



UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO

Edge FR - Desenvolvimento de um sistema de reconhecimento facial para dispositivos embarcados de baixo custo

Trabalho de Conclusão de Curso

Leandro de Jesus Dias Santana



São Cristóvão – Sergipe

2024

UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO

Leandro de Jesus Dias Santana

Edge FR - Desenvolvimento de um sistema de reconhecimento facial para dispositivos embarcados de baixo custo

Trabalho de Conclusão de Curso submetido ao Departamento de Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador(a): MSc. Rafael Andrade da Silva
Coorientador(a): Prof. Dr. Leonardo Nogueira Matos

São Cristóvão – Sergipe

2024

Este trabalho é dedicado aos amigos que criei nesses anos, aos professores que me passaram tanto conhecimento e à minha família que sempre esteve lá por mim quando precisei.

Resumo

Nos últimos anos as redes neurais convolucionais têm sido cada vez mais utilizadas para solução de diversos tipos de desafios, e entre eles está o reconhecimento facial, que após tantos avanços já consegue se comparar e, possivelmente até superar a habilidade humana nessa tarefa. Porém, deve-se atentar que possui uma desvantagem, que é a sua grande complexidade computacional para modelos maiores, o que limita sua aplicação em dispositivos de borda mais próximos da coleta dos dados. Essa grande complexidade acaba por necessitar de um alto poder de processamento e de quantidade de memória para a obtenção de bons resultados de predição. Este trabalho aborda o desafio de implementar sistemas de reconhecimento facial eficazes em dispositivos embarcados com recursos limitados na borda, mantendo ao máximo bons níveis de acurácia e latência. Para isso o modelo compacto de reconhecimento facial MobileFaceNet foi escolhido. Um *pipeline* e sistema foi construído a partir dele, com etapas incluindo detecção facial, extração de características e comparação par a par. Uma base de rostos foi construída para avaliar a acurácia do reconhecimento. A etapa final foi a de embarcar o modelo em um Orange Pi 3 LTS e comparar seu desempenho com uma Jetson Nano e um notebook i7. Para a Orange Pi 3 LTS, foi alcançado um resultado de 2,27 QPS para a tarefa de reconhecimento facial em tempo real. O MobileFaceNet obteve nas avaliações uma acurácia de 99.1%, *Recall* de 1.0 e *F1-Score* de 0.995. Isso demonstrou uma otimista capacidade de dispositivos embarcados na borda para realização de reconhecimento facial.

Palavras-chave: CNN, Detecção facial, Reconhecimento facial, Visão computacional, Computação na borda, Compressão, Sistemas embarcados

Abstract

In recent years, convolutional neural networks have been increasingly used to solve different types of challenges, and among them is facial recognition, which after so many advances can now compare to the human ability and sometimes even surpass it in the task. However, it should be noted that it has a disadvantage, which is its great computational complexity for larger models, which limits its application on edge devices closer to data collection. This great complexity ends up requiring a high processing power and amount of memory to obtain good prediction results. This work addresses the challenge of implementing effective facial recognition systems on embedded devices with limited resources at the edge, while maintaining good levels of accuracy and latency as much as possible. To achieve this, the face recognition compact model MobileFaceNet was chosen. A pipeline and system was created from it, with steps including face detection, feature extraction and pair to pair comparison. A face dataset was created to evaluate the accuracy of the recognition. The final step was to embed the model into an Orange Pi 3 LTS and compare its performance with a Jetson Nano and a i7 notebook. For the Orange Pi 3 LTS, was achieved a result of 2.27 FPS for the task of face recognition in real time. The MobileFaceNet achieved an accuracy of 99.1%, *Recall* of 1.0 and *F1-Score* of 0.995 on the tests. This demonstrated an optimistic ability of edge embedded devices to perform face recognition.

Keywords: CNN, Face detection, Face recognition, Computer vision, Edge computing, Compression, Embedded systems

Lista de ilustrações

Figura 1 – Modelo não-linear de um neurônio	17
Figura 2 – Diagrama do Perceptron Multicamadas	19
Figura 3 – Exemplo de um mapa de características que realiza extração de bordas	20
Figura 4 – Exemplo de uma camada convolucional	21
Figura 5 – Quantização ciente do treinamento e Quantização pós treinamento respectivamente	24
Figura 6 – Visualização das características HOG de uma imagem	26
Figura 7 – Placa Orange Pi 3 LTS	28
Figura 8 – Placa Face-RK3399	29
Figura 9 – NVIDIA Jetson Nano	30
Figura 10 – Placa SLN-VIZNAS-IOT	30
Figura 11 – Um típico bloco de bottleneck	35
Figura 12 – <i>Pipeline</i> de Reconhecimento Facial Utilizado	35
Figura 13 – Exemplo de Detecção Facial	36
Figura 14 – Experimento de Coleta de Faces Realizado no DComp-UFS	38
Figura 15 – Exemplo de Comparação Par a Par	39

Lista de tabelas

Tabela 1 – Resultados da Revisão Sistemática	31
Tabela 2 – Comparação Entre os Três Hardwares	40

Lista de abreviaturas e siglas

ABNT	Associação Brasileira de Normas Técnicas
DCOMP	Departamento de Computação
UFS	Universidade Federal de Sergipe
CNN	Convolutional Neural Network
LFW	Labeled Faces in the Wild
AI	Artificial Intelligence
NN	Neural Network
IoT	Internet of Things
HOG	Histogram of Gradients
QPS	Quadros Por Segundo

Lista de símbolos

ϕ Letra grega Fi

Σ Letra grega Sigma

Sumário

1	Introdução	11
1.1	Motivação	12
1.2	Objetivos	13
1.3	Metodologia	13
1.4	Estrutura do Documento	14
2	Fundamentação Teórica	16
2.1	Redes Neurais	16
2.1.1	Neurônios artificiais	16
2.1.2	Processo de Aprendizado	18
2.1.3	Camadas da rede neural	18
2.1.4	Rede Neural Perceptron Multicamadas	19
2.2	Redes Neurais Convolucionais	19
2.2.1	Estrutura e Funcionamento das CNNs	20
2.2.2	Métricas de Avaliação de uma CNN	21
2.2.3	Técnicas de diminuição do sobreajuste	22
2.2.3.1	Aumentação de Dados	22
2.2.3.2	Regularização	22
2.3	Métodos de Compressão de Redes Neurais	23
2.3.1	Quantização	23
2.3.2	<i>Pruning</i>	24
2.3.3	Destilação de Conhecimento Professor-Aluno	25
2.4	Detecção e Reconhecimento Facial	25
2.4.1	Alguns Métodos de Detecção Facial	26
2.4.2	Problemas Comuns na Detecção Facial	27
2.4.3	Reconhecimento Facial vs Verificação Facial	27
3	Trabalhos Relacionados	28
3.1	Soluções Comerciais	28
3.1.1	<i>Face-RK3399 Face Recognition Main Board</i>	29
3.1.2	<i>Nvidia Jetson Nano</i>	29
3.1.3	<i>NXP EdgeReady Board</i>	30
3.2	Trabalhos Acadêmicos	31
3.2.1	<i>A Resource Constrained Pipeline Approach to Embed Convolutional Neural Models (CNNs)</i>	32

3.2.2	<i>Deep Unified Model For Face Recognition Based on Convolution Neural Network and Edge Computing</i>	32
3.2.3	<i>DGFaceNet: Lightweight and efficient face recognition</i>	33
3.2.4	<i>Real-Time Multiple Face Recognition using Deep Learning on Embedded GPU System</i>	33
4	Desenvolvimento	34
4.1	Escolha do Modelo	34
4.2	Pipeline do Reconhecimento Facial	35
4.2.1	Detecção de Face	36
4.2.2	Extração de Características	36
4.2.3	Checagem de Pares com Distância do Cosseno	37
4.3	Experimento Prático e Criação do Banco de Rostos	37
4.4	Avaliação do modelo	38
4.5	Dados de Aplicação em Dispositivos Embarcados	39
5	Conclusão	41
	Referências	43

1

Introdução

As redes neurais estão agregando valor ao mercado e à sociedade. Elas são utilizadas em várias áreas, incluindo o reconhecimento facial, que vem ganhando cada vez mais popularidade nos últimos anos principalmente pelos avanços significativos na área. Outro motivo para esse aumento na popularidade é sua grande faixa de aplicações, dentre elas entretenimento, segurança da informação, vigilância e, claro, como forma de autenticação. Este último é um tópico relevante em visão computacional, reconhecimento de padrões e processamento de imagens ([BARNOUTI SINAN SAMEER MAHMOOD AL-DABBAGH, 2016](#)).

A percepção de rostos é uma tarefa feita com sucesso e sem muito esforço por humanos mas não é uma tarefa fácil para computadores. O sistema visual humano acomoda caminhos neurais complexos para processamento de características estáticas e dinâmicas de rostos para poder reconhecer rostos em relação ao conhecimento contextual ([TASKIRAN; KAHRAMAN; ERDEM, 2020](#)). O reconhecimento de faces pode ser aproximado como um problema de identificação ou problema de verificação. A identificação de faces é um problema de correspondência 1:N, no qual a face desconhecida é comparada com todas as faces de uma base de dados de identidades conhecidas e uma decisão é feita como resultado de todas as comparações. A identidade do rosto de consulta é confirmada ou rejeitada ao compará-la com os dados faciais da identidade alegada no banco de dados.

Um fator muito importante para o sucesso do reconhecimento facial está na capacidade de sistemas computacionais identificarem e verificarem rostos humanos com precisão e rapidez. Um dos pilares fundamentais por trás desse avanço é o uso de Redes Neurais Convolucionais (CNN), um tipo de modelo de aprendizado profundo eficaz na extração de características complexas de imagens.

Modelos de CNNs, como *DeepFace*, *VGGFace*, *FaceNet* e outros, têm alcançado altas taxas de acerto (>99% de acurácia) em tarefas de verificação e identificação facial ([TASKIRAN; KAHRAMAN; ERDEM, 2020](#)). Isso se deve, em parte, ao treinamento em grandes conjuntos

de dados e ao uso de técnicas de aprendizado profundo. Porém, com o avanço da tecnologia, surge também uma crescente preocupação com a privacidade e a ética, à medida que se buscam técnicas de reconhecimento facial que respeitem a privacidade dos indivíduos.

A implementação de sistemas de reconhecimento facial baseados em CNNs nem sempre é direta, especialmente quando se trata de dispositivos com recursos limitados, como dispositivos embarcados, que precisam de mais cuidado no seu projeto, incluindo a sua complexidade, que determina o seu tamanho e acurácia, que é de extrema importância ao se tratar de reconhecimento facial, pois um falso positivo pode causar desde um bloqueio no acesso de sua conta do banco a uma prisão errada.

