por 8000 épocas, utilizando o mesmo otimizador e parâmetros do classificador. Também selecionamos no final o modelo com maior acurácia no conjunto de validação. Para esse cálculo, precisamos levar em consideração que, por mais que treinemos cada preditor para encontrar uma orelha em sua seção, essa restrição é suscetível a falhas durante a inferência, e eles podem encontrar orelhas que deveriam ser encontradas por seções vizinhas com maior precisão. Dito isso, o que fazemos aqui é selecionar a predição com maior IOU com a marcação da orelha. Fazemos essa associação de forma unívoca, ou seja, cada predição é selecionada por apenas uma marcação.

4.5 GERAÇÃO DE NOVAS IMAGENS

Para reduzir a chance de *overfitting* e simular um aumento à quantidade de imagens de treino, utilizamos técnicas de transformação de imagens, conhecidas também pelo nome de *data augmentation*. Estas técnicas foram aplicadas durante o treino, ou seja, durante o processo de seleção das imagens que compõem o *batch* a ser utilizado a cada iteração. Realizando o *augmentation* dessa forma, sempre que colocarmos novamente uma das imagens do treino no *batch* a ser utilizado, ela será modificada de diferentes formas. Listaremos todas as técnicas aplicadas abaixo. Todas essas técnicas foram aplicadas em conjunto, mas podem ser vistas sendo aplicadas tanto em conjunto quanto separadamente na Figura 4.5 para uma melhor compreensão. A biblioteca *imgaug*[1] foi utilizada para realizar as transformações.

4.5.1 Espelhamento

Um problema não mencionado anteriormente que tivemos com a nossa base tem relação com a quantidade de orelhas esquerdas e direitas. A Coleção A, que dividimos em treino e validação, é composta quase por completo por orelhas direitas, enquanto a Coleção B

[1] https://imgaug.readthedocs.io/en/latest/

é composta por um maior número de orelhas esquerdas. Para resolver esse problema, criamos uma cópia das imagens de treino e de validação espelhadas, transformando as orelhas direitas em orelhas esquerdas, e vice-versa. Essa é a única transformação que foi aplicada antes do treinamento da rede, e que também foi aplicada no conjunto de validação, a fim de verificarmos se o nosso detector estava aprendendo a detectar ambas as orelhas.

4.5.2 Transformações Lineares: Escala, Rotação e Translação

Transformações lineares são bastante utilizadas como uma técnica de *data augmentation*. As transformações foram aplicadas escolhendo um valor aleatório dentro do intervalo especificado para cada imagem em tempo de execução. Os intervalos de valores utilizados em cada transformação podem ser vistos na Tabela 4.3.

Tabela 4.3 Tabela de valores usados para geração de novas imagens usando transformações lineares

Transformação	Intervalo de Valores
Translação no eixo x	[-48, +48]
Translação no eixo y	[-48, +48]
Rotação	[-45°, +45°]
Escala no eixo x	[75%, 125%]
Escala no eixo y	[75%, 125%]

4.5.3 Variações de Cor: Escurecimento, Clareamento e Tom de Cinza

Como queremos que o nosso detector consiga encontrar orelhas em diversos ambientes, aplicamos variação de iluminação às imagens, multiplicando cada canal de cor por um valor aleatório no intervalo $[0.5, 1.5]$. Os valores são truncados em 255, maior valor possível para um canal de cor. Além disso, aproximamos cada imagem ao tom de cinza por um fator δ aleatório entre $[0, 1]$. Quando $\delta = 0$ a imagem não sofria alterações, quando $\delta = 1$ a imagem era convertida completamente para o tom de cinza. Um exemplo visual da aplicação do acinzamento utilizando diferentes valores de δ pode ser visto na Figura 4.3

(a) 0 (b) 0.33 (c) 0.66 (d) 1

Figura 4.3 Ilustração do processo de acinzamento atribuindo diferentes valores para o δ.

4.5.4 Aumentando o Tamanho das Orelhas: Recorte de imagem

Para que o nosso detector seja robusto a escala, ele deve ser treinado com orelhas de diferentes tamanhos. A transformação anterior de escala apenas oferece pequena variação no tamanho das orelhas nas imagens. Para obter uma maior variação, recorremos ao procedimento utilizado na versão inicial para o recorte de faces utiizando a região da orelha. Porém, ao invés de recortar a face, recortamos aleatoriamente um retângulo contido na região da face, cujo a região da orelha estivesse contida nele. Essa região recortada é redimensionada para 160×160 no final do procedimento. A Figura 4.4 mostra visualmente o procedimento.

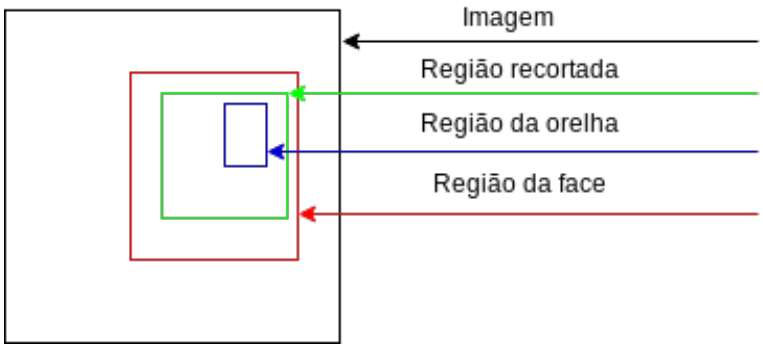

Figura 4.4 Processo de recorte na imagem para geração de novas imagems. A imagem acima ilustra como é feito o procedimento explicado. A região de corte escolhida sempre contém a região da orelha e está contida na região da face.

4.5.5 Aumentando a Quantidade de Orelhas: Colagem de Imagens

Por último, precisávamos aumentar a quantidade de orelhas por imagem durante o treino. Poucas imagens da base de treino possuem mais de uma orelha, e isso poderia impactar negativamente o desempenho do nosso detector. Para alcançar um número maior de orelhas, realizamos colagem de imagens. O procedimento foi realizado da seguinte forma: Sorteamos um valor K entre 1 e 4. Pegamos K imagens, redimensionamos para 80×80 e criamos uma imagem preta de tamanho 160×160. Essa imagem preta pode ser vista como a união de 4 blocos de tamanho 80×80. Colocamos cada uma das K imagens em um desses blocos, deixando vazios os que não foram selecionados. Dessa forma, conseguimos criar imagens artificiais com mais de uma orelha presente. Realizamos essa colagem sorteando valores pequenos pois em alguns casos as orelhas presentes nas imagens após o redimensionamento tornariam-se pequenas demais para serem vistas.

Figura 4.5 Técnicas de geração de imagens utilizadas. Apresentamos cada uma delas separadamente e mostramos o resultado da aplicação de todas em conjunto.

Capítulo 5

EXPERIMENTAÇÃO E RESULTADOS

5.1 MÉTRICAS DE AVALIAÇÃO

Antes de discutirmos os resultados, apresentaremos as métricas utilizadas para verificar a performance do nosso método. Como pôde ser visto no Capítulo 2, cada trabalho teve seus resultados reportados utilizando diferentes métricas. Sem um padrão para seguir, calculamos o resultado do nosso método por diferentes perspectivas, para que seja possível realizar comparações com alguns dos métodos mais recentes citados anteriormente.

