Pós-Graduação em Ciência da Computação

Marcelo Victor Batista da Silva

Arquitetura integrada 5G-MEC para o processamento de aplicações de visão computacional

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

Recife
2024

Marcelo Victor Batista da Silva

Arquitetura integrada 5G-MEC para o processamento de aplicações de visão computacional

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Área de Concentração: *Redes de Computadores e Sistemas Distribuídos*
Orientador: *Kelvin Lopes Dias*

Recife
2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Silva, Marcelo Victor Batista da.
Arquitetura integrada 5G-MEC para o processamento de aplicações de visão computacional / Marcelo Victor Batista da Silva. - Recife, 2024.
54 p. : il., tab.

Orientador(a): Kelvin Lopes Dias
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2024.
Inclui referências.

1. 5G. 2. MEC. 3. Computação de borda. 4. Nuvem. 5. Visão Computacional. I. Dias, Kelvin Lopes. (Orientação). II. Título.

000 CDD (22.ed.)

Marcelo Victor Batista da Silva

Arquitetura integrada 5G-MEC para o processamento de aplicações de visão computacional

Trabalho de Conclusão de Curso apresentado como pré-requisito para conclusão do Curso de Bacharelado em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco. Defendida e aprovada em 29 de julho de 2024 pela seguinte banca examinadora:

Kelvin Lopes Dias
Orientador/CIn-UFPE

Nelson Souto Rosa
Examinador/CIn-UFPE

Eu dedico essa dissertação para toda minha família.

AGRADECIMENTOS

Primeiramente, agradeço minha família, sou profundamente grato pelo amor, incentivo e apoio constante.

Agradeço aos meus professores pelas conversas inspiradoras sobre minha pesquisa, que foram fundamentais para o desenvolvimento deste trabalho. Aos meus amigos, deixo minha eterna gratidão por me encorajarem e por estarem ao meu lado durante esta jornada.

Agradeço à UFPE pelo ambiente e recursos que permitiram o crescimento dos meus conhecimentos. Agradeço também aos membros da banca pela disponibilidade e contribuição na avaliação desta proposta.

ABSTRACT

Processing computer vision applications on mobile devices is challenging due to the limitations related to battery consumption and computational power of these devices for this type of application. While cloud-based remote processing offers abundant computational resources, it results in significant delays that can cause problems with critical and real-time applications. Computational offloading to edge servers, located close to the end device, has been adopted by industry and academic research as a way to address these challenges. Additionally, 5G access can also benefit computer vision applications by providing lower latency and higher bandwidth compared to previous generations of cellular networks. With the increasing number of mobile operators and Internet service providers based on 5G access, it becomes necessary to design solutions for the execution of real-time applications with the assistance of edge computing. Furthermore, open-source platforms for Multi-Access Edge Computing (MEC) and the 5G core can be deployed for rapid prototyping and testing of applications. This graduation project first proposes an end-to-end solution composed of 5G core and MEC based on open-source platforms, interoperating with a wireless access network based on proprietary 5G radio. Then, a computer vision application for sentiment analysis was developed and integrated into the proposed 5G-MEC platform. Finally, is performed a performance evaluation of the solution and compared it with a traditional cloud-based remote processing approach to highlight the benefits of our proposal.

Keywords: MEC. Cloud. 5G. Edge. Computer Vision.

RESUMO

O processamento de aplicações de visão computacional em dispositivos móveis é desafiador devido às limitações relacionadas ao consumo de bateria e poder computacional desses equipamentos para este tipo de aplicação. Embora o processamento remoto baseado em nuvem ofereça recursos computacionais abundantes, acarreta atrasos significativos que podem prejudicar aplicações críticas e com requisitos de tempo real. O *offloading* computacional para servidores de borda, próximos ao dispositivo final, tem sido adotado pela indústria e pela pesquisa acadêmica como forma de endereçar tais desafios. Além disso, o acesso 5G também pode beneficiar as aplicações de visão computacional, proporcionando menor latência e maior largura de banda em comparação com gerações anteriores de redes celulares. Com o aumento do número de operadoras móveis e provedores de serviços de Internet baseados no acesso 5G, torna-se de suma importância a concepção de soluções para a execução de aplicações de tempo real com a assistência da computação de borda. Adicionalmente, plataformas baseadas em código aberto para *Multi-Access Edge Computing* (MEC) e para o núcleo 5G podem ser implantadas para a prototipagem rápida e teste de aplicações. Este Trabalho de graduação primeiramente propõe uma solução fim a fim composta por núcleo 5G e MEC baseados em plataformas de código aberto, interoperando com uma rede de acesso sem fio baseada em rádio 5G proprietário. Em seguida, uma aplicação de visão computacional para a análise de sentimentos foi desenvolvida e integrada à plataforma 5G-MEC proposta. Por fim, é realizada uma avaliação de desempenho da solução e a comparamos com uma abordagem tradicional de processamento remoto em nuvem computacional para destacar os benefícios dessa proposta.