Para um sistema de reconhecimento facial em dispositivos embarcados de baixo custo, pressupõem-se aplicações mais limitadas e específicas, como no caso de segurança para condomínios, onde se espera rostos isolados e bem iluminados, que reduzem a complexidade do problema e consequentemente o poder computacional necessário. Este trabalho se propõe como prova de conceito para uma aplicação semelhante à apresentada.

1.1 Motivação

O uso de Redes Neurais Convolucionais para o reconhecimento facial representa um avanço no campo da visão computacional. As CNNs têm demonstrado um alto desempenho e ainda crescente na extração de características de imagens, tornando-as uma escolha natural para sistemas de reconhecimento facial. No entanto, o desafio surge quando se trata de implementar essas redes em dispositivos embarcados, que muitas vezes possuem recursos limitados em termos de poder de processamento e memória. Existem vários métodos de compressão que podem ser utilizados para permitir que uma rede profunda possa ser reduzida o suficiente para ser utilizada em um desses dispositivos sem que haja uma grande perda de acurácia.

Muitas tentativas têm sido feitas para processar os dados coletados de dispositivos na borda em modelo de aprendizagem profunda. Porém esses dispositivos possuem recursos computacionais limitados e bateria (no caso de dispositivos móveis) para localmente executar modelos de aprendizado profundo de larga escala, essas tarefas normalmente são despachadas para uma plataforma computacional na nuvem (LEE; NA, 2022). O uso de dispositivos na borda com recursos limitados possibilita que o aparelho esteja próximo da coleta dos dados. Isso, por sua vez, reduz a latência na comunicação e oferece uma alternativa mais econômica em comparação com a utilização de servidores em nuvem. Essa abordagem adquire maior relevância ao considerar que esses dispositivos restritos geralmente possuem capacidade de memória e poder de processamento substancialmente inferiores em comparação com os dispositivos convencionais utilizados para resolver esse tipo de problema, resultando, consequentemente, em custos a longo prazo menores. Portanto, o desenvolvimento de um modelo que seja compatível com dispositivos com limitações de hardware pode ser uma solução ideal para atender a necessidades específicas.

Diante disso, embora as CNNs demonstrem um desempenho notável no reconhecimento de padrões em imagens, com acurárias de até 99.63% ao utilizar modelos como FaceNet, o ambiente de recursos limitados dos dispositivos de borda, devido a sua disponibilidade de energia limitada e pouca capacidade computacional, torna-os uma plataforma desafiadora para implantação da análise de dados desejada, especialmente em aplicações em tempo real (PLASTIRAS et al., 2018). Neste contexto, a compressão de modelos tem se destacado como uma abordagem promissora para permitir que redes grandes e bem treinadas sejam adaptadas a dispositivos com recursos limitados e na borda, preservando a acurácia. Além disso, a migração de tarefas de processamento de dados para dispositivos na borda, em contraste com o envio de dados para servidores em nuvem, oferece uma solução mais eficaz em termos de custos (ao longo prazo) e latência de comunicação. Esse trabalho destaca a importância de considerar dispositivos de borda como ambientes viáveis para a realização de reconhecimento facial e apresenta o potencial dessa abordagem para enfrentar desafios atuais da visão computacional.

1.2 Objetivos

Este trabalho tem como objetivo principal explorar a aplicação de Redes Neurais Convolucionais em dispositivos embarcados para o reconhecimento facial, por meio de um modelo de rede neural e a criação de um pipeline de reconhecimento facial, destacando os desafios dessa aplicação e sua eficácia.

Para alcançar esse objetivo, foram definidos objetivos específicos a serem realizados, são eles:

- Escolha de um modelo de reconhecimento facial da literatura;
- Criação de um *pipeline* e, consequentemente, de um algoritmo para a tarefa de reconhecimento facial utilizando o modelo escolhido;
- Criação de uma base de dados de rostos para testes utilizando o algoritmo e avaliação de sua acurácia;
- Implementação do sistema em um dispositivo embarcado limitado e comparação com outros hardwares.

1.3 Metodologia

Para atingir os objetivos específicos deste estudo, a metodologia foi dividida em 6 etapas, cada uma focada em aspectos cruciais do desenvolvimento do sistema de reconhecimento facial em dispositivos embarcados. As principais etapas incluem:

1. Revisão Sistemática de Artigos Relacionados ao Tema:

Essa revisão sistemática tem como propósito de fazer o levantamento do estado da arte para reconhecimento facial em dispositivos embarcados, analisando as abordagens utilizadas e seus resultados.

2. Escolha do Modelo:

Essa etapa consiste na seleção de um modelo de reconhecimento facial que seja leve o suficiente para ser utilizado em um dispositivo embarcado.

3. Criação do Pipeline e Algoritmo:

Essa etapa consiste na criação de um *pipeline* para a tarefa de reconhecimento facial e, a partir dele, a criação de um algoritmo em Python para sua execução. As etapas do *pipeline* seguirão a literatura sobre reconhecimento facial, incluindo detecção facial, extração de características e comparação.

4. Teste Prático e Coleta de Faces:

Com o algoritmo desenvolvido, será realizado um teste prático com o objetivo de obter rostos de voluntários para a criação de um banco de dados de rostos. Além disso, nesse teste será avaliado possíveis problemas do modelo e do *pipeline*.

5. Avaliação da Acurácia:

Com o banco de faces, será composto um algoritmo de teste de acurácia do modelo escolhido. Essa verificação utilizará comparação par a par, comparando a taxa de verdadeiro positivos com o total de faces utilizadas no teste.

6. Implementação do Sistema em Dispositivo Embarcado e Comparações:

Por fim, será realizado testes para comprovação da eficácia do *pipeline* desenvolvido em um dispositivo embarcado limitado (Orange Pi 3 LTS), e também comparações com outros dispositivos, concluindo com uma discussão sobre os resultados.

1.4 Estrutura do Documento

Este documento está dividido em capítulos com propostas diferentes, que podem ser verificados abaixo:

- Capítulo 2 - **Fundamentação Teórica:** Apresenta uma descrição dos assuntos que serão tratados durante o trabalho para um melhor entendimento do leitor.
- Capítulo 3 - **Trabalhos Relacionados:** Apresenta uma revisão de trabalhos relacionados e suas soluções apresentadas.
- Capítulo 4 - **Desenvolvimento:** Apresenta o que foi desenvolvido para alcançar os objetivos propostos, escolha do modelo, como criação de um *pipeline* e experimentações.

- Capítulo 5 - **Conclusão:** Aponta as considerações finais do trabalho, e os resultados obtidos.

2

Fundamentação Teórica

Este capítulo discute alguns conceitos necessários para o entendimento do projeto. A seção 2.1 apresenta uma introdução a respeito de rede neural, a seção 2.2 apresenta as redes neurais convolucionais, a seção 2.3 lista algumas técnicas de compressão de redes neurais e, por fim, a seção 2.4 aborda o problema do trabalho, o reconhecimento facial.

2.1 Redes Neurais

De modo geral, uma rede neural é uma máquina que é feita para modelar a forma que o cérebro desempenha uma tarefa em particular ou uma função de seu interesse (HAYKIN, 2009). No entanto, o objetivo das pesquisas modernas em redes neurais não é modelar o cérebro perfeitamente, mas desenvolver máquinas que façam aproximações funcionais avançadas, proporcionando uma compreensão de alto nível sobre o funcionamento do cérebro humano (GOODFELLOW; BENGIO; COURVILLE, 2016). Esta rede pode ser implementada utilizando componentes eletrônicos ou simulada em software, a segunda sendo mais comum.

A rede neural artificial lembra o cérebro em dois aspectos (HAYKIN, 2009):

1. O conhecimento da rede é adquirido a partir do ambiente através de um processo de aprendizado;
2. A força da conexão interneural, conhecida como pesos sinápticos, são usados para guardar o conhecimento adquirido.

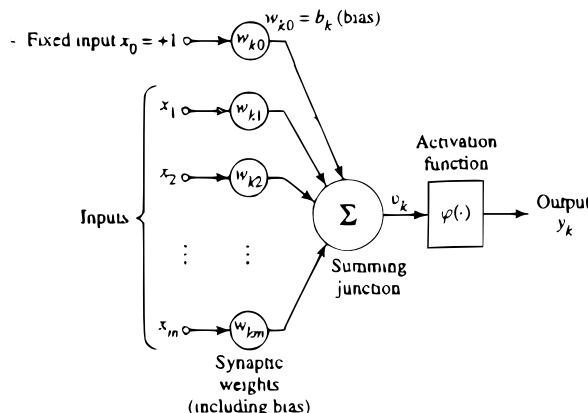
2.1.1 Neurônios artificiais

Segundo (HAYKIN, 2009), um neurônio artificial é uma unidade de processamento fundamental para a operação de uma rede neural. Essa unidade se assemelha a um neurônio no sentido de que ele recebe uma entrada de várias outras unidades e computa seu próprio valor de

ativação (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 1 mostra o modelo de um neurônio artificial. Abaixo, estão os três principais elementos básicos de um neurônio artificial:

1. Um conjunto de sinapses, cada uma é caracterizada por seu peso. Um sinal x_j na entrada de sinapse j conectada ao neurônio k é multiplicado pelo peso sináptico w_{kj} . Diferentemente dos pesos de uma sinapse no cérebro, os pesos sinápticos de um neurônio artificial podem variar em um faixa que inclui valores tanto positivos quanto negativos;
2. Um somador Σ para somar os sinais de entrada, ponderado pelas respectivas forças sinápticas do neurônio. Essa operação é descrita como uma combinação linear;
3. Uma função de ativação para limitar a amplitude da saída do neurônio. Ela limita a faixa de amplitude permitida do sinal de saída a algum valor finito. Além de restringir a faixa do sinal de saída, ela é crucial para introduzir não-linearidade, o que melhora a capacidade de predição da rede neural.

Figura 1 – Modelo não-linear de um neurônio



Fonte: (HAYKIN, 2009)

Ainda segundo (HAYKIN, 2009), o modelo também inclui um *bias* aplicado externamente, denotado por b_k . Este *bias* tem o efeito de aumentar ou diminuir a entrada total da função de ativação. Em termos matemáticos, descreve-se o neurônio k no par de equações a seguir:

$$u_k = \sum_{j=1}^m w_{kj} x_j$$

e

$$y_k = \varphi(u_k + b_k)$$

onde x_1, x_2, \dots, x_m são os sinais de entrada; $w_{k1}, w_{k2}, \dots, w_{km}$ são os pesos das ligações (sinápticas) respectivas do neurônio k ; u_k é a combinação linear dos sinais de entrada; b_k é o bias relativo ao neurônio; $\varphi(\cdot)$ a função de ativação e y_k é o sinal de saída final do neurônio.