5.1.1 Métricas a nível de pixel

Calculamos as mesmas métricas utilizadas por Emeršič *et al.* (2018) para que fosse possível comparar nossos resultados com os obtidos por ele. Este trabalho reporta também resultados obtidos pelo método de Abaza, Hebert e Harrison (2010), implementado pelos autores para realizar comparações com seu próprio método. Portanto, estas métricas serão utilizadas para comparar nossos resultados com outros dois trabalhos. Elas são: acurácia, IOU, precisão e *recall*, definidas abaixo:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \tag{5.1}$$

$$IOU = \frac{VP}{VP + FP + FN} \tag{5.2}$$

$$Precisão = \frac{VP}{VP + FP} \tag{5.3}$$

$$Recall = \frac{VP}{VP + FN} \tag{5.4}$$

As notações VP, VN, FP e FN são oriundas de seus nomes: verdadeiro positivo, verdadeiro negativo, falso positivo e falso negativo, respectivamente. VP corresponde à quantidade de *pixels* corretamente classificados como verdadeiro, VN à quantidade de

pixels corretamente classificados como falso. FP indica a quantidade de *pixels* erroneamente classificados como verdadeiros, enquanto FN nos diz a quantidade erroneamente classificada como falso. *Pixels* da orelha são por definição verdadeiros, enquanto todos os outros são considerados falsos. Contudo, o nosso método não realiza a detecção de orelhas da mesma forma que Emeršič *et al.* (2018). Para que fosse possível calcular estes valores, consideramos os *pixels* dentro das *bounding boxes* de orelha da base como verdadeiros e todos os outros como falsos. Em relação ao nosso detector, interpretamos os *pixels* da mesma forma, sendo os das *bounding boxes* detectadas verdadeiros, e todos os outros *pixels* da imagem fora dessas regiões falsos. Dessa forma, temos uma conversão entre a análise do nosso método de detecção para a análise do método de detecção de Emeršič *et al.* (2018), possibilitando o cálculo das métricas apresentadas acima. A Figura 5.1 mostra visualmente o que cada métrica apresentada acima representa.

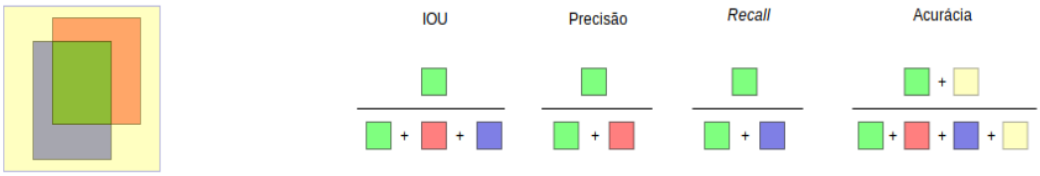

Figura 5.1 Representação visual das métricas apresentadas. Podemos definir o retângulo amarelo como nossa imagem de entrada e o azul como a região a ser detectada. O retângulo em vermelho representa a nossa predição, enquanto a região verde é a área de interseção entre a predição e a marcação original.

5.1.2 Métricas Relacionadas à Detecção

Para medir a performance do nosso método, calculamos a taxa de falsa descoberta, denominado por FDR (*false discovery rate*), e a taxa de falso negativo, denominado por FNR (*false negative rate*). O primeiro valor nos diz a porcentagem de detecções realizadas pelo nosso algoritmo que não correspondem a orelhas. O segundo valor se refere à porcentagem de orelhas que não foram detectadas pelo nosso método. Eles são calculados da seguinte forma:

$$FDR = \frac{FP}{VP + FP} \tag{5.5}$$

$$FNR = \frac{FN}{VP + FN} \tag{5.6}$$

Aqui, VP corresponde à quantidade de orelhas corretamente detectadas, FP ao número de orelhas erroneamente detectadas e FN à quantidade de orelhas que não foram encontradas pelo nosso detector. Uma orelha é corretamente detectada se a IOU entre a região marcada e a região proposta é maior ou igual a um certo limiar definido. Juntas,

essas duas métricas são adequadas para mostrar a eficácia do nosso método na detecção. Um bom detector deve conseguir detectar o maior número de orelhas possível, ao mesmo tempo que detecta o menor número possível de falsos positivos. Podemos extrair os valores de precisão e *recall* baseado no critério definido acima através das equações 5.7 e 5.8.

$$Precisão = 1 - FDR \tag{5.7}$$
$$Recall = 1 - FNR \tag{5.8}$$

5.2 NON-MAXIMAL SUPPRESSION

Durante a discussão sobre o preditor da rede no Capítulo 4, nos deparamos com um problema e levamos ele em consideração durante o treinamento da rede: a detecção da orelha de uma determinada seção por outra. Nosso classificador também é suscetível a essa falha. Com variações em escala e pose, o centro de uma região de orelha pode ter diferentes características. Assim, mais de uma seção do classificador pode afirmar possuir o centro da mesma orelha. Isto nos leva, portanto, a receber diversas detecções para uma única orelha. Para eliminá-las, utilizamos um algoritmo simples chamado de *Non-Maximal Suppression*. Pegamos todas as predições de saída e as ordenamos de forma decrescente utilizando o valor de confiança, com predições de mesmo valor podendo estar permutadas de qualquer forma, sem critérios de desempate. Para cada região dada nessa ordem, adicionamos ela em uma lista contendo as predições que serão retornadas. Em seguida, descartamos todas as regiões ainda não selecionadas com IOU maior ou igual a um limiar de supressão. Repetimos esse processo até não termos mais nenhuma região a ser processada. Este método é bastante conhecido e utilizado também por alguns trabalhos citados nos capítulos anteriores. Utilizamos um limiar de supressão de 0.25 (ou 25%) em nossos experimentos. A Figura 5.2 ilustra o algoritmo descrito acima. Pode ser visto na Figura 5.3 o processo de detecção do nosso método, que contém um exemplo da aplicação da técnica mostrada nessa seção.

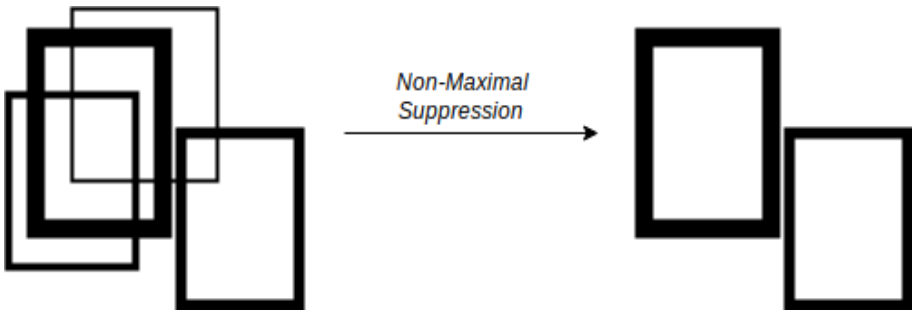

Figura 5.2 Ilustração do procedimento realizado pelo *Non-Maximal Suppression*. O valor de confiança é representado pela espessura da caixa retangular. Quanto mais espessa, maior é o valor de confiança.