Palavras-chave: MEC. Nuvem. 5G. Borda. Visão Computacional.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

1G	*First Generation of Mobile Networks*
2G	*Second Generation of Mobile Networks*
3G	*Third Generation of Mobile Networks*
3GPP	*3rd Generation Partnership Project*
4G	*Fourth Generation of Mobile Networks*
5G	*Fifth Generation of Mobile Networks*
5GC	*5G Core Network*
AMF	*Network Function*
API	*Application Programming Interface*
AR	Realidade Aumentada
CNN	*Convolutional Neural Network*
ETSI	*European Telecommunications Standards Institute*
FER	*Facial Expression Recognition*
gNB	*Next Generation Node B*
IA	Inteligência Artificial
Iot	*Internet of Things*
IoT	*Internet of Things*
ITU	*International Telecommunications Union*
LCM	*Life-Cycle Management*
Mbps	*Mega bits per second*
MEC	*Multi-access Edge Computing*
MEC App	Aplicação MEC
MEC *System*	Sistema MEC
MEH	MEC *Host*
MEO	*MEC Orchestrator*
MEP	Plataforma MEC
MEPM	MEC Platform Manager
MIMO	*Multiple Input Multiple Output*
ML	*Machine Learning*
mmWW	*Millimeter Wave*
MTCNN	*Multi Cascade Convolutional Network*
NF	*Network Function*
O-Net	*Output Network*
OAI	*OpenAirInterface*
OOS	*Operator's Operations Support System*

P-Net	*Proposal Network*
QoS	*Quality of Service*
R-Net	*Refine Network*
RAN	*Radio Access Networks*
RTT	*Round-Trip Time*
SBA	*Service Based Achitecture*
SDN	*Software Defined Network*
SMF	*Session Management Function*
SMS	*Short Message Service*
UE	*User Equipment*
UPF	*User Plane Function*
V2X	*Vehicle-to-everything*
VANTs	Veículo Aéreo Não Tripulado
VIM	*Virtualization/Virtualized Infrastructure Manager*
VR	*Virtual Reality*

LISTA DE SÍMBOLOS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo são apresentados o contexto e a motivação deste trabalho de conclusão de curso. Posteriormente, o problema da pesquisa e os objetivos são evidenciados. Por último, são apresentadas a metodologia e a organização do documento.

1.1 CONTEXTO E MOTIVAÇÃO

As aplicações de visão computacional ganharam impulso em várias indústrias verticais e na sociedade. Estima-se que tais aplicações gerem uma receita de US$ 17,25 bilhões até 2024 e deverá atingir US$ 39 bilhões até 2029 (MordorIntelligence, 2024). Aplicações baseadas em detecção, reconhecimento e rastreamento de objetos estão começando a ser amplamente implantadas na agricultura de precisão, vigilância, chão de fábrica, cidades inteligentes e educação, para citar alguns exemplos. No entanto, essas aplicações exigem considerável poder de processamento para executar modelos de aprendizado de máquina. Da perspectiva do usuário final e do consumo de energia de equipamentos do usuário (*User Equipment* (UE)) alimentados por bateria, em geral, não é viável depender de smartphones, drones ou dispositivos de *Internet of Things* (IoT) para executar essas aplicações que demandam muito processamento. Para atender a essas demandas, as tarefas de processamento de visão computacional geralmente são executadas em *data centers* remotos/ambientes de computação em nuvem (Motlagh *et al.*, 2017).

Embora esse método tenha resolvido os problemas de alto consumo de energia e bateria em dispositivos móveis, essas aplicações em tempo real também têm requisitos rigorosos em termos de latência. Aplicações que dependem de computação em tempo real requerem respostas quase instantâneas para operar de maneira eficiente. A localização física dos servidores de computação em nuvem impacta diretamente o atraso na resposta das aplicações, o que pode ser problemático em casos onde a latência é crítica, como em aplicações de visão computacional.

Apesar dos benefícios de processamento da computação em nuvem tradicional, para reduzir efetivamente a latência para aplicações sensíveis ao tempo, é necessário processamento e armazenamento mais próximos do usuário final ou da fonte de dados. Atualmente, essa abordagem é descrita como computação de borda (Lin *et al.*, 2019), com outros termos dependendo da tecnologia de acesso, localização da borda, dispositivos envolvidos, distribuição do processa-

mento e contexto (por exemplo, *fog*, *mist* e *dew computing*). Ao depender da computação de borda, as aplicações sensíveis ao tempo podem descarregar suas tarefas do dispositivo móvel para um servidor de borda mais potente, sendo esse processo chamado comumente de *offloading*. Isso permite que o dispositivo móvel conserve sua vida útil da bateria e recursos computacionais, enquanto aproveita as capacidades superiores de processamento do servidor de borda. O descarregamento de computação de aplicações e tarefas executadas em dispositivos móveis para servidores de borda tem sido realizado principalmente através de redes Wi-Fi ou celulares (por exemplo, 4G).

Com o advento da *Fifth Generation of Mobile Networks* (5G) e sua implantação por operadoras de telecomunicações em cenários externos e, mais recentemente, como redes privativas 5G em indústrias, cidades e ambientes corporativos, as aplicações em tempo real podem se beneficiar da baixa latência, alta largura de banda e confiabilidade suportadas por esta nova geração de redes celulares. No entanto, mesmo com as melhorias da interface aérea do 5G em relação às gerações anteriores, manter o processamento na nuvem remota ainda pode ser prejudicial para aplicações em tempo real. Assim, as operadoras de telecomunicações também devem adotar o processamento de aplicações próximo ao usuário final, adotando abordagens de computação de borda. Para esse fim, o *European Telecommunications Standards Institute* (ETSI) (European Telecommunications Standards Institute (ETSI), 2024) padronizou o *Multi-access Edge Computing* (MEC) (ETSI MEC ISG, 2024).

A infraestrutura conjunta 5G-MEC parece uma abordagem promissora não apenas para otimizar o tempo de resposta para aplicações sensíveis, mas também para permitir processamento remoto eficiente, proporcionando uma experiência mais ágil e responsiva para os usuários da rede.