2.1.2 Processo de Aprendizado

O processo de aprendizado de uma rede neural envolve a capacidade da rede em ajustar seus pesos e parâmetros internos para realizar tarefas específicas, como reconhecimento de padrões, classificação, regressão ou processamento de linguagem natural. Porém, para isso, o processo de aprendizado (ajuste de seus pesos e parâmetros) pode ser das seguintes categorias ([HAYKIN, 2009](#)):

- **Aprendizado supervisionado** - requer a disponibilidade de uma resposta desejada para a realização de um mapeamento específico de entrada e saída ao minimizar a função de custo de interesse, que desempenha o papel de medir a discrepância entre as previsões feitas por um modelo e as saídas reais (ou alvos) nos dados de treinamento.
- **Aprendizado não supervisionado** - depende apenas de exemplo não categorizados, consistindo simplesmente se um conjunto de sinais de entrada, os quais normalmente existem uma grande quantidade.
- **Aprendizado por reforço** - está entre o aprendizado supervisionado e não supervisionado, realiza o mapeamento de entrada e saída através de uma contínua interação do um sistema de aprendizado com o ambiente. O sistema de aprendizado realiza a ação e aprende a partir da resposta do ambiente à aquela ação.

2.1.3 Camadas da rede neural

Uma rede neural típica é composta por três camadas, a camada de entrada, as camadas ocultas e a camada de saída.

Camada de entrada: É por onde os dados entram na rede neural. o tipo de entrada pode ser variado, em um exemplo de classificação de imagens, cada neurônio na camada de entrada representaria um pixel.

Camadas ocultas: São as camadas intermediárias e também extremamente importantes para o aprendizado da rede. Elas agem como detectores de características. Enquanto o processo de aprendizado progride pelas camadas da rede neural, os neurônios começam a gradualmente "descobrir" as características proeminentes que caracterizam os dados de treinamento. E isso é possível pela realização de transformações não-lineares nos dados de entrada para um novo espaço, chamado de espaço de características. ([HAYKIN, 2009](#))

Camada de saída: Representa a predição final da rede neural. Em um exemplo de classificação, na qual existem 4 opções (classes), a camada final dessa rede teria 4 neurônios, cada uma representando uma classe.

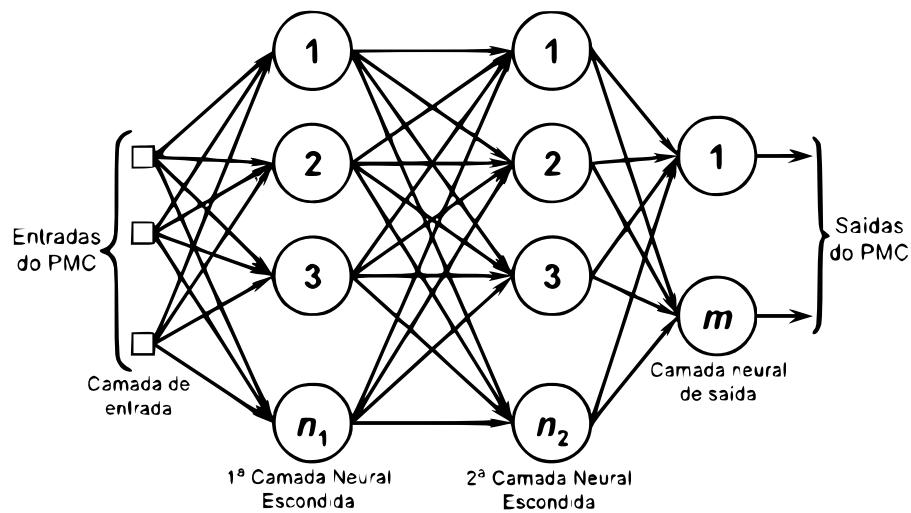
2.1.4 Rede Neural Perceptron Multicamadas

A rede neural Perceptron multicamadas é uma das redes neurais mais conhecidas. Uma de suas características principais é ser totalmente conectada, ou seja, cada nó em uma camada se conecta, com um determinado peso, a cada nó da camada seguinte. A Figura 2 apresenta o diagrama de um Perceptron Multicamadas.

Os seguintes pontos destacam as características básicas dos perceptrons multicamada ([HAYKIN, 2009](#)):

- Cada neurônio da rede inclui uma função de ativação não linear que é diferenciável.
- A rede contém uma ou mais camadas ocultas de ambos nós de entrada e saída.
- A rede exibe grande grau de conectividade, que é determinada pelos pesos sinápticos, que ligam a rede.

Figura 2 – Diagrama do Perceptron Multicamadas



Fonte: ([HAYKIN, 2009](#))

2.2 Redes Neurais Convolucionais

As Redes Neurais Convolucionais, ou CNN, são um tipo especializado de rede neural profunda projetada para tarefas de visão computacional, como reconhecimento de imagens, detecção de objetos e segmentação semântica. O estudo das CNNs começaram a partir do estudo do córtex visual do cérebro, e vem sendo usadas em reconhecimento de imagens desde 1980. Mas apenas nos últimos anos, devido ao aumento do poder computacional, e também da quantidade disponível de dados para treinamento, é que as CNNs conseguiram alcançar desempenho comparável ao de humanos em tarefas visuais mais complexas ([GERON, 2017](#)).

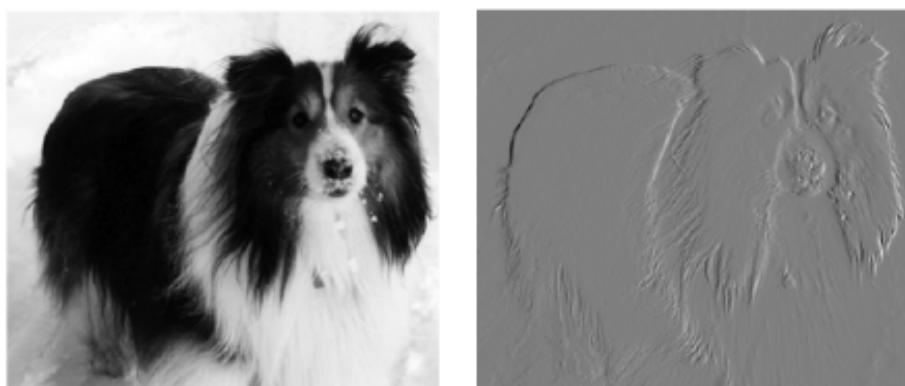
Uma rede neural convolucional é uma tipo específico de rede neural para processamento de dados que possui uma topologia em forma de grade. Exemplos incluem principalmente imagens, que podem ser pensadas como uma grade 2-D de pixels (GOODFELLOW; BENGIO; COURVILLE, 2016). O termo "convolucional" indica que a rede aplica a operação matemática chamada de operação correlação cruzada para sinais discretos 2D de duração finita que, por abuso de notação, convencionou-se chamar convolução. Pode-se resumir uma rede neural convolucional em simplesmente uma rede neural que utiliza uma camada convolucional ao invés da multiplicação de matrizes em pelo menos uma de suas camadas.

2.2.1 Estrutura e Funcionamento das CNNs

As CNNs são compostas por várias camadas interconectadas, incluindo camadas de convolução, camadas de *pooling* e camadas totalmente conectadas. A camada de convolução é o principal componente das CNNs levando maior parte do tempo dentro da rede e é responsável por extrair características relevantes de uma imagem (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Ela opera aplicando filtros , também conhecidos como *kernels*, à imagem de entrada. Esses filtros são utilizados nas operações de convolução para destacar padrões como bordas, texturas e formas.

Os filtros são modificados durante o treinamento da rede, permitindo que a CNN identifique automaticamente características discriminativas em diferentes níveis de abstração. Neurônios na primeira camada convolucional não estão conectados a cada um pixel da imagem de entrada, mas sim aos pixels em seus campos receptivos (GERON, 2017). A Figura 3 mostra um mapa de características, que é o resultado da aplicação de um filtro dentro da CNN.

Figura 3 – Exemplo de um mapa de características que realiza extração de bordas

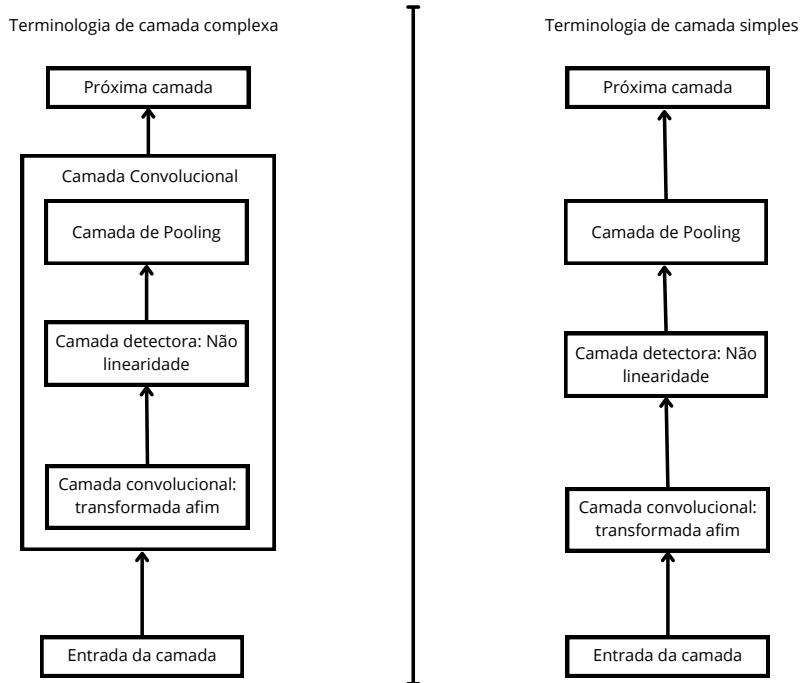


Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

Uma camada convolucional típica consiste em três estágios, como mostrado na Figura 4. No primeiro estágio, a camada realiza diversas convoluções simultaneamente para gerar um conjunto de ativações lineares. No segundo estágio, cada ativação linear é processada por uma função de ativação não-linear, como a função de ativação *rectified linear* (ReLU). Já na

terceira etapa, é utilizado uma função de *pooling* para reduzir a saída da camada ainda mais ([GOODFELLOW; BENGIO; COURVILLE, 2016](#)). Essa função de *pooling* substitui a saída da rede em uma determinada área por um valor que representa os valores próximos. Por exemplo, a operação *max pooling* retorna como saída o maior valor dentro de uma vizinhança retangular.

Figura 4 – Exemplo de uma camada convolucional



Fonte: Autor

No final da rede, há uma camada totalmente conectada, que combina as informações extraídas das camadas anteriores para realizar a tarefa de classificação de por exemplo, objetos em uma imagem. O treinamento de uma CNN envolve a otimização dos pesos dos filtros e conexões entre as camadas, utilizando algoritmos de otimização, como o gradiente descendente. Durante o treinamento, a CNN aprende a mapear automaticamente padrões de entrada para rótulos de saída desejados, permitindo que ela seja usada para tarefas de reconhecimento e classificação.