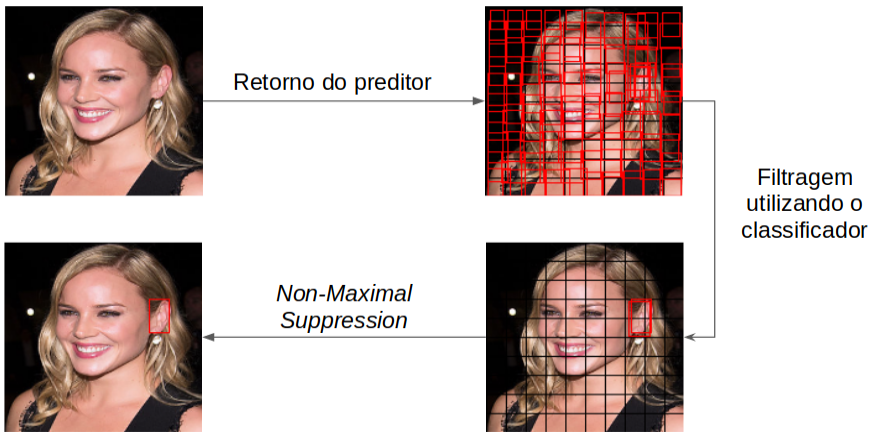

Figura 5.3 Ilustração do processo de detecção do método em uma das imagens da base de teste. É possível ver as predições para cada uma das seções e o resultado após a filtragem utilizando o valor de confiança do classificador. Por fim, para eliminar as múltiplas detecções, é utilizado o *Non-Maximal Suppression*.

5.3 RESULTADOS OBTIDOS

Como foi explicado no Capítulo 4, utilizamos os valores de confiança da saída do classificador para decidir se utilizamos ou não as detecções dadas pelo preditor. Utilizando um determinado limiar de confiança, descartamos todas as predições com valor de confiança abaixo deste limiar. Durante a experimentação, utilizamos diferentes limiares, a fim de encontrar o melhor para o nosso detector.

A rede foi treinada em uma máquina com dois processadores Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz e uma placa gráfica Nvidia GeForce GTX Titan Xp com 12 GB de memória. A inferência foi realizada em outra máquina, com um processador Dual Core Intel(R) Core(TM) i7-5500U CPU @ 2.40GHz com 8GB de memória RAM e uma placa gráfica Nvidia GeForce 830M com 2GB de memória. A inferência leva em média 132 milissegundos, considerando o processamento de ambas as redes, o *Non-Maximal Suppression* e a marcação das predições na imagem. O detector foi implementado na linguagem *python 3*, com o auxílio das bibliotecas *Tensorflow*[1], *NumPy*[2] e *Matplotlib*[3], além das outras bibliotecas citadas no decorrer do texto.

Apresentamos primeiramente nossos resultados relacionados à detecção. Para isso, foram geradas curvas de $FDR \times FNR$ variando limiares de confiança começando em 0 e terminando em 1, utilizando um passo de 0.01. Cada curva utiliza um limiar de IOU distinto para decidir se uma detecção foi correta. O resultado pode ser visto na Figura 5.4. A partir dessas curvas, pudemos extrair o limiar de confiança que nos leva ao melhor

[1]https://www.tensorflow.org/
[2]https://www.numpy.org/
[3]https://matplotlib.org/

resultado do detector. Definimos que o melhor ponto da curva é aquele onde a soma do FNR e FDR são minimizados, visto que ambos os tipos de erros são indesejados. Este ponto foi calculado através dos resultados numéricos e foi definido sendo 0.42. Os valores para este ponto da curva podem ser vistos na Tabela 5.1.

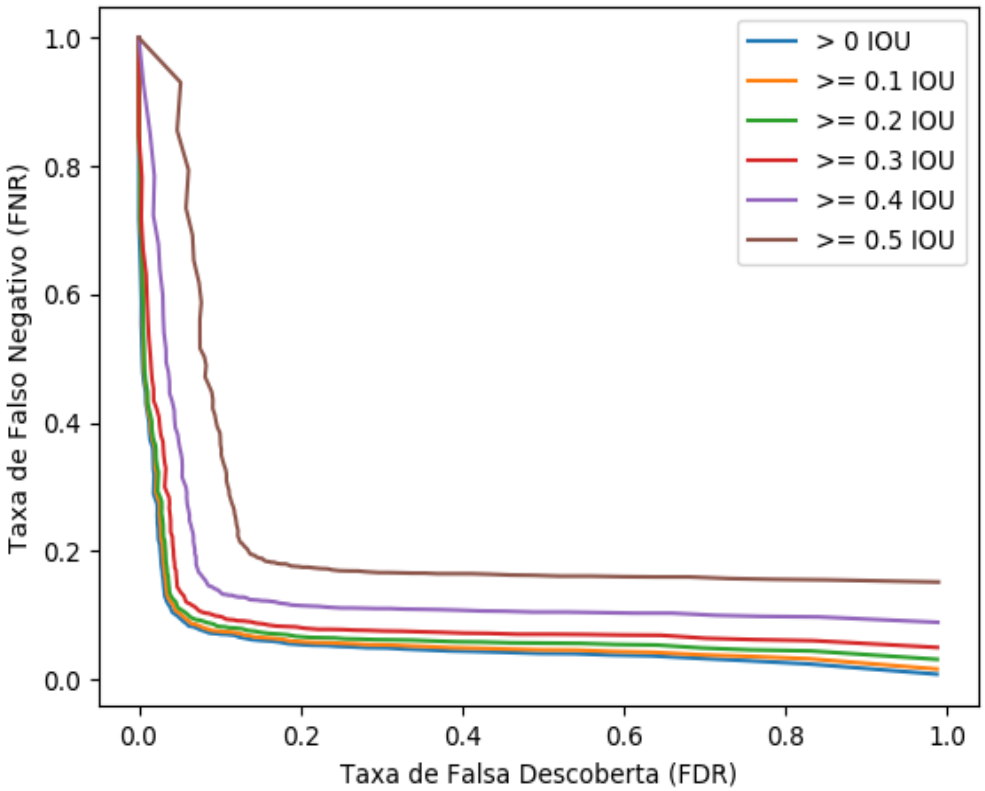

Figura 5.4 Curva $FDR \times FNR$ variando os valores do limiar de confiança para diferentes limiares de IOU. A legenda no canto superior direito identifica cada uma das curvas em relação ao limiar de IOU utilizado.

Tabela 5.1 Valores numéricos do FDR e FNR para o limiar de confiança de 0.42

Limiar de IOU	FDR	FNR
> 0	0.0532	0.0927
≥ 0.1	0.0567	0.0960
≥ 0.2	0.0611	0.1003
≥ 0.3	0.0716	0.1104
≥ 0.4	0.0998	0.1373
≥ 0.5	0.1530	0.1883

A partir dos dados da tabela, observamos que 5.32% de todas as predições feitas não possuem sobreposição com nenhuma orelha marcada. Podemos visualizar todas as outras predições como tentativas de detectar orelhas, porém com diferentes limiares de IOU. À medida que elevamos este limiar, algumas detecções positivas se tornam falsos positivos, como pode ser visto na tabela. Levando em consideração qualquer limiar para o IOU, 9.27% das orelhas da base não foram detectadas. Porém, não há muito interesse nesse valor, visto que queremos um detector preciso. Considerando um limiar de 50% para o IOU, o nosso detector é capaz de encontrar 81.17% das orelhas da base de teste. Este é um limiar adequado para decidir se uma detecção é correta, e é geralmente utilizado para validação de trabalhos envolvendo detecção de objetos, como no caso da YOLO. Além disso, deve-se levar em consideração que, em geral, orelhas ocupam uma pequena região da imagem, e deslizes de poucos *pixels* na detecção resultam em quedas drásticas no valor do IOU. Tomando como exemplo uma orelha marcada com região retangular de tamanho 18×12 e uma predição cuja região retangular tivesse o mesmo tamanho da marcação, porém estivesse transladada um *pixel* para a esquerda e um para baixo. O IOU entre a região marcada e a predição seria de apenas 76.32%, porém teria uma sobreposição de 86.57%. Portanto, em casos de objetos pequenos, a métrica mostra ser mais punitiva, e podemos ser um pouco mais flexíveis ao analisar um detector que trabalha, na maior parte do tempo, com esses casos. Utilizando um limiar de 40% para o IOU, nosso método é capaz de detectar 86.27% das orelhas da base de teste.