Com o aumento no número de operadoras móveis e provedores de serviços com acesso 5G, é de suma importância elaborar soluções para suportar aplicações em tempo real com a assistência da computação de borda. Além disso, plataformas baseadas em código aberto para MEC e núcleo 5G podem ser implantadas para prototipagem rápida e teste de aplicações (Ren *et al.*, 2019).

1.2 OBJETIVOS

O objetivo geral deste trabalho é fornecer uma solução fim a fim composta por núcleo 5G e MEC *open-source* em conjunto com um rádio 5G proprietário e realizar a integração de todas essas plataformas para a realização de testes de desempenho de diferentes cenários para o *offloading* computacional, sendo estes baseados em plataforma MEC ou nuvem computacional remota. Visa-se, também, a implementação de uma aplicação baseada em visão computacional para dispositivos móveis para avaliar os cenários utilizando métricas de qualidade de serviço. Para chegar ao objetivo geral, alguns objetivos específicos devem ser alcançados:

1. Elaborar e implementar uma arquitetura 5G-MEC que integre e compatibilize plata-

formas de código aberto para o núcleo 5G e MEC, bem como utilizar na proposta um rádio proprietário na parte da rede de acesso. Dessa forma, será concebido um ambiente de computação de borda 5G para auxiliar no processamento remoto de aplicações de visão computacional.

2. Desenvolver uma aplicação para análise de sentimentos baseada em visão computacional e offloading de carga. Encontrando um algoritmo que melhor se encaixam para a análise da arquitetura proposta.

3. Caracterizar as métricas que precisam ser avaliadas em um teste de *offloading* real e identificar quais parâmetros influenciam essas métricas.

4. Analisar quantitativamente a eficiência das plataformas de *offloading* remoto e comparar os resultados obtidos em cada arquitetura.

1.3 CONTRIBUIÇÕES DO TRABALHO

Com base no que foi exposto, as principais contribuições deste trabalho incluem:

1. Concepção e implementação de um ambiente 5G com processamento offloading na borda, considerando diferentes plataformas de código aberta para o núcleo da rede e para a plataforma MEC em conjunto do rádio proprietário para a rede de acesso.

2. Implementação de uma aplicação que captura imagens do ambiente e realiza processamento remoto de algoritmos de visão computacional enquanto captura métricas da rede e, em seguida, retorna o resultado dos algoritmos.

3. Implementação de uma aplicação Android que envie os quadros (do inglês, *frames*) da câmera para o processamento remoto tanto para plataforma MEC como para plataforma *Cloud*.

1.4 ORGANIZAÇÃO DO DOCUMENTO

Este projeto foi estruturado em capítulos conforme a seguinte organização:

- **Capítulo 2** - Apresenta os fundamentos teóricos do estudo, realiza a conceituação dos elementos para o entendimento da referida proposta de projeto de conclusão de curso;

- **Capítulo 3** - Traz os estudos relacionados às soluções de offloading com servidores de borda e de nuvem.

- **Capítulo 4** - Retrata o ambiente e a soluções propostas, incluindo módulos, serviços, bibliotecas, implementação e funcionamento;

- **Capítulo 5** - Aborda os resultados das medições e capturas de métrica das arquiteturas de rede propostas;
- **Capítulo 6** - Sumariza a pesquisa e os resultados e manifesta os trabalhos futuros

2

FUNDAMENTAÇÃO

Esta seção fornece uma visão geral dos conceitos básicos de visão computacional e os blocos de construção relacionados ao design e implementação da infraestrutura 5G-MEC proposta. Primeiro, são apresentados uma visão geral de visão computacional, em seguida é apresentado os fundamentos do 5G, e, por fim, a arquitetura MEC do ETSI é descrita.

2.1 VISÃO COMPUTACIONAL

A visão computacional é um campo da Inteligência Artificial (IA) que permite aos computadores extrair informações específicas de imagens e vídeos, e a partir dessa análise, é possível atuar de diferentes formas. Essa tecnologia abrange algoritmos, desde técnicas tradicionais de processamento de imagem até *Convolutional Neural Network* (CNN) e outras formas de aprendizado profundo (Paneru & Jeelani, 2021).

Os algoritmos de visão computacional incluem detecção de objetos, reconhecimento facial, segmentação de imagens, rastreamento de movimento e reconhecimento de emoções, entre outros. As áreas de aplicação também são bastante diversas, incluindo saúde com diagnóstico assistido por imagem e monitoramento de pacientes (Ling *et al.*, 2022); segurança com vigilância, controle de acesso e sistemas de reconhecimento facial (Motlagh *et al.*, 2017); indústria com inspeções de qualidade e automação de processos (Rasheed *et al.*, 2020); automotivo com carros autônomos e assistência ao motorista (Janai *et al.*, 2020); e entretenimento com realidade aumentada e virtual para jogos.

A visão computacional é importante e efetiva, pois, apesar das técnicas de processamento de informações visuais não serem um tópico recente, elas exigiam um certa intervenção humana, tornando o processo ser mais lento e propenso a erros. Era necessário que desenvolvedores marcassem manualmente *datasets* de milhares de imagens com pontos e dados importante e relevantes na imagem para treinar algoritmos de reconhecimento facial, por exemplo (Amazon Web Services, 2023).