2.2.2 Métricas de Avaliação de uma CNN

A avaliação do desempenho de uma Rede Neural Convolucional é essencial para compreender sua eficácia em tarefas de visão computacional, como classificação de imagens, detecção de objetos e reconhecimento facial. Para medir a qualidade das previsões de uma CNN, várias métricas são comumente utilizadas. Esta seção apresenta algumas métricas fundamentais para avaliar uma CNN ([GERON, 2017](#)):

- 1. Acurácia (Accuracy):** A acurácia é uma métrica amplamente utilizada para avaliar uma CNN. Ela representa a proporção de previsões corretas em relação ao total de amostras de teste. No entanto, a acurácia sozinha pode ser enganosa quando as classes não estão balanceadas.

2. Matriz de Confusão (*Confusion Matrix*): A matriz de confusão é uma tabela que mostra o desempenho de uma CNN em todas as classes de um problema de classificação. A ideia geral é contar o número de vezes que uma instância A foi classificada como uma classe B.

3. Precisão (*Precision*): A precisão mede a proporção de previsões positivas corretas em relação ao total de previsões positivas. É particularmente relevante em problemas de classificação binária, onde é importante evitar falsos positivos.

4. Revocação (*Recall*): A Revocação, também conhecida como sensitividade, mede a proporção de previsões positivas corretas em relação ao total de amostras positivas reais. É importante em problemas onde é crucial evitar falsos negativos.

5. F1-score: O F1-score é uma métrica que combina precisão e revocação em uma única medida, sendo a média harmônica dos dois. Ele é usado para avaliar modelos de classificação, especialmente em casos de desequilíbrio entre classes. Um F1-score alto indica bom desempenho do modelo.

6. Curva ROC (*Receiver Operating Characteristic Curve*): A curva ROC é uma representação gráfica do desempenho de uma CNN em problemas de classificação binária. Ela mostra a taxa de verdadeiros positivos em relação à taxa de falsos positivos em vários pontos de corte.

2.2.3 Técnicas de diminuição do sobreajuste

O sobreajuste ou *overfitting* é um problema recorrente em modelos de CNN, ele ocorre quando o modelo se ajusta muito bem ao conjunto de dados de treino, mas se mostra ineficaz para dados não vistos. Existem várias técnicas para diminuição de sobreajuste, são mostrado algumas delas nos subseções a seguir.

2.2.3.1 Aumentação de Dados

Aumentação de Dados, traduzida do inglês *Data Augmentation*, é usado como uma forma de diminuição do sobreajuste principalmente quando existe uma base de dados menor do que a desejável. Esta é uma técnica na qual é possível ajustar as imagens ligeiramente, ao realizar diferentes ações nelas como rotação, corte, etc., para geração de mais dados para o modelo (SUBRAMANIAN, 2018). Esta técnica melhora significativamente a habilidade do modelo de classificar novas imagens fora dos dados de treinamento, pois apresenta novos dados não presentes no conjunto original de treinamento.

2.2.3.2 Regularização

Um dos métodos da regularização é a própria aumentação de dados, ao aumentar a variabilidade dos dados em diferentes estágios de uma CNN. Ao trabalhar com imagens, um método mais direto é modificações randômicas na imagem, como rotação e inversão. (SANTOS;

(PAPA, 2022). Além disso, entre outros métodos utilizados está a adição de penalidades à função de perda durante o treinamento, como por exemplo uma multiplicação por 0.5. Essa penalidade desencoraja o modelo a se tornar muito complexo ou possuir parâmetros com valores muito elevados. Alguns métodos de regularização são:

Dropout: A técnica de *Dropout* possui várias aplicações em redes neurais, mas a principal delas é a de diminuir o sobreajuste do modelo. Ela funciona da seguinte maneira, durante cada iteração no treinamento, cada neurônio é omitido da rede com uma probabilidade p . Entre as justificativas do porquê essa técnica funciona para diminuição do *overfitting* está que como há um grande número de estruturas de rede neural possíveis como resultado de randomicamente ocultar os neurônios do modelo, implicitamente o *dropout* realiza uma média sobre esse conjunto de possíveis redes, embora na prática apenas uma instância do modelo esteja sendo treinada (LABACH; SALEHINEJAD; VALAEE, 2019).

Regularização L1 & L2: A regularização L1, também conhecida como "*Lasso*", é uma técnica que adiciona uma penalidade à função de perda de um modelo, com base na soma dos valores absolutos dos coeficientes (pesos) dos atributos (*features*). Já a regularização L2, também conhecida como "*Ridge*", funciona adicionando uma penalidade à função de perda do modelo com base na soma dos quadrados dos coeficientes (pesos) dos atributos. Ao contrário da regularização L1, a L2 não leva a pesos de recurso exatamente iguais a zero, mas diminui a magnitude de todos os pesos (DEMIR-KAVUK et al., 2011). As duas técnicas funcionam muito bem para redução de sobreajuste, se diferenciando principalmente no fato da regularização L1 ser mais usada para seleção de atributos enquanto a regularização L2 ajuda a diminuir a magnitude dos pesos.

2.3 Métodos de Compressão de Redes Neurais

As redes neurais convolucionais (CNNs) são ao mesmo tempo intensivas no uso computacional e no uso de memória, as tornando difíceis de serem implantadas em sistemas embarcados de baixo custo (WANG et al., 2016). É por isso que existem métodos e processos já conhecidos para compressão de redes neurais. Esses métodos podem variar desde a diminuição da precisão de cada neurônio (seu tipo de dado), corte das conexões entre neurônios, transmissão de informações de uma rede a outra menor, etc. Porém, algo que todas elas possuem como objetivo é a diminuição do tamanho da rede sem que haja uma perda significativa na capacidade de predição do modelo.

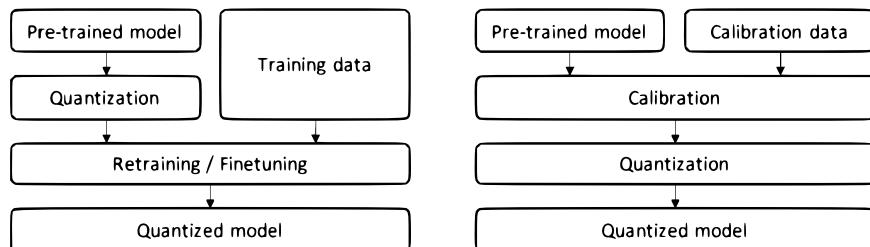
2.3.1 Quantização

Quantização é um processo de reduzir a precisão dos pesos, *biases* e ativações de uma forma que a rede neural consuma menos memória, além de também consumir menos energia (WANG et al., 2016). De uma forma que mesmo com a redução de precisão, ainda continue com

um bom valor de acurácia. Isso é adquirido a partir da conversão dos tipos utilizados nos tensores da rede neural, que são uma generalização de vetores e matrizes que podem existir em mais de duas dimensões, a qual por padrão utiliza pontos flutuantes de 32-bits para, por exemplo, inteiros de 8 bits.

Existem duas formas de aplicação de quantização mais utilizadas. A primeira é a quantização aplicada após o treinamento do modelo da rede neural, que acaba por diminuir a acurácia, pelo fato de que todos os parâmetros de quantização dos pesos e ativações são determinados sem a necessidade do retreinamento do modelo de rede neural ([GHOLAMI et al., 2021](#)). Porém, isso resulta em uma grande diminuição no uso de memória e processamento além de ser o método mais rápido de quantização. A segunda é a quantização ciente do treinamento, que possui a desvantagem do custo operacional devido ao retreinamento do modelo de rede neural, para que a acurácia possa ser recuperada. Se a acurácia for algo muito importante para o modelo, então esse investimento vale o custo. Na Figura 5 é possível visualizar a diferença entre as duas abordagens.

Figura 5 – Quantização ciente do treinamento e Quantização pós treinamento respectivamente



Fonte:([GHOLAMI et al., 2021](#))

2.3.2 Pruning

Também conhecido como Poda, é outro método muito utilizado para diminuição do uso computacional e de memória de CNNs. O método de *Network Pruning* (Poda de Rede) remove unidades redundantes ou irrelevantes - nós, filtros, ou até camadas inteiras - do modelo que não são críticas para a performance ([YEOM et al., 2021](#)). Ela é robusta a uma variedade de cenários e apresentam boas taxas de compressão enquanto afeta a acurácia do modelo minimamente. Além disso, a técnica de poda se mostrou efetiva na redução de complexidade de modelos de CNN e simultaneamente melhorando problemas de sobreajuste([YEOM et al., 2021](#)).

Técnicas de poda de rede atuais tornam pesos ou canais esparsos pela remoção de conexões não informativas e requerem um critério apropriado para identificação de quais unidades do modelo realmente não são relevantes para a solução de um problema ([YEOM et al., 2021](#)). Dessa forma, é crucial decidir a relevância dos parâmetros para serem eliminados sem sacrificar a performance de predição do modelo. Em estudos anteriores, critérios de poda de uma rede foram

propostos principalmente baseados na magnitude de seus pesos e gradientes. O critério de peso, é uma proposta em que a poda é realizada nos pesos nos quais a magnitude está abaixo de um certo limiar, transformando seu valor em 0 ou aplicando um ajuste fino. Já o critério de gradiente utiliza a magnitude do gradiente para encontrar atributos relevantes e não relevantes na rede.

Os algoritmos de Poda são divididos em duas categorias, poda estruturada e poda não estruturada (LIU et al., 2021). A poda não estruturada é refinada, e seu propósito é o de cortar as conexões de peso não importantes na rede neural pré-treinada. Isso resulta em CNNs esparsas com irregularidades, as quais usualmente requerem softwares especiais e aceleração de hardware para acelerar a velocidade de inferência. Em contraste, a poda estruturada é grosseira e pode remover completamente filtros não importantes. Dessa maneira, facilita alcançar o propósito de aceleração computacional.

2.3.3 Destilação de Conhecimento Professor-Aluno

Esta técnica funciona ao treinar um modelo menor (aluno) a partir dos dados de um modelo maior (professor) já bem treinado. A Destilação de conhecimento para compressão de modelos é similar à forma na qual os seres humanos aprendem (GOU et al., 2021). Após o treinamento do modelo professor, que apresenta bons resultados, o conhecimento adquirido pelo treinamento é destilado para o modelo estudante, e essa transferência pode ocorrer de algumas formas, o que define o tipo de destilação de conhecimento.

Dentre estas formas de destilação estão conhecimento baseado na resposta (*response-based knowledge*) que procura imitar a predição final do modelo do professor, conhecimento baseado em característica (*feature-based knowledge*) que procura utilizar as ativações dos mapas de características do professor no modelo do estudante e conhecimento baseado em relação (*relation-based knowledge*) que busca explorar mais as relações entre camadas e as amostras de dados (GOU et al., 2021).