Seguimos agora para a apresentação dos resultados a nível de *pixel*. Os valores foram calculados utilizando o limiar de confiança de 0.42, definido a partir dos resultados anteriores. Para cada imagem, calculamos as métricas apresentadas no início deste capítulo, e em seguida tiramos a média dos valores entre todas as imagens. Nosso método alcança um IOU de 66.34%, precisão de 81.26%, *recall* de 76.67% e acurácia de 99.49%. A acurácia, porém, não é uma métrica adequada para a avaliação de performance quando existe desbalanceamento entre as classes. No nosso caso, por exemplo, se uma imagem 160×160 possuir apenas uma orelha de tamanho 16×24 e classificarmos toda a imagem como não possuindo *pixels* pertencentes à orelhas, nossa acurácia seria de 98.50%. Contudo, estes valores, incluindo a acurácia, foram calculados apenas para fins comparativos.

As Figuras 5.5 e 5.6 apresentam alguns resultados de detecção, acompanhado dos valores de IOU a nível de *pixel* para cada imagem. Na Figura 5.5 podemos ver resultados de sucesso, tanto em casos com uma única orelha quanto onde existem duas ou mais. Nosso detector ainda é capaz de encontrar orelhas quando não há uma face inteiramente visível, como é mostrado nas Figuras 5.5(g), 5.5(h) e 5.5(i). As Figuras 5.5(e) e 5.5(f) mostram exemplos de sucesso na detecção de orelhas pequenas. Contudo, ainda existem falhas, que podem ser vistas nas Figuras 5.6(h) e 5.6(i). É possível observar que nenhuma predição é realizada para muitas delas, e é uma das causas para o grande número de falsos negativos. O grande número de falsos positivos está relacionado à detecção de outros pontos da face e regiões próximas às mãos (exemplos são vistos nas Figuras 5.6(a), 5.6(b), 5.6(c), 5.6(d) e 5.6(g)). Casos onde partes do fundo da imagem são confundidos com uma orelha são mais incomuns, porém ainda ocorrem, como pode ser visto na Figura 5.6(f).

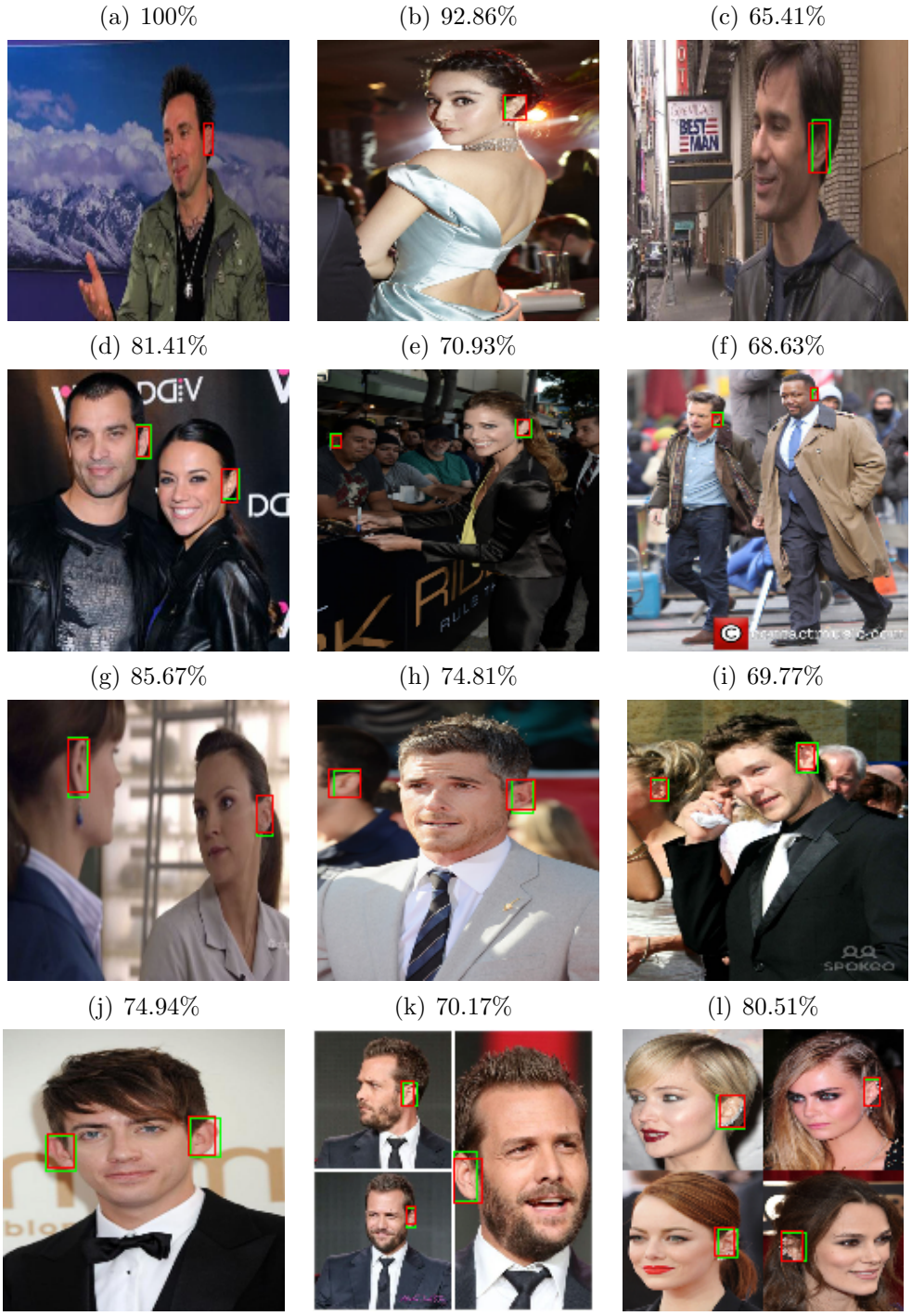

Figura 5.5 Resultado de algumas detecções bem sucedidas da versão final do método. Cada imagem é acompanhada do seu valor de IOU a nível de *pixel*. As regiões em verde correspondem à marcação da base, enquanto as vermelhas à saída do detector.