Apesar de haver vários algoritmos de detecção de informações em uma imagem, as etapas que uma aplicação de visão computacional permanece a mesma ou muito semelhante para todos esses algoritmos (Figura 1) (Babcock, 2021)(Verre, 2021). Sendo elas:

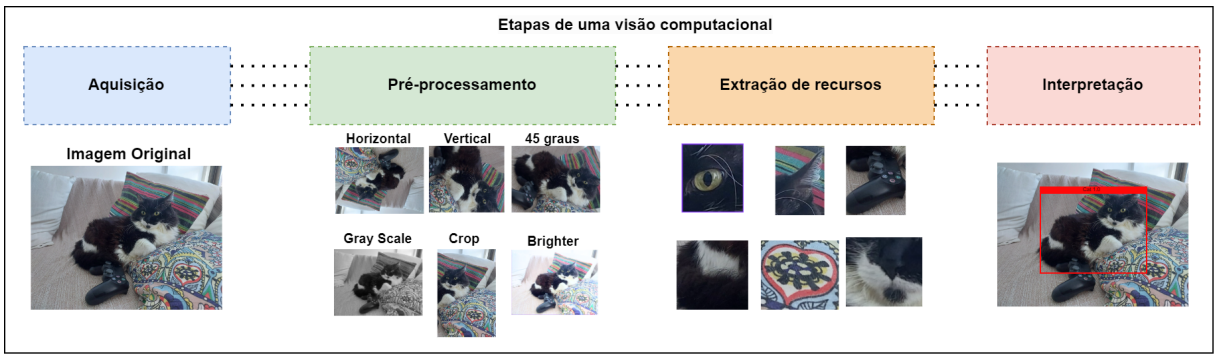

Figura 1: Etapas de uma aplicação genérica de visão computacional.

1. **Aquisição** Essa etapa é a captura de imagens. Esta pode ser realizada por meio de câmeras digitais, scanners, ou outros dispositivos de captura de imagem. A qualidade da imagem adquirida é crucial para o sucesso das etapas subsequentes.

2. **Pré-processamento de Imagens** Antes de qualquer análise, as imagens passam geralmente por uma fase de pré-processamento. O desenvolvedor escolhe o tipo de processamento, mas esta etapa inclui operações como remoção de ruído, ajuste de contraste, redimensionamento da imagem e equalização de histograma. O objetivo é melhorar a qualidade da imagem e/ou realçar características importantes.

3. **Extração de recursos** Esta etapa utiliza os algoritmos para a extração de informações da imagem preprocessada. As informações extraídas são arestas, bordas, formas e padrões de texturas encontradas na imagem.

4. **Interpretação, análise ou saída** Essa etapa consiste em análise dos recursos encontrados, modelos treinados em identificar objetos, pessoas, animais dentre outras coisas utilizam as informações extraídas para indicar a localização das classes detectadas dentro da imagem.

Conforme mencionado anteriormente na seção 1.1, as aplicações de visão computacional são geralmente executadas localmente nos dispositivos. No entanto, devido ao alto consumo de bateria (Goel *et al.*, 2020) e à limitada capacidade de computação dos dispositivos móveis, houve uma mudança para processar essas aplicações remotamente usando servidores na nuvem. Embora essa abordagem resolva alguns problemas de capacidade de processamento e consumo de energia, ela introduz o desafio da latência, que pode afetar negativamente a experiência do usuário em aplicações que exigem respostas em tempo real. Reduzir a latência é crucial, especialmente em aplicações que requerem interações rápidas, onde atrasos perceptíveis podem causar desconforto e comprometer a experiência do usuário (Morín *et al.*, 2022)(Mao *et al.*, 2017).

Para evitar desconforto, a latência em aplicações de jogos na nuvem deve ser em torno de 60 ms e 100 ms para jogos casuais e com latências ainda menores necessárias para jogos

competitivos (Baena *et al.*, 2024). No entanto, para aplicações em dispositivos móveis, essa latência mínima varia significativamente dependendo da aplicação, pois atrasos ligeiramente maiores podem ser suficientes para determinadas aplicações.

Na elaboração desse trabalho dois algoritmos principais foram utilizados para avaliar o ambiente proposto, sendo eles o *Haar Cascade classifier* e o MTCNN. Cada um possui suas vantagens e desvantagens e serão apresentados a seguir:

2.1.1 *Haar Cascade classifier*

O *Haar Cascade classifier* é um algoritmo de *Machine Learning* (ML) de detecção de objetos que consegue identificar objetos em uma imagem e em vídeos (Viola & Jones, 2001)(OpenCV, 2017). Este algoritmo é conhecido por sua velocidade na detecção de faces em uma imagem e em um vídeo, e por fazer isso em tempo real. O funcionamento do classificador pode ser dividido em cinco etapas, sendo elas:

1. **Características de Haar:** As características de Haar são padrões usados para detectar objetos em imagens, como faces. Elas funcionam ao comparar áreas claras e escuras de uma imagem para identificar bordas, linhas e outras formas simples. Na Figura 2 é possível ver as possíveis características de Haars. Esses quadrados são aplicados sobrepondo a imagem a ser classificada em vários pontos da imagem, comparando se os píxeis da parte branca do retângulo são mais claros que os píxeis que estão na parte preta do retângulo. Ao fazer isso sob a imagem por completo com todos esses classificadores, é possível encontrar características de Haar de arestas, linhas, bordas e centros na imagem. Na Figura 3 é possível ver um exemplo da aplicação dessas características em uma imagem. Encontrar essas características em uma imagem grande pode ser custoso e difícil, e então, faz-se necessária a ideia da "imagem integral".