2.4 Detecção e Reconhecimento Facial

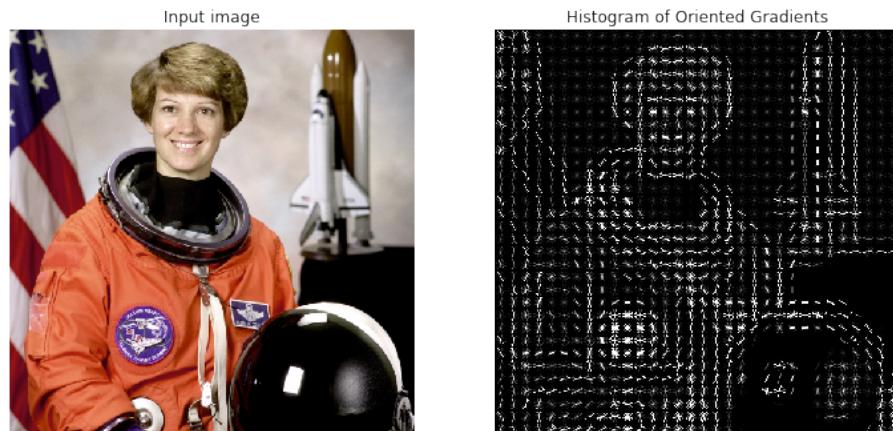
Dentro do *pipeline* do reconhecimento facial, duas etapas são as mais importantes (WANG et al., 2022): Detecção facial e reconhecimento facial. Após a captura do quadro de uma câmera que está gravando uma ou mais pessoas, a primeira etapa será a procura e corte dos rostos presentes nele. Isso é feito pelo método escolhido de detecção facial. E a segunda etapa, a extração de características, é realizada por um modelo de rede neural treinado para extrair informações dos rostos das imagens de entrada, transformando a imagem de entrada em um vetor, que armazena em forma de números as informações do rosto.

2.4.1 Alguns Métodos de Detecção Facial

Alguns métodos mais conhecidos para detecção facial são (RAHMAD et al., 2020):

- **Haar Cascade** - É um método conhecido de detecção facial, no qual classificadores treinados para detecção facial utilizando filtros são aplicados na imagem de entrada. Se a imagem passar com sucesso por todos os classificadores, significa que um rosto está presente na área selecionada. Esse método não utiliza redes neurais.
- **Histogram of Oriented Gradients (HOG)** - HOG são descritores de características que têm sido usada com sucesso para detecção de faces e também de objetos. Sua aplicação inicia ao transformar a imagem original em um vetor de características, calculando o histograma de gradientes da imagem, que mantém as partes mais importantes da imagem (suas bordas). Por fim, esse histograma é usado para treinar um modelo de inteligência artificial, como *Support Vector Machine* (SVM), para determinar se é um rosto ou não.. Este método também não utiliza redes neurais. Na Figura 6 é mostrado a aplicação do método HOG.

Figura 6 – Visualização das características HOG de uma imagem



Fonte: Opengenus¹

- **CNN** - Uma rede neural convolucional pode ser treinada para detectar faces em uma imagem. Inclusive, este é o método mais preciso, pois não depende do alinhamento das faces na imagem e é mais robusto a alguns problemas na detecção, como a oclusão facial. No entanto, isso vem com o custo de maior poder computacional em comparação com os algoritmos anteriores.

¹ Disponível em: <<https://iq.opengenus.org/object-detection-with-histogram-of-oriented-gradients-hog/>>. Acessado em Abril de 2024.

2.4.2 Problemas Comuns na Detecção Facial

Vários problemas podem ocorrer na etapa de detecção facial, que podem impedir a detecção apropriada dos rostos, alguns deles são ([KUMAR; KAUR; KUMAR, 2019](#)):

- **Resolução** - A resolução da imagem de entrada é importante na hora da detecção. Um rosto muito pixelado, causado por uma imagem de baixa resolução, é um problema para o algoritmo.
- **Iluminação** - Imagens com iluminação ruim causam grande dificuldade na detecção, principalmente quando o método escolhido não utiliza redes neurais.
- **Oclusão** - Oclusões faciais, como óculos, máscaras, entre outros, podem impedir a detecção facial e diminuir a precisão do reconhecimento facial posterior.
- **Expressões faciais** – Algumas expressões faciais complexas, como caretas, podem dificultar a detecção do rosto e até mesmo o reconhecimento posterior.

2.4.3 Reconhecimento Facial vs Verificação Facial

O problema de reconhecimento facial, que é o principal tema desse texto, se difere em alguns aspectos da verificação facial. Começando pelas semelhanças, os dois problemas precisam de um modelo de rede neural treinado para extrair características dos rostos, transformando a imagem de entrada de um rosto em um vetor de características. Já o problema da verificação facial é 1:1 ([OLOYEDE; HANCKE; MYBURGH, 2020](#)), ou seja, na verificação facial, como por exemplo ao desbloquear um celular pelo rosto, o algoritmo vai comprovar se a pessoa em questão é quem diz ser, ou seja, o dono do celular. Isso é feito ao comparar o vetor de características da foto tirada no momento do desbloqueio com um já armazenado no dispositivo.

Já o problema do reconhecimento facial trata de uma correspondência 1:N ([OLOYEDE; HANCKE; MYBURGH, 2020](#)). O problema consiste em verificar se uma pessoa está presente em um banco de imagens de faces. Um exemplo disso é na procura de criminosos pela polícia, que possui um banco de dados com vetores de características de muitos criminosos. Uma imagem de uma pessoa é inserida no modelo e, após seu vetor de características ser adquirido, ocorre uma comparação entre ele e todas as pessoas no banco. O melhor método de comparação pode ser escolhido a partir do problema proposto.

3

Trabalhos Relacionados

Este capítulo apresentará soluções relacionados ao trabalho, concentrando-se em produtos comerciais existentes, usadas para a tarefa de reconhecimento facial, e trabalhos acadêmicos com temas semelhantes ao proposto.

3.1 Soluções Comerciais

O dispositivo de baixo custo que foi utilizado nos testes futuros é a placa Orange Pi 3 LTS¹, de aproximadamente \$50 dólares, equipado com 2 GB RAM e uma CPU de 4 cores 1.8GHz. Essa placa não é específica para tarefas com inteligencia artificial, porém pode ser utilizada para essa tarefa por suportar sistema Linux e as biblioteca necessárias. A seguir será apresentado soluções comerciais populares que são utilizadas para solução de tarefas de reconhecimento facial. Na Figura 7 está a Orange Pi 3 LTS utilizada no trabalho.

Figura 7 – Placa Orange Pi 3 LTS



Fonte: Autor

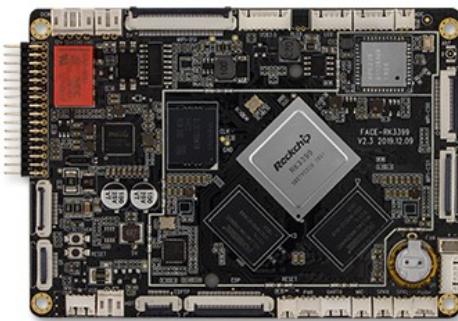
¹ Disponível em: <<http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/orange-pi-3-LTS.html>>. Acessado em Abril de 2024.

3.1.1 Face-RK3399 Face Recognition Main Board

A placa Face-RK3399 ² é pequena e possui performance para reconhecimento facial, suporta algoritmos de reconhecimento facial de vários rostos e modos de reconhecimento. É rica em interfaces periféricas, suportando diversos equipamentos. Ela cumpre os requisitos para aplicações de controle de acesso, atendimento e pagamento com o rosto.

Sobre suas especificações, possui uma processador Arm Cortex-A72 + quad-core Cortex-A53, 2GB LPDDR4 de memória RAM e 16GB eMMC de memória secundária. Possui Wi-Fi e utiliza Ubuntu como sistema operacional. Seu uso se dá por meio de uma API de reconhecimento facial, que já vem instalada no dispositivo. Seu custo é de \$149.00 dólares. É possível vê-la na Figura 8.

Figura 8 – Placa Face-RK3399



Fonte: FireFly³

3.1.2 Nvidia Jetson Nano

A Jetson Nano ⁴ é uma placa de desenvolvimento criada pela NVIDIA voltada para IA, incluindo diversas aplicação como reconhecimento facial. Ela veio ao mercado com o objetivo de utilizar, de forma compacta e de baixo custo energético, a tecnologia CUDA dos chips gráficos da NVIDIA. A própria disponibiliza diversas bibliotecas específicas para aplicações com IA, sendo um computador completo, porém com a possibilidade de aplicações complexas em Inteligencia artificial. Suas especificações incluem um processador ARM A57 quad-core, GPU NVIDIA com 128 CUDA cores, 4 GB de RAM e 16GB de memoria eMMC. Seu custo é de \$99.00 dólares, sendo relativamente barata, porém deve ser configurada e não possui uso otimizado para reconhecimento facial. Na Figura 9 é possível ver a Jetson Nano.

² Disponível em: <<https://en.t-firefly.com/product/facerk3399.html?theme=pc>>. Acessado em Abril de 2024.

³ Disponível em: <<https://www.firefly.store/goods.php?id=105>>. Acessado em Abril de 2024.

⁴ Disponível em: <<https://www.nvidia.com/pt-br/autonomous-machines/embedded-systems/jetson-nano/product-development/>>. Acessado em Abril de 2024.

⁵ Disponível em: <<https://www.amazon.com.br/desenvolvedor-Nano-Jetson-NVIDIA-945-13450-0000-100/dp/B084DSDDL>>. Acessado em Abril de 2024.

Figura 9 – NVIDIA Jetson Nano

Fonte: Amazon⁵

3.1.3 NXP EdgeReady Board

NXP EdgeReady⁶ é uma placa que aplica reconhecimento facial utilizando uma solução baseada em MCU, com o MCU crossover i.MX RT106F, que possibilita aos desenvolvedores adicionar rapidamente e facilmente reconhecimento facial com detecção de vivacidade aos seus produtos. Pode ser implementado em câmeras IR e RGB de baixo custo, sem a necessidade de câmeras 3D caras. O kit de desenvolvimento utilizado nessa solução, o SLN-VIZNAS-IOT, vem com software turnkey integrado, dessa forma, o reconhecimento facial é realizado inteiramente na borda. Sobre suas especificações, seu MCU é o MIMXRT106FDVL6A/B em conjunto com Arm® Cortex®-M7, 1 MB de SRAM e 32MB de SDRAM, com suporte a FreeRTOS. Seu custo é de \$228.85 dólares. Na Figura 10 é possível ver a placa.

Figura 10 – Placa SLN-VIZNAS-IOT

Fonte: NXP⁷

⁶ Disponível em: <<https://www.nxp.com/design/design-center/development-boards-and-designs/nxp-edgeready-mcu-based-solution-for-face-recognition-with-liveness-detection:SLN-VIZNAS-IOT>>. Acessado em Abril de 2024.

⁷ Disponível em: <<https://www.nxp.com/design/design-center/development-boards-and-designs/nxp-edgeready-mcu-based-solution-for-face-recognition-with-liveness-detection:SLN-VIZNAS-IOT>>. Acessado em Abril de 2024.