5.4 COMPARAÇÃO COM OUTROS MÉTODOS

Utilizando os valores calculados a nível de *pixel* realizamos uma comparação com o método de Emeršič *et al.* (2018). O tempo de inferência deste método é em média 87.5 milis-

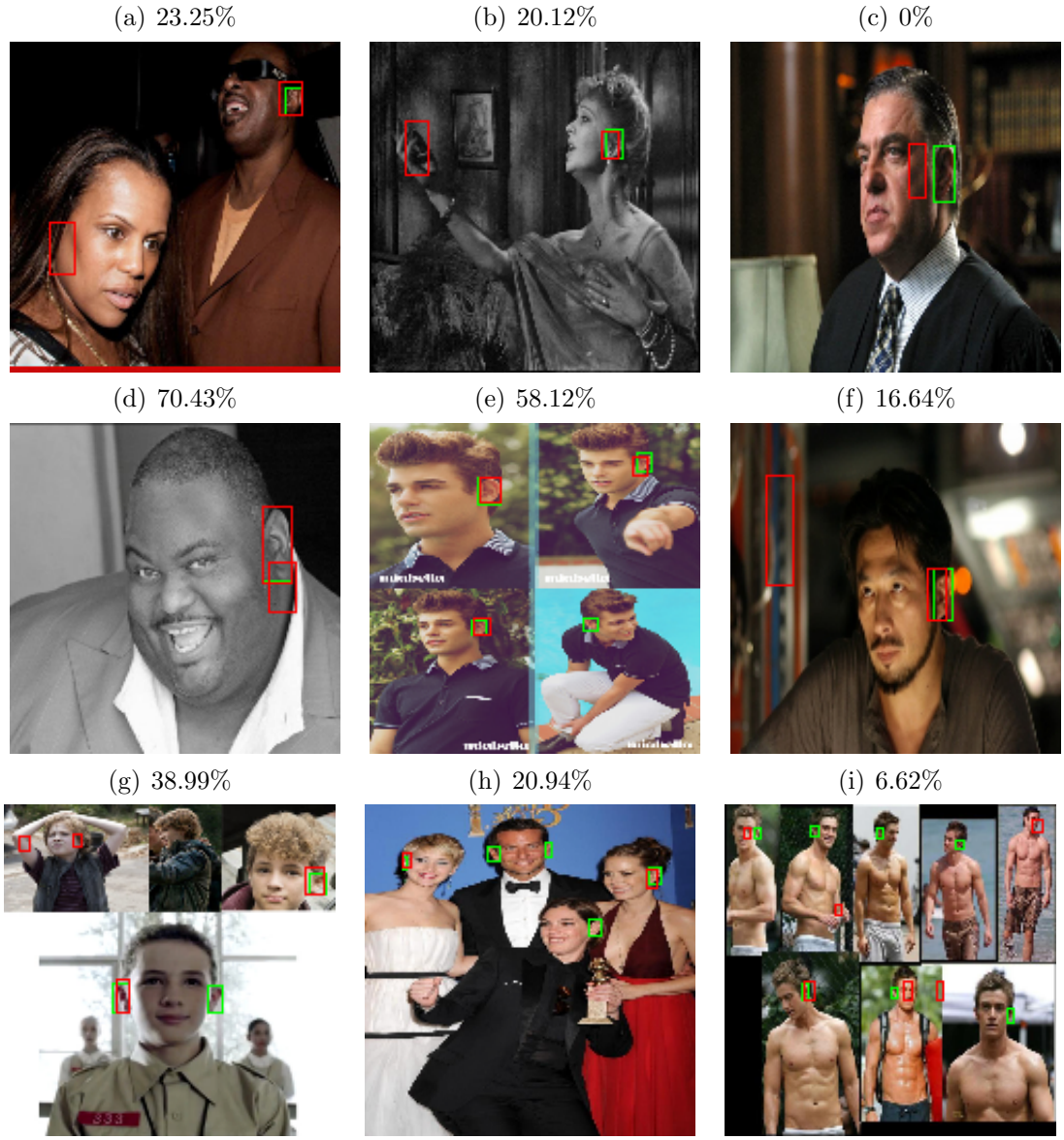

Figura 5.6 Resultado de algumas detecções mal sucedidas da versão final do método no mesmo formato apresentado na figura 5.5.

segundos por imagem em um Intel(R) Core(TM) i7-6700K CPU com 32GB de RAM e uma placa gráfica Nvidia GeForce GTX 980 Ti com 6GB de memória. Comparado ao tempo do nosso método, ele é inferior, porém é utilizada uma máquina com maior poder computacional. Os resultados podem ser vistos na Tabela 5.2. Os valores mostrados correspondem à média entre todas as imagens da base, seguido do desvio-padrão. Também é comparado o método de Abaza, Hebert e Harrison (2010), implementado e testado por Emeršič *et al.* (2018) na mesma base utilizada em seu trabalho, como foi dito anteriormente.

Podemos observar que nossos resultados são superiores em relação a todas as métricas

calculadas, principalmente o IOU e a precisão. Deve também ser levado em consideração o uso de diferentes bases de dados, e por isso uma comparação direta não pode ser feita. O método de Abaza, Hebert e Harrison (2010), por outro lado, pode ser comparado diretamente ao método de Emeršič *et al.* (2018), e tem uma performance inferior. Contudo, o método de Emeršič *et al.* (2018) assume a presença de apenas uma face na imagem, detectando apenas as orelhas presentes nela. Nosso método, no entanto, não impõe restrições em relação à quantidade e nem à presença de faces, além de ter sido testado em 2058 imagens. Portanto, a partir desta comparação, podemos ver que o nosso detector supera os resultados alcançados por estes dois métodos. Algumas imagens de resultados, extraídas de ambos os trabalhos, podem ser vistas na Figura 5.7. Os valores mostrados se referem à média e o desvio-padrão para cada uma das métricas.

Tabela 5.2 Comparativo entre o método proposto e os métodos de Emeršič *et al.* (2018) e Abaza, Hebert e Harrison (2010).

Método	Base de dados(Nº Imagens)	IOU[%]	Acurácia[%]	Precisão[%]	*Recall*[%]
Abaza, Hebert e Harrison (2010)	AWE(250)	27.23 ± 36.47	98.76 ± 1.13	36.67 ± 46.60	28.51 ± 38.40
Emeršič *et al.* (2018)	AWE(250)	48.31 ± 23.01	99.21 ± 0.58	60.83 ± 25.97	75.86 ± 33.11
Método Proposto	*In-the-wild Ear*(2058)	66.34 ± 18.82	99.49 ± 0.49	81.26 ± 19.45	76.67 ± 18.12