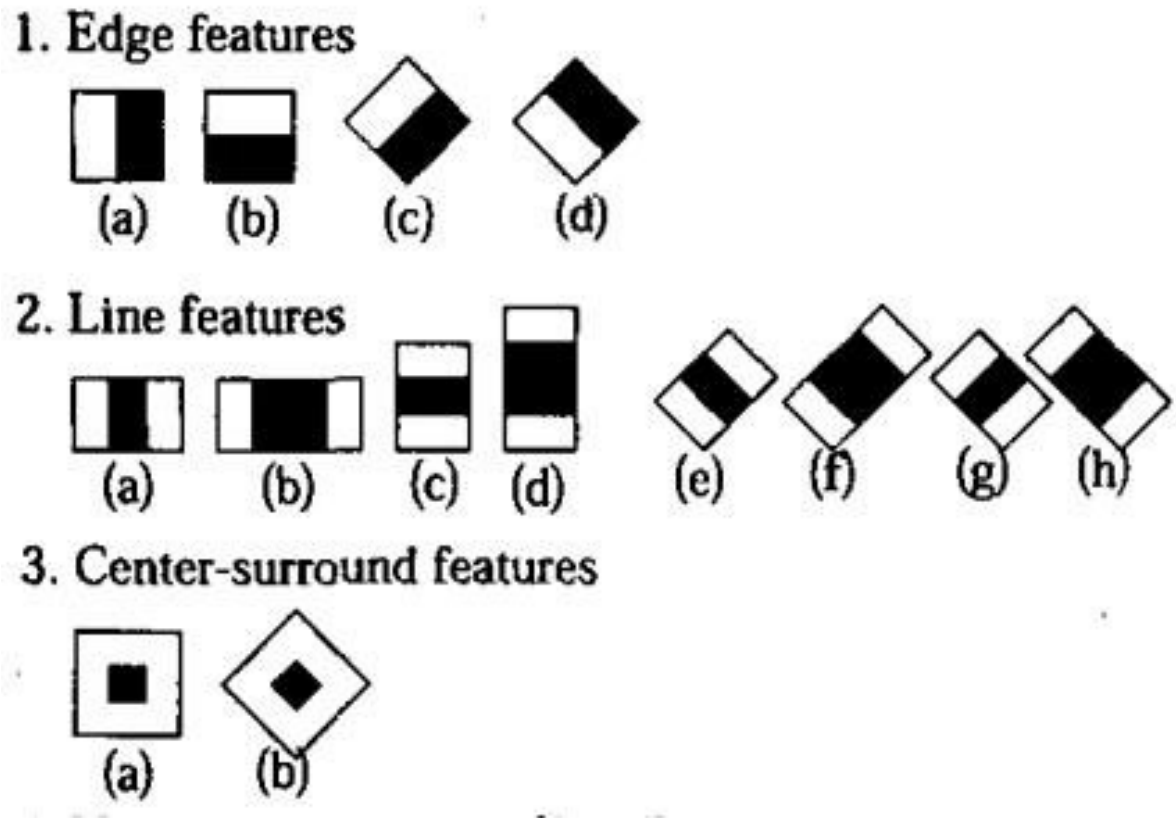

Figura 2: Características Haars possíveis (Yang *et al.*, 2018)

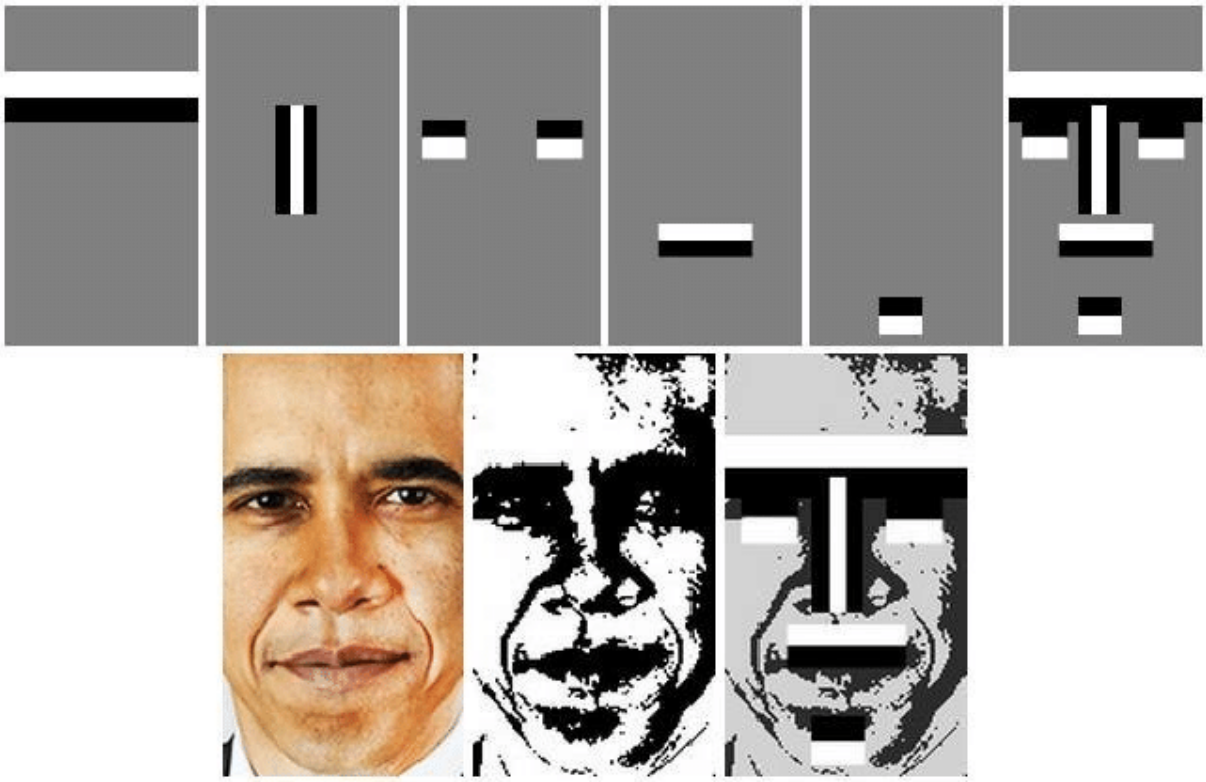

Figura 3: Aplicação das característica Haars em uma imagem (Kadir *et al.*, 2014)