3.2 Trabalhos Acadêmicos

Antes do desenvolvimento do sistema, foi realizada uma revisão sistemática em 3 bases, utilizando os critérios abaixo para sua seleção:

- Artigos disponíveis integralmente no portal Periódicos CAPES;
- Devem ser relacionados ao tema de CNN, com aplicação a reconhecimento de face, compressão de modelos e/ou embarcados;
- Trabalhos publicados após 2018.

Para a realização da busca, foram utilizadas as seguintes bibliotecas digitais: IEEE Xplore, ACM Digital Library e ScienceDirect. As strings de buscas utilizadas foram:

- CNN AND (FACE AND (DETECTION OR RECOGNITION)) AND ((EDGE AND DEVICE) OR EMBEDDED OR COMPRESSION)

O processo de seleção dos estudos foi realizado por meio da pesquisa de artigos em bases conhecidas, utilizando as strings de busca. Obtendo os seguintes resultados:

Tabela 1 – Resultados da Revisão Sistemática

Bases	Artigos Selecionados
ScienceDirect	40
ACM	35
IEEE Xplore	13

Fonte: Autor

Posteriormente, os resumos dos artigos foram filtrados com base nos critérios de inclusão abaixo:

- **CI1** - Estudos que desenvolveram soluções para dispositivos de baixo custo
- **CI2** - Estudos que tenham como tema central detecção e reconhecimento de faces humanas em tempo real
- **CI3** - Estudos que utilizem técnicas de compressão
- **CI4** – Estudos que utilizem o dataset LFW

Entre os selecionados, foram selecionados aqueles que atendiam a mais critérios de inclusão (CI). Além dos selecionados, um outro artigo foi adicionado fora da revisão sistemática, o artigo [3.2.1](#) foi escolhido por ser o artigo de tema semelhante, ou seja, CNN em dispositivo embarcado, e por ser a dissertação de mestrado do Orientador do presente trabalho.

3.2.1 A Resource Constrained Pipeline Approach to Embed Convolutional Neural Models (CNNs)

Neste trabalho de dissertação, ([SILVA, 2023](#)) propõe uma solução de baixo custo e baixo consumo de energia para detecção de placas de trânsito, utilizando técnicas de compressão de rede neural como poda e quantização para que o modelo criado pelo autor possa caber na memória de um microcontrolador ESP32. Foi criado um pipeline completo para o modelo da rede neural, com objetivo final de reduzir a complexidade da rede, isso tudo com um bom nível de acurácia.

O *dataset* utilizado foi construído pelo próprio autor. A arquitetura usada foi baseada na LeNet, utilizando imagens de entrada de 32x32 pixels. A arquitetura possui 3 camadas convolucionais apenas, para diminuir significativamente a complexidade do modelo, com hiperparâmetros escolhidos a partir da técnica de otimização Bayesiana, para conseguir os melhores valores possíveis que compensem a compressão que foi utilizada. Como destacado, foram utilizadas várias técnicas para diminuição do tamanho do modelo, entre elas destilação de conhecimento, poda usando a técnica de poda de pesos baseada em magnitude e quantização dos pesos do modelo. Como resultado, foi alcançado um modelo de apenas 59KB com acurácia de 85.91%.

3.2.2 Deep Unified Model For Face Recognition Based on Convolution Neural Network and Edge Computing

Neste artigo, ([KHAN et al., 2019](#)) propuseram um novo modelo integrado chamado *Integrated Deep Model* (IDM) a partir de outros dois modelos já existentes, para melhorar a detecção de faces e bordas e superar técnicas mais tradicionais. O *dataset* usado para treinamento do modelo foi LFW([HUANG et al., 2008](#)) entre outros datasets *In The Wild*. Para validação da eficiência, foi proposto como teste uma sala de aula inteligente no qual as informações foram transmitidas através de uma arquitetura IoT utilizando computação na borda. De uma única imagem com 40 alunos, foi possível a detecção de 35 faces e reconhecimento de 30.

Foi alcançada uma redução de 62% na detecção de falsos positivos e uma acurácia de 97.9% nos dados de teste em comparação ao modelo original. Após estudos comparativos, a arquitetura proposta pelo autor conseguiu superar outras técnicas tradicionais em termo de latência e resposta em tempo real. No entanto, o sistema tem limitações em relação à distância, gerando resultados imprecisos em rostos desfocados em algumas situações, o método se mostrou eficaz em distâncias de cerca de 20 a 25 pés. Além disso, está atualmente integrado com sistemas de gerenciamento de aprendizado de várias instituições educacionais, com planos futuros de aprimoramento da robustez do sistema para enfrentar desafios como rostos inclinados, barbas, e observar tipos de comportamento com base no algoritmo de reconhecimento facial proposto.

3.2.3 *DGFaceNet: Lightweight and efficient face recognition*

Neste artigo, (ZHAO et al., 2023) propõem uma abordagem para reconhecimento facial de forma eficiente e leve, do ponto de vista computacional. Iniciam por apontar as dificuldades em portar modelos de reconhecimento facial complexos para dispositivos de borda e, simultaneamente, discutem sobre como escolher a arquitetura e uma função de perda leves podem ocasionar em uma perda significativa na acurácia, sendo esse um dos principais temas decorridos no texto.

Dessa forma, os autores propõem uma nova arquitetura de rede e função de perda para solucionar esse problema. A arquitetura proposta se baseia em uma estrutura de rede leve e eficiente, chamada de DGFace-Net. Após a descrição de como foi concebida, os autores apresentaram as suas capacidades, entre elas: a redução na quantidade de computações e no número de parâmetros requeridos pelo modelo de reconhecimento facial, tornando viável sua aplicação em dispositivos de borda. Além disso, foi proposta uma função de perda leve, chamada CML-softmax, que torna o treinamento desses modelos mais estável. Também foram realizadas comparações entre diversas arquiteturas de reconhecimento facial e a DGFace-Net, com dados de acurácia e processamento.

3.2.4 *Real-Time Multiple Face Recognition using Deep Learning on Embedded GPU System*

Neste artigo, (SAYPADITH; ARAMVITH, 2018) propõem um framework para reconhecimento de múltiplas faces a ser implementado em um sistema embarcado com GPU. A placa utilizada foi uma NVIDIA Jetson TX2 que, apesar de embarcada, é bastante poderosa. O modelo utilizado para o reconhecimento foi treinado em uma arquitetura com parâmetros reduzidos, e foi adicionada uma técnica de rastreamento de rostos para reduzir o tempo de processamento.

Uma breve descrição do *framework* proposto é a seguinte: primeiro, é aplicada a detecção e alinhamento de face no *frame* do vídeo de entrada. Após isso, o rosto é utilizado como entrada para o modelo de reconhecimento facial, e, finalmente, o rastreador de rostos é utilizado com a *label* do rosto detectado na imagem original. Dessa forma, juntamente com a capacidade de processamento em paralelo da GPU NVIDIA, é possível detectar até 8 rostos no vídeo de entrada, com uma velocidade de aproximadamente 4,34 QPS.

4

Desenvolvimento

O objetivo do presente trabalho, como discutido na Seção 1.2, é desenvolver um sistema para reconhecimento facial, utilizando um modelo de rede neural, que possa ser usado em dispositivos embarcados de baixo custo. Este capítulo discute as etapas necessárias para a criação do modelo, dentre elas a escolha de um modelo com baixa quantidade de parâmetros e tamanho pequeno, a criação de um banco de rostos para consequentemente avaliar a precisão do modelo escolhido e por fim, a extração de dados de sua aplicação em dispositivos embarcados. As subseções a seguir retratarão essas etapas.

4.1 Escolha do Modelo

Inicialmente, cogitou-se treinar um modelo de CNN do zero, com base em arquiteturas já existentes na literatura. No entanto, alguns fatores tornaram essa abordagem inviável para o presente trabalho. Isso inclui a demora no treinamento eficaz do modelo sem o hardware especializado, além da necessidade de realizar o treinamento novamente sempre que uma mudança na arquitetura do modelo fosse feita. Em vez disso, optou-se por outra abordagem: selecionar um modelo de tamanho leve, conhecido na literatura e com um modelo pré-treinado disponível na internet. Outro fator importante para escolha foi a popularidade do modelo, garantindo documentação sobre sua utilização.

O site *Papers With Code*¹ no tópico de *Lightweight Face Recognition* foi utilizado para escolha do modelo. Após analisar os modelos disponíveis, o MobileFaceNet (CHEN et al., 2018) foi escolhido por possuir uma pegada de memória baixa, de menos de 5MB, e por ser o modelo mais implementado no site. Após adquirir o modelo pré-treinado em formato Tensorflow, foram feitos pequenos testes de verificação facial par a par, no qual foram apresentados pares de imagens de pessoas iguais e depois de pessoas diferentes, o modelo se saiu bem no teste, acertando todas

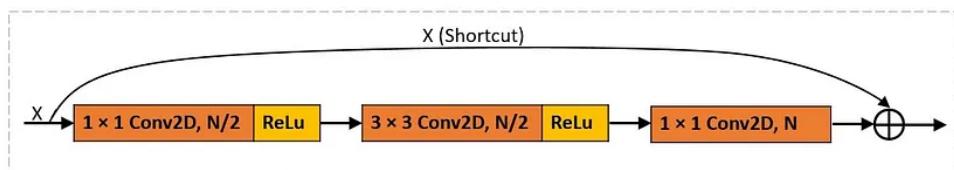
¹ Disponível em: <<https://paperswithcode.com/task/lightweight-face-recognition>>. Acessado em Abril de 2024.

as previsões e, dessa forma, foi de fato escolhido para ser utilizado nas próximas etapas.

O modelo MobileFaceNets foi inicialmente proposto em 2018, ele possui menos de 1 milhão de parâmetros e foi personalizado especificamente para reconhecimento facial em tempo real com alta acurácia em dispositivos móveis e embarcados. Foi treinado no *dataset* MS-Celeb-1M utilizando a função de perda *ArcFace*, seu tamanho é de apenas 4 MB, e conseguiu uma acurácia de 99.55% no *dataset* LFW. Além disso, no momento do lançamento, ele apresentava maior eficiência em comparação às CNN móveis existentes no estado da arte, que possuíam centenas de MB de tamanho.

O modelo utiliza blocos de camadas de 'bottleneck', que são um tipo específico de bloco de camadas de convolução, com o objetivo de melhorar a eficiência e reduzir a complexidade computacional. O termo 'bottleneck' ou 'gargalo', em português, vem do fato de reduzir o número de canais dos tensores de entrada antes de aplicar as operações de convolução e, em seguida, restaurar o número de canais. A Figura 11 mostra um exemplo de bloco de *bottleneck*.