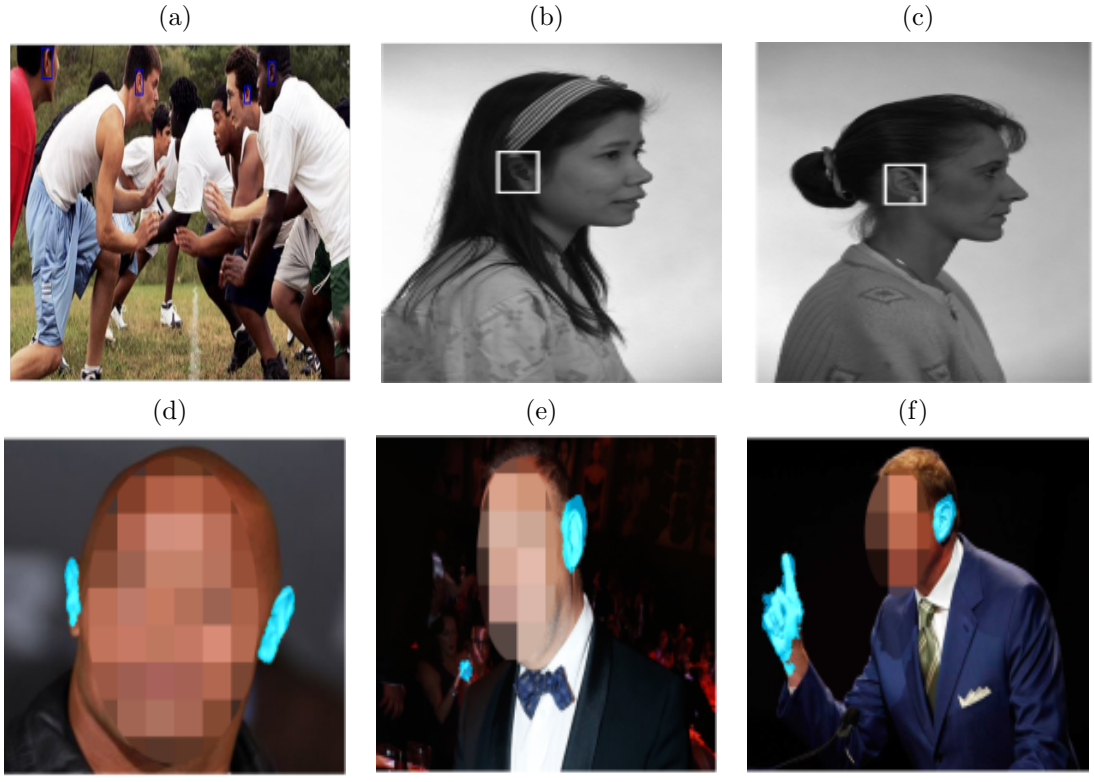

Figura 5.7 Algumas imagens publicadas dos métodos de Abaza, Hebert e Harrison (2010) na primeira linha e de Emeršič *et al.* (2018) na segunda.

Alguns resultados do método de Wang, Du e Huang (2017) podem ser vistos na Figura

5.8. O tempo de inferência não foi mostrado neste trabalho. Ele reporta a acurácia, precisão e *recall* em relação ao número de subimagens processadas pelo seu método. O nosso método não permite essa avaliação, visto que não realizamos a classificação de subimagens como orelha ou não-orelha. Dito isso, os resultados mostrados na Tabela 5.3 para o nosso método e para o método de Wang, Du e Huang (2017) não são diretamente comparáveis. Mostramos, então, os valores de precisão e *recall* a nível de detecção para o nosso método, utilizando os limiares de IOU de 40% e 50%. Estes valores foram extraídos da forma explicada no início do capítulo.

O trabalho de Wang, Du e Huang (2017) não revela de forma clara como é realizada a distinção entre uma detecção correta ou não, visto que uma subimagem classificada como orelha pode não conter a orelha como um todo, como mostra a Figura 5.8(a) com maior clareza. Com o limiar de 50% para o IOU, conseguimos resultados próximos, porém não melhores que os atingidos por Wang, Du e Huang (2017). Porém, a partir de um limiar de 40%, nossos resultados conseguem superá-lo. Entretanto, como foi dito anteriormente, não é possível realizar conclusões baseado nesse comparativo, visto que foram utilizadas formas diferentes para o cálculo das métricas. Contudo, podemos ver que eles possuem uma performance similar. A métrica da acurácia aqui possui os mesmos problemas de quando é usada a nível de *pixel*. Existem mais subregiões da imagem que não correspondem à orelhas do que o contrário, tornando o valor alto mesmo quando nenhuma orelha é detectada. Da mesma forma que na análise anterior, a capacidade de nosso método de detectar orelhas de mais de uma pessoa na imagem é um ponto positivo a favor do nosso método. O trabalho de Wang, Du e Huang (2017) não faz menção e não reporta resultados em imagens com essa característica, deixando incerta a performance do seu método nessas situações.

Tabela 5.3 Comparativo entre o método proposto e o método de Wang, Du e Huang (2017).

Método	Base de dados(Nº Imagens)	Acurácia[%]	Precisão[%]	*Recall*[%]
Wang, Du e Huang (2017)	AWE(200)	98.19	85.97	83.25
Método Proposto (IOU > 50%)	*In-the-wild Ear*(2058)	—	84.70	81.17
Método Proposto (IOU > 40%)	*In-the-wild Ear*(2058)	—	90.02	86.27

Seguimos agora para o método de Zhang e Mu (2017). Algumas imagens presentes no trabalho são mostradas na Figura 5.9. O tempo de inferência em uma imagem gira em torno de 220 milissegundos em uma máquina com uma placa gráfica Nvidia GeForce GTX Titan X. Este é um tempo razoável, porém quase duas vezes maior do que o nosso método testado em uma máquina menos potente. Outras informações, como o processador e a memória, não foram especificadas. Foram reportados os valores de acurácia, precisão, *recall* e *F1 Score*. Este último não será utilizado, pois é definido através dos valores de precisão e *recall*. Novamente, não é informado com clareza como esses valores são calculados, principalmente a respeito da decisão sobre uma detecção ser correta ou não. O método foi avaliado em três diferentes bases, e os resultados obtidos em cada uma delas pode ser visto na Tabela 5.4. Na Tabela 5.5 utilizamos a média dos valores das três bases para a comparação com o nosso método com um limiar de IOU de 50%.

Observamos que os resultados reportados por esse método são significativamente me-

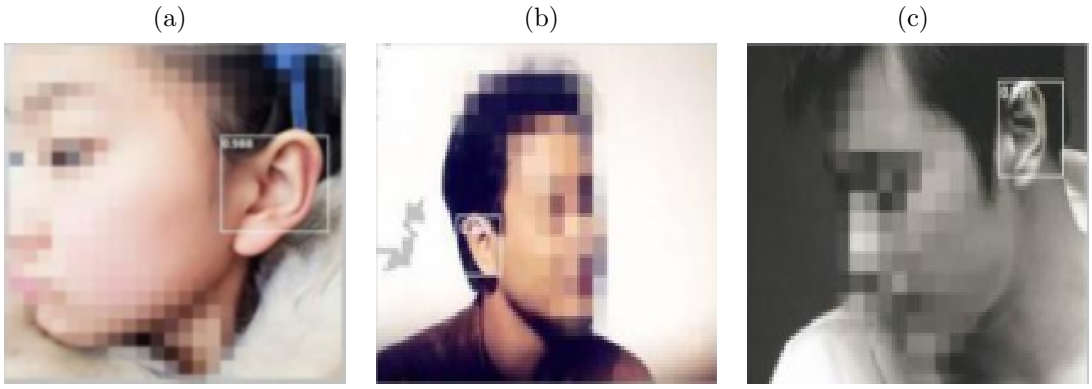

Figura 5.8 Algumas imagens publicadas do método de Wang, Du e Huang (2017).

lhores. Porém, também não é possível realizar uma comparação direta, visto que não sabemos como as métricas foram calculadas com exatidão. Uma desvantagem do método de Zhang e Mu (2017) é a necessidade da detecção em múltiplos níveis, detectando a região da face, uma região menor contendo a orelha e a região da orelha. O nosso método é capaz de detectar orelhas sem o auxílio de nenhuma outra informação, como por exemplo na ausência da face, mostrado anteriormente em exemplos da Figura 5.5. Outro fator é a ausência de resultados para imagens com mais de uma orelha presente. Isso pode indicar que o método é funcional apenas quando há uma única orelha na imagem, ao contrário do nosso método.

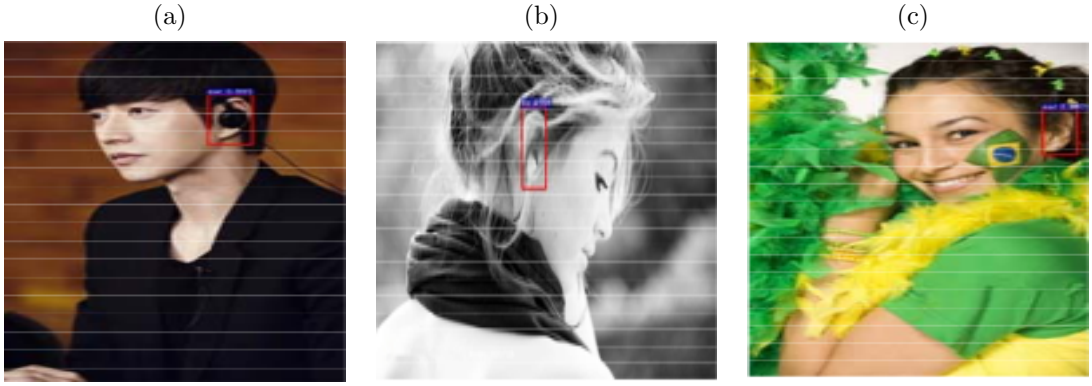

Figura 5.9 Algumas imagens publicadas do método de Zhang e Mu (2017).