2. **Imagem integral:** A imagem integral é construída a partir da imagem original de forma iterativa. Ela é criada somando todos os píxeis acima e à esquerda de um píxel específico. Com isso, é possível acelerar o cálculo dessas características de Haar. Em vez de calcular em cada píxel, ela cria sub-retângulos e gera referências de matriz para cada um desses sub-retângulos (Figura 4). Essas referências são então usadas para calcular as características de Haar. Quase nenhuma característica de Haar será relevante para a detecção de determinado objeto, sendo assim, é necessário o emprego do algoritmo Adaboost.

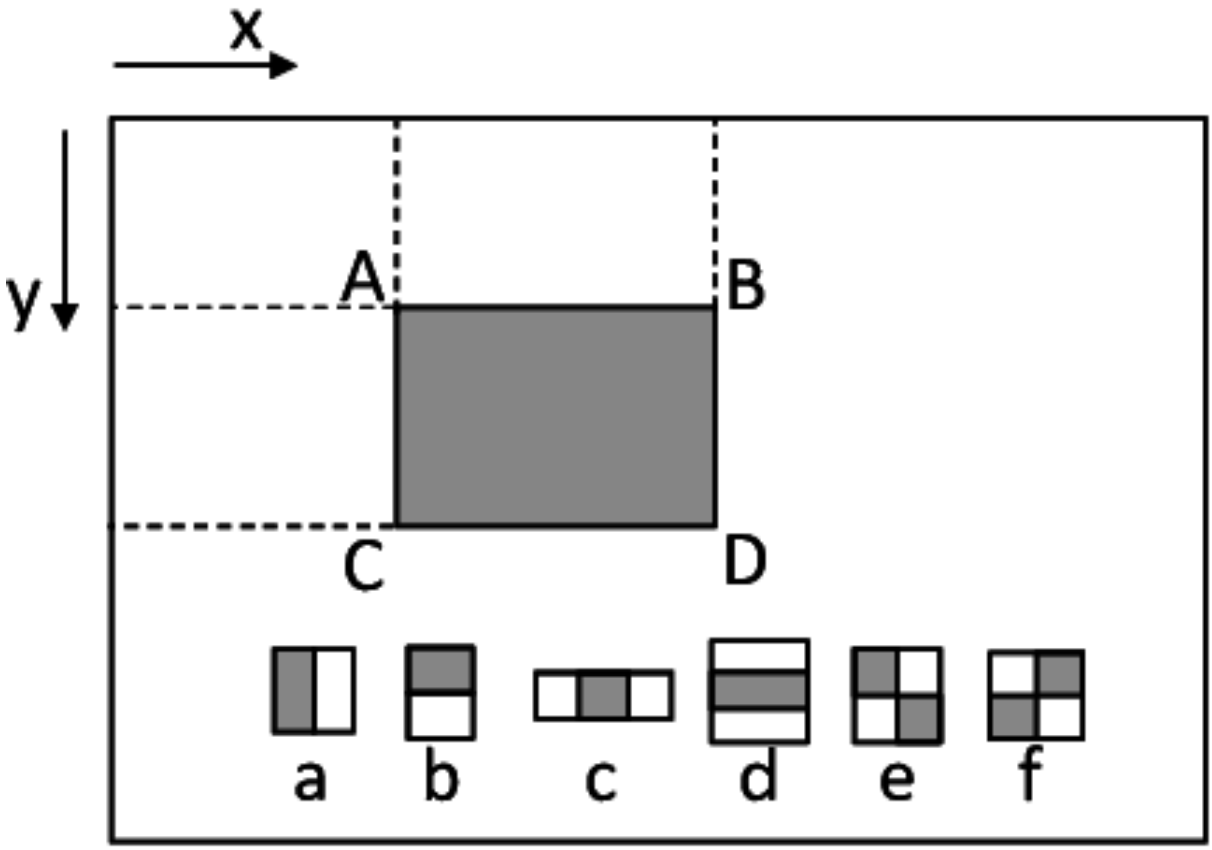

Figura 4: Matriz referência que compõe uma imagem integral (Zhang & Zhang, 2010)

3. **Treinamento com AdaBoost:** O processo de treinamento do classificador utiliza o algoritmo AdaBoost para selecionar as características de Haar mais relevantes a partir de um conjunto enorme de possíveis características. Ele atribui pesos maiores às características que têm melhor desempenho na separação entre objetos e não-objetos. Esse algoritmo utiliza a ideia de combinar múltiplos classificadores "fracos" em um grande classificador "forte", como pode ser visto na Figura 5. Para então, assim, utilizar esses classificadores "fortes" para detectar objetos de não objetos. Para esse treinamento, é necessário ter um *dataset* com muitas imagens negativas e positivas. As positivas são imagens que contêm o objeto a ser detectado e o negativo é o que não tem o objeto a ser detectado.