Figura 11 – Um típico bloco de bottleneck

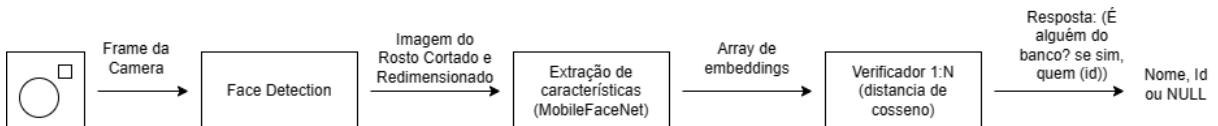


Fonte: Neetu Choudhary²

4.2 Pipeline do Reconhecimento Facial

Antes de adquirir o banco de rostos, foi necessário criar um *pipeline* e, consequentemente, um algoritmo que utilize o modelo escolhido para realizar o reconhecimento facial. A seguir estão descritas as etapas do *pipeline* e o algoritmo desenvolvido pode ser consultado no repositório do Github³. A Figura 12 mostra o *pipeline* desenvolvido para o trabalho.

Figura 12 – *Pipeline* de Reconhecimento Facial Utilizado



Fonte: Autor

² Disponível em: <<https://medium.com/@neetu.sigger/a-comprehensive-guide-to-understanding-and-implementing-bottleneck-residual-blocks-6b420706f66b>>. Acessado em Maio de 2024.

³ Disponível em: <https://github.com/leld21/teste_face_recognition>. Acessado em Maio de 2024.

4.2.1 Detecção de Face

Nesta primeira etapa, a partir de um imagem ou *frame* de uma câmera, o objetivo é utilizando algum método de detecção facial, descobrir a localização das faces na imagem e retornar uma nova imagem, do rosto recortado e redimensionado para ser utilizado posteriormente pelo modelo de extração de características (WANG et al., 2022). A partir das coordenadas da face retornada pelo método do HOG ou Haar Cascade, foi utilizado a biblioteca OpenCV⁴ para recortar e redimensionar a face da imagem original. Na Figura 13, é possível visualizar uma face destacada no *frame* da webcam com a utilização do método HOG.

Existem vários métodos para localização de faces na imagem. No entanto, para este *pipeline*, foi escolhido o HOG devido à sua confiabilidade, especialmente em imagens de boa resolução e com poucos rostos, além de possuir um custo computacional baixo em comparação com opções mais precisas, como uma CNN treinada para detectar faces na imagem. Posteriormente foi também utilizado o método Haar Cascade, para acelerar consideravelmente a fase de detecção de face ao embarcar. O método com CNN foi excluído por demandar muito poder computacional, inviabilizando sua aplicação em dispositivos embarcados.

Figura 13 – Exemplo de Detecção Facial



Fonte: Autor

4.2.2 Extração de Características

Nessa etapa o modelo escolhido, MobileFaceNet, é utilizado para adquirir um vetor de características a partir da face recortada retornada pelo detector de face. Cada modelo possui uma dimensão específica de entrada, e a do MobileFaceNet é de 112x112 pixels. Essa conversão para essa dimensão é realizada na etapa anterior. O retorno do modelo é um vetor de 256 valores *float* de 32 *bytes*, que representam as características da face presente na imagem de entrada. Esse vetor pode variar dependendo do modelo e também caso haja utilização de métodos de compressão.

⁴ Disponível em: <<https://opencv.org/get-started/>>. Acessado em Abril de 2024.

O modelo foi convertido, usando métodos do próprio Tensorflow, para o formato tflite⁵. Esse formato possui otimizações para aplicação em dispositivos móveis e embarcados que utilizam arquitetura ARM. Em conjunto com a arquitetura do MobileFaceNet, que utiliza blocos *bottleneck* para otimizar o processamento, a etapa de extração de características possui um tempo de processamento mínimo.

4.2.3 Checagem de Pares com Distância do Cosseno

A última etapa desse *pipeline* de reconhecimento facial é a de utilizar o vetor de características retornado pelo modelo na etapa anterior, e de alguma forma comparar par a par com o banco de faces já existente para checar se a face da pessoa já está cadastrada ou não. O método escolhido para isso foi a distância do cosseno (NGUYEN; BAI, 2010), que compara dois vetores - neste caso, vetores de 256 valores *float* - e retorna o quanto distantes eles estão um do outro. Normalmente, esse valor está entre 0 e 1, e um valor de *threshold* é escolhido com base no modelo. Caso o valor retornado pela função da distância do cosseno for menor que esse *threshold*, o par de faces dado como parâmetro pertencem à mesma pessoa, e caso seja maior, são pessoas diferentes.

Esse processo é repetido para cada vetor de características, que representa a face de uma pessoa existente no banco de faces. Após isso, é feito um filtro com todos valores retornados pela função de distância do cosseno, e é escolhido o de menor valor, que representa a melhor escolha entre as que passaram pelo *threshold*. O retorno final do pipeline, caso haja algum valor menor que o *threshold* escolhido, é o Id ou Nome cadastrado no banco referente a pessoa. Caso não haja, significa que a face dada de entrada ao pipeline não está cadastrada no banco.

4.3 Experimento Prático e Criação do Banco de Rostos

Para a realização de uma avaliação robusta do modelo escolhido foi necessário a criação de um banco de rostos. Para isso, o programa criado com o pipeline detalhado na subsessão 4.2 foi implantado em um computador com webcam e ficou em funcionamento no período de 18 a 22 de março no corredor de entrada do DComp na Universidade Federal de Sergipe.

O programa possibilitava o reconhecimento facial em tempo real, e caso a pessoa não estivesse cadastrada no banco de rostos poderia realizar sua inserção na base de dados local, ao clicar no botão "Adicionar seu Rosto" que se posicionava abaixo da tela de reconhecimento. Ao término do experimento, foram obtidos aproximadamente 200 novos rostos de várias pessoas que colaboraram durante os 5 dias de exposição. A Figura 14 apresenta uma imagem do experimento.

Além da criação do banco de dados, o objetivo deste experimento foi testar o algoritmo e o modelo de reconhecimento facial em uma aplicação real, buscando identificar alguns problemas

⁵ Disponível em: <<https://www.tensorflow.org/lite?hl=pt-br>>. Acessado em Maio de 2024.

que podem surgir na prática. Após a pessoa cadastrar seu rosto no banco, ela poderia testar se o modelo conseguia identificar corretamente a pessoa. Caso a pessoa fosse identificada, apareceria na tela o texto "Bem vindo, {nome da pessoa}" como mostrado na Figura 13.

Figura 14 – Experimento de Coleta de Faces Realizado no DComp-UFS



Fonte: Autor

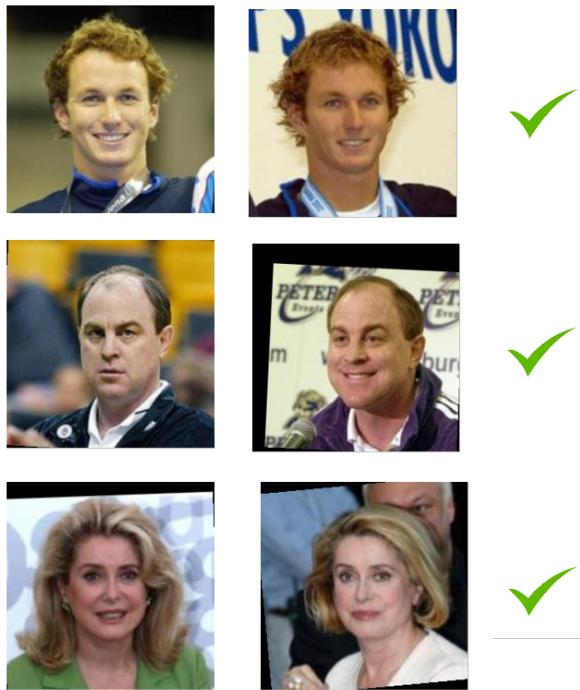
Alguns problemas foram encontrados como, por exemplo, a dificuldade do detector facial quando há um grande contraste entre o rosto da pessoa e o cenário de fundo. Outro problema encontrado, desta vez relacionado ao modelo, foi a oclusão da face, especialmente com o uso de óculos. A detecção facial ocorre sem nenhum problema, porém o fato da pessoa usar óculos é relevante na hora da extração de características do modelo, fazendo com que duas pessoas com óculos parecidos, caso não se diferenciem muito em características faciais, possa aparecer falsos positivos.

4.4 Avaliação do modelo

Com o banco de rostos adquirido, foi possível realizar a avaliação de acurácia do modelo MobileFaceNet. Além dos 200 rostos adquiridos no experimento, foram adicionados aproximadamente 100 rostos retirados do dataset LFW⁶ para que houvesse comparações par a par entre outros exemplares dos rostos das pessoas adicionadas com as 300 presentes no banco. 112 rostos foram comparados, par a par com os presentes no banco. Como resultado, foram obtidos 111 reconhecimentos corretos (verdadeiros positivos) e 1 incorreto (falso positivo), resultando em uma acurácia de 99,1%, *Recall* de 1.0 e *F1-Score* de 0.995 após a adição das faces do dataset. Um dos motivos para um valor alto de *Recall* está no fato de haver varias tentativas em um vídeo para reconhecer o rosto. Isso demonstra uma capacidade de reconhecimento do modelo escolhido, sendo confiável para as características em que as imagens foram obtidas, como distância, iluminação, fundo, dentre outras. Na Figura 15 é mostrado um exemplo de comparação par a par.

⁶ Disponível em: <<https://vis-www.cs.umass.edu/lfw/>>. Acessado em Maio de 2024.

Figura 15 – Exemplo de Comparaçāo Par a Par



Fonte: Autor

4.5 Dados de Aplicação em Dispositivos Embarcados

O algoritmo desenvolvido que utiliza o modelo MobileFaceNet para reconhecimento facial foi testado em três dispositivos diferentes com o objetivo de extrair dados comparativos. Um notebook com processador i7 de 11^a geração, e dois dispositivos embarcados, sendo um mais potente, NVIDIA Jetson Nano com GPU, e outro com menor poder computacional e baixo custo como proposto no trabalho: um Orange Pi 3 LTS. Essa placa é uma concorrente mais barata que um Raspberry Pi, e vem equipada com processador AMD Cortex-A53, 2 GB RAM e 8 GB de memória secundária eMMC. Seu valor comercial é inferior a US\$ 50 ou R\$ 250, se mostrando uma opção viável, principalmente a médio e longo prazo.

Os principais dados analisados foram: velocidade da detecção facial utilizando dois métodos, velocidade de inferência do modelo, pegada de memória e latência de todo processo do pipeline. As configurações foram as mesmas para os três hardwares, *frame* de entrada de 640x480 *pixels* e todos *frames* recebidos sendo processados para reconhecimento facial. Abaixo é possível visualizar as médias e desvios padrão de cada equipamento, que foram extraídos a partir dos testes realizados. O dado da primeira linha é a quantidade de quadros processados utilizando o método de HOG para detecção facial e em seguida, o método de Haar Cascade. Nas 4^a e 5^a colunas, estão o tempo de processamento dos algoritmos de detecção facial, o tempo de inferência do modelo e a pegada de memória. Na inferência utilizando o Jetson Nano, foi utilizado sua GPU para acelerar o processamento, dessa forma, o uso de sua RAM como memória gráfico foi alto.