Tabela 5.4 Resultados do método de Zhang e Mu (2017) para diferentes bases.

Base de dados(Nº Imagens)	Acurácia[%]	Precisão[%]	*Recall*[%]
WebEar(200)	98	99.49	98.50
UND-J2(1800)	100	100	100
UBEAR(9121)	98.66	99.22	98.66

Tabela 5.5 Comparativo entre o método proposto e o método de Zhang e Mu (2017).

Método	Base de dados(Nº Imagens)	Acurácia[%]	Precisão[%]	*Recall*[%]
Zhang e Mu (2017)	Diversas(11121)	98.88	99.57	99.05
Método Proposto (IOU > 50%)	*In-the-wild Ear*(2058)	—	84.70	81.17

Por fim, temos o método de Ganapathi *et al.* (2018), cujo alguns resultados podem ser vistos na Figura 5.10. Eles reportam apenas a acurácia do método em duas diferentes bases. Sua acurácia é definida com base na porcentagem de orelhas detectadas corretamente. Eles definem um limiar de 80% de IOU para diferenciar uma detecção bem sucedida de uma mal sucedida. A Tabela 5.6 mostra o resultado do método de Ganapathi *et al.* (2018) nas duas bases utilizando um limiar de 80% de IOU, e o nosso resultado utilizando o limiar de 50%. Com este limiar para o nosso método, nossos resultados, realizando o mesmo cálculo, já são bastante inferiores. Conseguimos detectar 1927 orelhas de 2374 no total, resultando em 81.17% de acurácia. Este cálculo, entretanto, desconsidera a presença de falsos positivos. É citado no início do trabalho de Ganapathi *et al.* (2018) que seu método não detecta nenhum falso positivo, enquanto nosso método possui uma taxa de falsa descoberta de 15.30% com um limiar de 50% para o IOU. Contudo, Ganapathi *et al.* (2018) publicam algumas imagens mostrando a detecção de falsos positivos e falsos negativos, porém não é especificado de onde essas imagens foram retiradas. Esses resultados podem ser vistos nas Figuras 5.10(d), 5.10(e) e 5.10(f).

Tabela 5.6 Resultados do nosso método e o método de Ganapathi *et al.* (2018) para suas diferentes bases.

Método	Base de dados(Nº Imagens)	Acurácia[%]
Ganapathi *et al.* (2018)	IIT Indore Collection-A(4162)	98.20
Ganapathi *et al.* (2018)	AWE(600)	99.52
Método Proposto (IOU > 50%)	*In-the-wild Ear*(2058)	81.17

Em relação à qualidade das detecções de ambos os métodos, façamos uma análise qualitativa dos resultados mostrados nas Figuras 5.10(a), 5.10(b) e 5.10(c), e de alguns de nossos resultados, incluindo as nossas versões anteriores, vistos nas Figuras 5.5(c), 3.5(b), 3.5(c) e 3.4(c). Podemos observar que nossas detecções com IOU entre 58% e 70% envolvem a orelha de uma forma mais precisa que as detecções consideradas corretas pelo método de Ganapathi *et al.* (2018), que afirma possuírem IOU maior ou igual a 80%. É também citado no trabalho que as marcações em ambas as bases testadas foram feitas à mão. Entretanto, nenhum exemplo dessas marcações foi mostrado e apenas as predições foram marcadas nos exemplos do resultado. Considerando a complexidade computacional, o método de Ganapathi *et al.* (2018) leva 2.1 segundos para realizar a inferência em uma máquina com Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz com 12GB de memória RAM e uma placa gráfica Nvidia GeForce GTX 1070 com 8GB de memória. Este tempo é aproximadamente 17 vezes superior ao tempo de inferência do nosso método em uma máquina muito menos poderosa e não pode ser usado em tempo real.

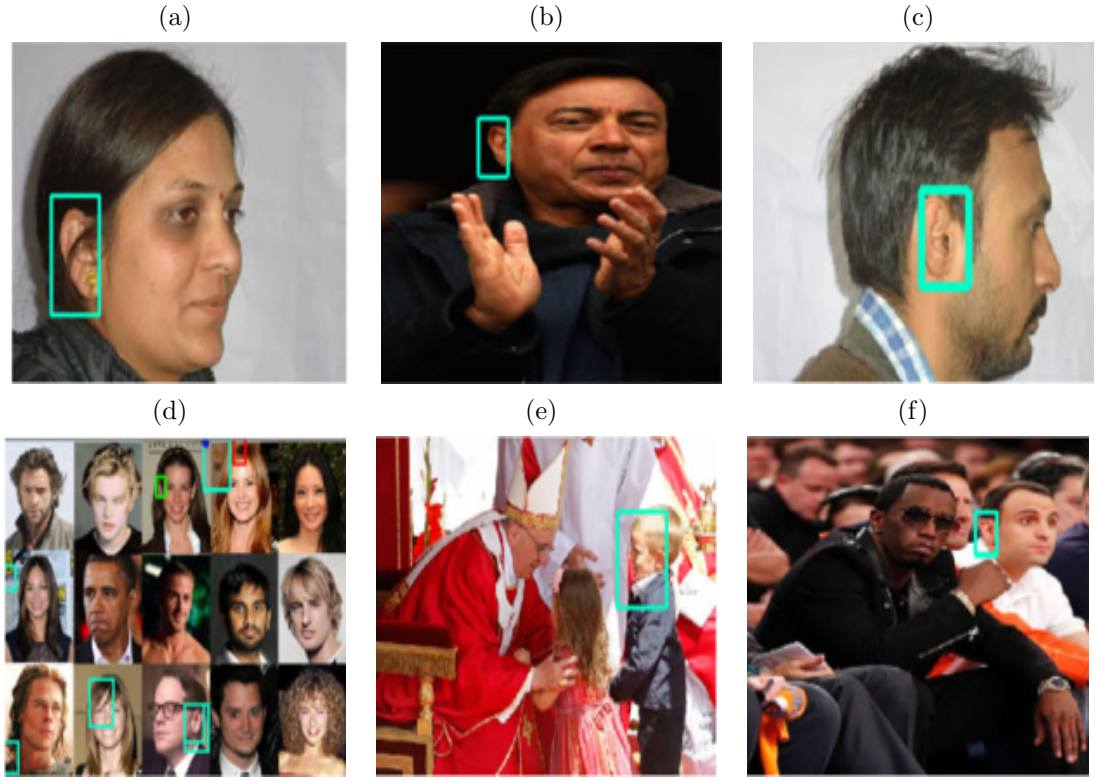

Figura 5.10 Algumas imagens publicadas do método de Ganapathi *et al.* (2018). Casos de detecções bem sucedidas são vistos na primeira linha. Alguns casos de detecção de falsos positivos e negativos do método são mostrados na segunda. Em relação a esta última, não foi especificado de onde as imagens testadas foram retiradas.