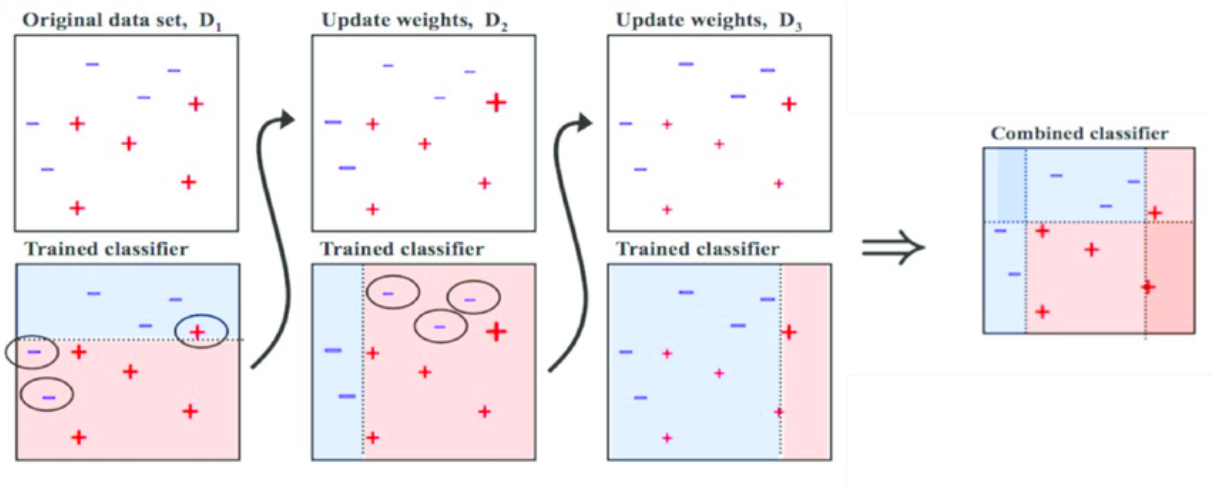

Figura 5: Exemplo de treino adaboosting (Marsh, 2016)

4. **Classificador *Cascade*:** O classificador em cascata é uma técnica usada para acelerar a detecção de objetos em imagens. Ele organiza vários classificadores "fracos" (chamados de "estágios"), que devido ao adaboosting cada estágio possui um classificado "forte", em uma sequência, onde cada estágio é responsável por rejeitar rapidamente uma grande parte das regiões da imagem que não contêm o objeto de interesse, permitindo que apenas as regiões mais promissoras passem para estágios subsequentes. Os estágios subsequente são cada vez mais e mais complexo. Este processo em cascata permite que o classificador seja rápido e eficiente, pois a maioria das sub-janelas são rejeitadas nas primeiras etapas de processamento. (talvez inserir uma imagem de estágios)

5. **Detecção:** Realizando o processo descrito anteriormente é possível criar um *pipeline* do funcionamento do algoritmo (Figura 3), a imagem é analisada em várias escalas e posições usando uma janela deslizante. Para cada posição da janela, as características de Haar são calculadas e passadas através dos estágios do classificador *cascade*. Se a sub-janela passa por todos os estágios, ela é marcada como contendo o objeto de interesse.

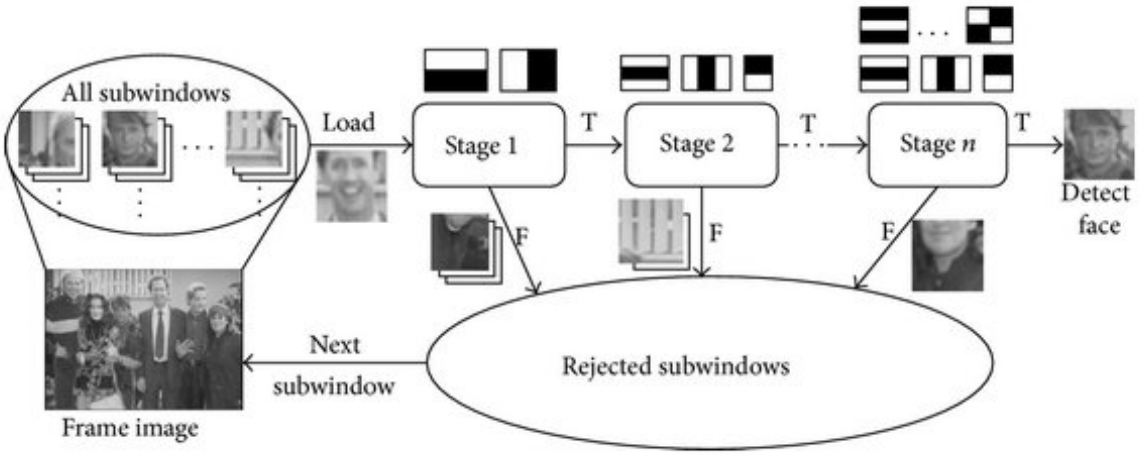

Figura 6: *Pipeline* de representação do algoritmo *Haar Cascade Classifier* (Kim *et al.*, 2015)

O classificador Haar *Cascade* é muito interessante para a utilização em aplicações de *offloading* em tempo real, visto que é um algoritmo leve em termos computacionais e possui uma velocidade de resposta muito rápida. No entanto, ele não demonstra uma precisão tão alta em imagens de baixa qualidade e também não requer um poder computacional tão robusto.