Tabela 2 – Comparação Entre os Três Hardwares

Critério	Notebook com i7 11th	NVIDIA Jetson Nano	Orange Pi 3 LTS
QPS(Hog)	8.21 +- 0.27 QPS	1.49 +- 0.09 QPS	0.90 +- 0.02 QPS
QPS(Haar)	18.35 +- 1.18 QPS	3.78 +- 0.31 QPS	2.27 +- 0.11 QPS
Tempo detecção (Hog)	85.14 +- 1.93 ms	603.48 +- 16.87 ms	927.95 +- 23.50 ms
Tempo detecção (Haar)	22.43 +- 1.13 ms	173.92 +- 11.90 ms	213.97 +- 19.73 ms
Tempo de inferência	28.70 +- 1.16 ms	23.19 +- 2.14 ms	161.23 +- 16.06 ms
Pegada de memória	325 MB	1260 MB	460 MB

Fonte: Autor

Com os dados comparativos, é possível levantar imediatamente algumas informações. Iniciando pelos dois métodos de detecção facial, o método de Haar Cascade é aproximadamente 3 a 4 vezes mais rápido que o método de HOG, porém ao custo da perda na precisão, que é inferior ao método HOG. A utilização da GPU do Jetson Nano proporcionou um tempo de inferência menor do que comparado ao do processador i7 do notebook, demonstrando o poder de processamento da GPU em tarefas de predição utilizando inteligência artificial. A razão do Jetson Nano ter um valor alto de uso memória está na utilização da GPU, que converte a memória RAM em VRAM para operações matemáticas. E por fim, analisando os dados extraídos da Orange Pi 3 LTS, houve resultado satisfatório, principalmente ao se utilizar o método Haar Cascade, de 2.27 QPS. Para aplicações controladas, sem a necessidade de alta velocidade de reconhecimento, a aplicação do algoritmo e modelo propostos em um dispositivo embarcado de baixo custo como o Orange Pi 3 LTS é viável.

5

Conclusão

O objetivo deste trabalho, foi a escolha de um modelo de rede neural para reconhecimento facial e consequente criação de um *pipeline* para que fosse possível embarcar em um dispositivo limitado. Para isso, além do estudo e seleção do modelo, foi desenvolvido um pipeline para a execução do modelo. Esse pipeline inclui etapas como detecção facial, extração de características pelo modelo escolhido e comparação entre vetores de *embeddings* utilizando a distância de cosseno.

Inicialmente, considerou-se a possibilidade de criar um modelo do zero. No entanto, devido a limitações de tempo e hardware, optou-se pela escolha de um modelo pré-treinado reconhecido na literatura. Esse modelo deveria ser compacto o suficiente para ser embarcado em um dispositivo de borda e possuir vasta documentação. Dessa forma, foi escolhido o modelo MobileFaceNet, que foi desenvolvido em 2018 para ser utilizado em dispositivos móveis. Além do modelo para a fase de extração de características, foi necessário a escolha do método para etapa de detecção facial, sendo testado para isso os métodos de Haar Cascade e HOG.

Com o algoritmo que segue o pipeline proposto desenvolvido, foi realizado experimentos práticos para analisar possíveis problemas na detecção e reconhecimento facial, além de obter faces de voluntários para criar uma base e futuramente realizar uma avaliação da acurácia do modelo. Alguns problemas como iluminação na fase de detecção e oclusão facial por óculos durante a extração de característica foram relatados. Isso expôs algumas das fragilidades que podem ocorrer no reconhecimento facial em tempo real. Após a colega das faces foi realizado a avaliação do modelo, resultando em uma acurácia de 99,1%, *Recall* de 1.0 e *F1-Score* de 0.995.

O dispositivo embarcado de baixo custo escolhido para ser testado foi o Orange Pi 3 LTS, de aproximadamente US\$ 50. O desempenho do processo de reconhecimento facial com o algoritmo desenvolvido foi comparado entre o Orange Pi 3 LTS, uma NVIDIA Jetson Nano, e um notebook equipado com processador i7 de 11º geração. O resultado da taxa de processamento de quadros na aplicação do pipeline no Orange Pi 3 LTS foi de 2.27 QPS para todo processo de

reconhecimento utilizando o método Haar Cascade para detecção facial. Isso demonstrou uma capacidade para reconhecimento em tempo real, adequado para aplicações que não requerem alta velocidade, como reconhecimento facial para controle de acesso em condomínios.

Em conclusão, este trabalho demonstrou que é viável realizar o reconhecimento facial em dispositivos embarcados com recursos limitados. Há espaço para melhorias visando o aprimoramento do desempenho do reconhecimento facial e sua aplicação em dispositivos ainda mais limitados do que o Orange Pi 3 LTS. Entre as possibilidades, está o desenvolvimento de um modelo ainda menor que o escolhido, com posterior aplicação de técnicas de compressão como poda e quantização. Além disso, há espaço para uma otimização do atual algoritmo para diminuir o tempo de processamento necessário pelo *pipeline* do reconhecimento.

Referências

- ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. In: *2017 International Conference on Engineering and Technology (ICET)*. [S.l.: s.n.], 2017. p. 1–6. Citado na página 20.
- BARNOUTI SINAN SAMEER MAHMOOD AL-DABBAGH, W. E. M. N. H. Face recognition: A literature review. *International Journal of Applied Information Systems*, Foundation of Computer Science (FCS), NY, USA, New York, USA, v. 11, n. 4, p. 21–31, Sep 2016. ISSN 2249-0868. Disponível em: <<https://www.ijais.org/archives/volume11/number4/935-2016451597/>>. Citado na página 11.
- CHEN, S. et al. Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices. 04 2018. Citado na página 34.
- DEMIR-KAVUK, O. et al. Prediction using step-wise l1, l2 regularization and feature selection for small data sets with large number of features. *BMC Bioinformatics*, v. 12, n. 1, p. 412, October 25 2011. ISSN 1471-2105. Disponível em: <<https://doi.org/10.1186/1471-2105-12-412>>. Citado na página 23.
- GERON, A. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. Sebastopol, CA: O'Reilly Media, 2017. ISBN 978-1491962299. Citado 3 vezes nas páginas 19, 20 e 21.
- GHOLAMI, A. et al. *A Survey of Quantization Methods for Efficient Neural Network Inference*. 2021. Citado na página 24.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <<http://www.deeplearningbook.org>>. Citado 4 vezes nas páginas 16, 17, 20 e 21.
- GOU, J. et al. Knowledge distillation: A survey. *International Journal of Computer Vision*, Springer Science and Business Media LLC, v. 129, n. 6, p. 1789–1819, mar 2021. Disponível em: <<https://doi.org/10.1007%2Fs11263-021-01453-z>>. Citado na página 25.
- HAYKIN, S. S. *Neural networks and learning machines*. Third. Upper Saddle River, NJ: Pearson Education, 2009. Citado 4 vezes nas páginas 16, 17, 18 e 19.
- HUANG, G. B. et al. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In: *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*. [S.l.: s.n.], 2008. Citado na página 32.
- KHAN, M. Z. et al. Deep unified model for face recognition based on convolution neural network and edge computing. *IEEE Access*, v. 7, p. 72622–72633, 2019. Citado na página 32.
- KUMAR, A.; KAUR, A.; KUMAR, M. Face detection techniques: A review. *Artificial Intelligence Review*, v. 52, 08 2019. Citado na página 27.
- LABACH, A.; SALEHINEJAD, H.; VALAEE, S. Survey of dropout methods for deep neural networks. *arXiv preprint arXiv:1904.13310*, 2019. Citado na página 23.

- LEE, J.; NA, W. A survey on mobile edge computing architectures for deep learning models. In: *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. [S.l.: s.n.], 2022. p. 2346–2348. Citado na página 12.
- LIU, Y. et al. Superpruner: Automatic neural network pruning via super network. *Sci. Program.*, Hindawi Limited, London, GBR, v. 2021, jan 2021. ISSN 1058-9244. Disponível em: <<https://doi.org/10.1155/2021/9971669>>. Citado na página 25.
- NGUYEN, H.; BAI, L. Cosine similarity metric learning for face verification. In: . [S.l.: s.n.], 2010. v. 6493, p. 709–720. ISBN 978-3-642-19308-8. Citado na página 37.
- OLOYEDE, M.; HANCKE, G.; MYBURGH, H. A review on face recognition systems: recent approaches and challenges. *Multimedia Tools and Applications*, v. 79, 10 2020. Citado na página 27.
- PLASTIRAS, G. et al. Edge intelligence: Challenges and opportunities of near-sensor machine learning applications. In: *2018 IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. [S.l.: s.n.], 2018. p. 1–7. Citado na página 13.
- RAHMAD, C. et al. Comparison of viola-jones haar cascade classifier and histogram of oriented gradients (hog) for face detection. *IOP Conference Series: Materials Science and Engineering*, v. 732, p. 012038, 01 2020. Citado na página 26.
- SANTOS, C. F. G. D.; PAPA, J. a. P. Avoiding overfitting: A survey on regularization methods for convolutional neural networks. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 54, n. 10s, sep 2022. ISSN 0360-0300. Disponível em: <<https://doi.org/10.1145/3510413>>. Citado na página 23.
- SAYPADITH, S.; ARAMVITH, S. Real-time multiple face recognition using deep learning on embedded gpu system. In: *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. [S.l.: s.n.], 2018. p. 1318–1324. Citado na página 33.
- SILVA, R. A. da. A resource constrained pipeline approach to embed convolutional neural models (cnns). 2023. Citado na página 32.
- SUBRAMANIAN, V. *Deep Learning with PyTorch*. First. Livery Place, 35 Livery Street, Birmingham, B3 2PB, UK.: Packt Publishing, 2018. Citado na página 22.
- TASKIRAN, M.; KAHRAMAN, N.; ERDEM, C. E. Face recognition: Past, present and future (a review). *Digital Signal Processing*, v. 106, p. 102809, 2020. ISSN 1051-2004. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S1051200420301548>>. Citado na página 11.
- WANG, X. et al. *A Survey of Face Recognition*. 2022. Citado 2 vezes nas páginas 25 e 36.
- WANG, Y. et al. Low power convolutional neural networks on a chip. In: . [S.l.: s.n.], 2016. p. 129–132. Citado na página 23.
- YEOM, S.-K. et al. Pruning by explaining: A novel criterion for deep neural network pruning. *Pattern Recognition*, v. 115, p. 107899, 2021. ISSN 0031-3203. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0031320321000868>>. Citado na página 24.

ZHAO, F. et al. Dgfacenet: Lightweight and efficient face recognition. *Engineering Applications of Artificial Intelligence*, v. 124, p. 106513, 2023. ISSN 0952-1976. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0952197623006978>>. Citado na página 33.