Capítulo 6

CONCLUSÃO

Os resultados obtidos pelo nosso método ainda são insatisfatórios para uso em tarefas reais. Porém, nosso método consegue melhores resultados em relação a outros métodos recentes em qualidade de detecção, e alguns em tempo de execução. Este método supera algumas barreiras existentes em outros trabalhos anteriores, como limitações no número de faces e orelhas por imagem e a necessidade de informações da face para a detecção.

As maiores limitações do nosso método são a falha na detecção de orelhas muito pequenas e a detecção de falsos positivos em regiões do corpo humano, como nariz e mãos. Além disso, a forma como o detector é modelado atualmente permite que o preditor encontre uma orelha, porém o classificador a rejeite, mesmo que seja uma detecção correta. Isso resulta em falsos negativos, e é possível que sejam evitados caso o detector seja remodelado para que o classificador e o preditor compartilhem informações. Isso pode ser feito enviando para o classificador as saídas do preditor, transformando-o em um classificador de orelhas.

Existem ainda outras brechas para evolução. Para melhorar os resultados do detector, pode ser construído e utilizado um refinador. A partir da saída do nosso método, este refinador poderia encontrar a orelha presente numa área mais limitada da imagem, próxima à predição. O uso de âncoras, caixas retangulares de tamanhos fixos, pode nos dar um melhor resultado comparado à realizar uma regressão para encontrar a região da orelha, que é como nosso preditor atua. São comumente utilizadas em métodos de detecção de objetos, como o YOLO e *Faster R-CNN*, e pode beneficiar também o nosso detector. Como orelhas em geral possuem proporções similares, poucas âncoras seriam necessárias, o que poderia melhorar o nosso desempenho. Dessa forma, utilizaríamos o preditor apenas dentro da região selecionada pela âncora, da mesma forma feita pelo *Faster R-CNN*.

O compartilhamento de camadas da rede entre o preditor e o classificador pode reduzir ainda mais o tempo de inferência de nossa rede. Isso nos possibilita trabalhar com imagens maiores, atenuando o problema da falha de detecção de orelhas muito pequenas. A união do classificador e do preditor em uma única rede também não é uma possibilidade

descartada. Através da escolha de parâmetros e modelamento da rede adequados, pode ser possível chegar aos resultados atuais ou até melhores com um ganho no tempo de execução.

Existem muitos caminhos que podem ser seguidos daqui em diante para alcançar melhores resultados. Este trabalho apresenta o método proposto com um grande nível de detalhamento sobre cada etapa do processo de construção. Isso auxilia para que outros interessados também participem e proponham métodos mais eficientes para a localização da orelha, despertando um interesse maior pelo uso dessa biometria.

REFERÊNCIAS BIBLIOGRÁFICAS

Abaza, A.; Hebert, C.; Harrison, M. A. F. Fast learning ear detection for real-time surveillance. In: *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. [S.l.: s.n.], 2010. p. 1–6.

Chen, H.; Bhanu, B. Human ear detection from side face range images. In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* [S.l.: s.n.], 2004. v. 3, p. 574–577 Vol.3. ISSN 1051-4651.

Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. [S.l.: s.n.], 2005. v. 1, p. 886–893 vol. 1. ISSN 1063-6919.

Emeršič, Z.; Struc, V.; Peer, P. Ear recognition: More than a survey. *CoRR*, abs/1611.06203, 2016. Disponível em: ⟨http://arxiv.org/abs/1611.06203⟩.

Emeršič, *et al.* Convolutional encoder–decoder networks for pixel-wise ear detection and segmentation. *IET Biometrics*, v. 7, n. 3, p. 175–184, 2018. ISSN 2047-4938.

Fan, T.; Mu, Z.; Yang, R. Multi-modality recognition of human face and ear based on deep learning. In: *2017 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*. [S.l.: s.n.], 2017. p. 38–42. ISSN 2158-5709.

Ganapathi, I. I. *et al.* Unconstrained ear detection using ensemble-based convolutional neural network model. *Concurrency and Computation: Practice and Experience*, p. e5197, 2018. E5197 cpe.5197. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5197⟩.

Gao, W. *et al.* The cas-peal large-scale chinese face database and baseline evaluations. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, v. 38, n. 1, p. 149–161, Jan 2008. ISSN 1083-4427.

Islam, S. M. S.; Bennamoun, M.; Davies, R. Fast and fully automatic ear detection using cascaded adaboost. In: *2008 IEEE Workshop on Applications of Computer Vision*. [S.l.: s.n.], 2008. p. 1–6. ISSN 1550-5790.

Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Lin, T. *et al.* Focal loss for dense object detection. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017. p. 2999–3007. ISSN 2380-7504.

Liu, L. *et al.* Deep learning for generic object detection: A survey. *CoRR*, abs/1809.02165, 2018. Disponível em: ⟨http://arxiv.org/abs/1809.02165⟩.

Pflug, A.; Winterstein, A.; Busch, C. Robust localization of ears by feature level fusion and context information. In: *2013 International Conference on Biometrics (ICB)*. [S.l.: s.n.], 2013. p. 1–8. ISSN 2376-4201.

Raposo, R. *et al.* Ubear: A dataset of ear images captured on-the-move in uncontrolled conditions. *2011 IEEE Workshop on Computational Intelligence in Biometrics and Identity Management (CIBIM)*, p. 84–90, 2011.

Redmon, J. *et al.* You only look once: Unified, real-time object detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 779–788. ISSN 1063-6919.

Redmon, J.; Farhadi, A. Yolo9000: Better, faster, stronger. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2017. p. 6517–6525. ISSN 1063-6919.

Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. Disponível em: ⟨http://arxiv.org/abs/1804.02767⟩.

Ren, S. *et al.* Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 39, n. 6, p. 1137–1149, June 2017. ISSN 0162-8828.

Saxe, J.; Berlin, K. Deep neural network based malware detection using two dimensional binary program features. In: *2015 10th International Conference on Malicious and Unwanted Software (MALWARE)*. [S.l.: s.n.], 2015. p. 11–20.

Tiwari, S. *et al.* Ear recognition for newborns. In: *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*. [S.l.: s.n.], 2015. p. 1989–1994.

Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.* [S.l.: s.n.], 2001. v. 1, p. I–I. ISSN 1063-6919.

Wang, M.; Deng, W. Deep face recognition: A survey. *CoRR*, abs/1804.06655, 2018. Disponível em: ⟨http://arxiv.org/abs/1804.06655⟩.

Wang, S.; Du, Y.; Huang, Z. Ear detection using fully convolutional networks. In: *Proceedings of the 2Nd International Conference on Robotics, Control and Automation*. New York, NY, USA: ACM, 2017. (ICRCA '17), p. 50–55. ISBN 978-1-4503-5327-4. Disponível em: ⟨http://doi.acm.org/10.1145/3141166.3141168⟩.

Yuan, L.; Mu, Z. Ear detection based on skin-color and contour information. In: *2007 International Conference on Machine Learning and Cybernetics*. [S.l.: s.n.], 2007. v. 4, p. 2213–2217. ISSN 2160-133X.

Yuan, L.; Zhang, F. Ear detection based on improved adaboost algorithm. In: *2009 International Conference on Machine Learning and Cybernetics*. [S.l.: s.n.], 2009. v. 4, p. 2414–2417. ISSN 2160-133X.

Zhang, Y.; Mu, Z. Ear detection under uncontrolled conditions with multiple scale faster region-based convolutional neural networks. *Symmetry*, v. 9, 04 2017.