2.1.2 MTCNN

O algoritmo Multi Cascade Convolutional Network (MTCNN) é um modelo utilizado para detecção facial em imagem conhecido pelo seu grande nível de precisão e eficiência (Zhang *et al.*, 2016), em contrapartida, esse modelo requer um recurso computacional elevado em relação ao *Haar cascade*. Este algoritmo combina várias CNN para detectar rostos, localizar o rosto nas imagens e vídeos, encontrar marcos faciais (olhos, nariz, boca, etc.) com uma alta precisão mesmo com os objetos de estudo com diferentes tamanhos e posições na imagem. O MTCNN consiste em uma estrutura de três CNNs em cascata (seguindo a ideia de cascata do *Haar Cascade*). A arquitetura MTCNN e suas etapas são ilustradas na Figura 7:

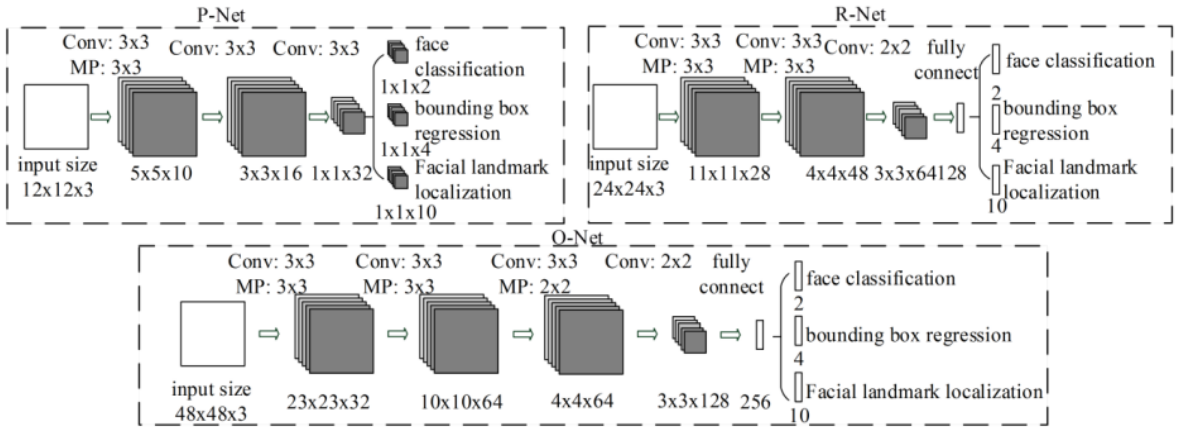

Figura 7: Arquitetura do algoritmo MTCNN e suas etapas (Zhang *et al.*, 2016)

1. ***Proposal Network*** **(P-Net)** A menor imagem, resultado do redimensionamento executado diversas vezes, é enviada para essa primeira etapa. O P-Net visa de gerar propostas iniciais de regiões onde os rostos podem estar localizados. A P-Net aplica uma série de operações convolucionais para extrair características da imagem. As convoluções são realizadas utilizando filtros pequenos, geralmente de tamanho 3x3, que percorrem a imagem para detectar padrões relevantes, as saídas dessas convoluções geram uma matriz menor que a original chamada de mapas de característica. Após cada camada convolucional, P-Net utiliza uma operação de *Max Pooling*, que reduz a dimensionalidade dos mapas de características ao selecionar o valor máximo em cada região de *pooling*. Este processo ajuda a destacar as características mais importantes e a reduzir a complexidade computacional.

   A saída da P-Net inclui uma série de caixas delimitadoras que indicam as regiões propostas onde os rostos podem estar. Cada caixa é acompanhada por uma pontuação que indica a probabilidade de a região conter um rosto. Além disso, a *pnet* determina uma localização inicial dos marcos faciais. Essas propostas iniciais são então passadas para a próxima etapa do pipeline.

2. ***Refine Network*** **(R-Net)** A função principal da segunda etapa é refinar as propostas geradas pela P-Net, eliminando as detecções falsas e ajustando ainda mais as caixas delimitadoras. A R-Net recebe como entrada as regiões propostas pela P-Net e aplica convoluções adicionais para extrair características mais detalhadas. As convoluções na R-Net também utilizam filtros de tamanho 3x3, seguidas por operações de *Max Pooling* para continuar reduzindo a dimensionalidade dos mapas de características.

   Além das camadas convolucionais, essa etapa inclui uma camada totalmente conectada que combina todas as características extraídas detecções anteriormente para tomar decisões finais. Esta camada possui 128 unidades e é responsável por integrar as informações provenientes das camadas anteriores. A saída da R-Net consiste

em classificações mais precisas das regiões de rosto, caixas delimitadoras ajustadas e uma melhor localização dos marcos faciais. A R-Net é projetada para rejeitar rapidamente as regiões que não contêm rostos, melhorando a eficiência do processo.

3. ***Output Network* (O-Net)** Esta última etapa realiza a detecção final dos rostos, proporcionando uma precisão ainda maior na localização e no ajuste das caixas delimitadoras, bem como na identificação dos marcos faciais. A O-Net recebe as propostas refinadas da R-Net e aplica convoluções adicionais para uma análise mais detalhada. As camadas convolucionais na O-Net também utilizam filtros 3x3, seguidas por operações de *Max Pooling*, e uma camada totalmente conectada com 256 unidades para combinar as características extraídas.

   A saída da O-Net inclui uma classificação final da presença de rostos, caixas delimitadoras ajustadas com precisão e uma localização exata dos marcos faciais. A combinação dessas três redes permite que o MTCNN forneça detecções de rosto altamente precisas e robustas, mesmo em condições desafiadoras, como variações de iluminação, poses e expressões faciais. A aplicação de todas essas etapas em uma imagem pode ser vista na Figura 8.

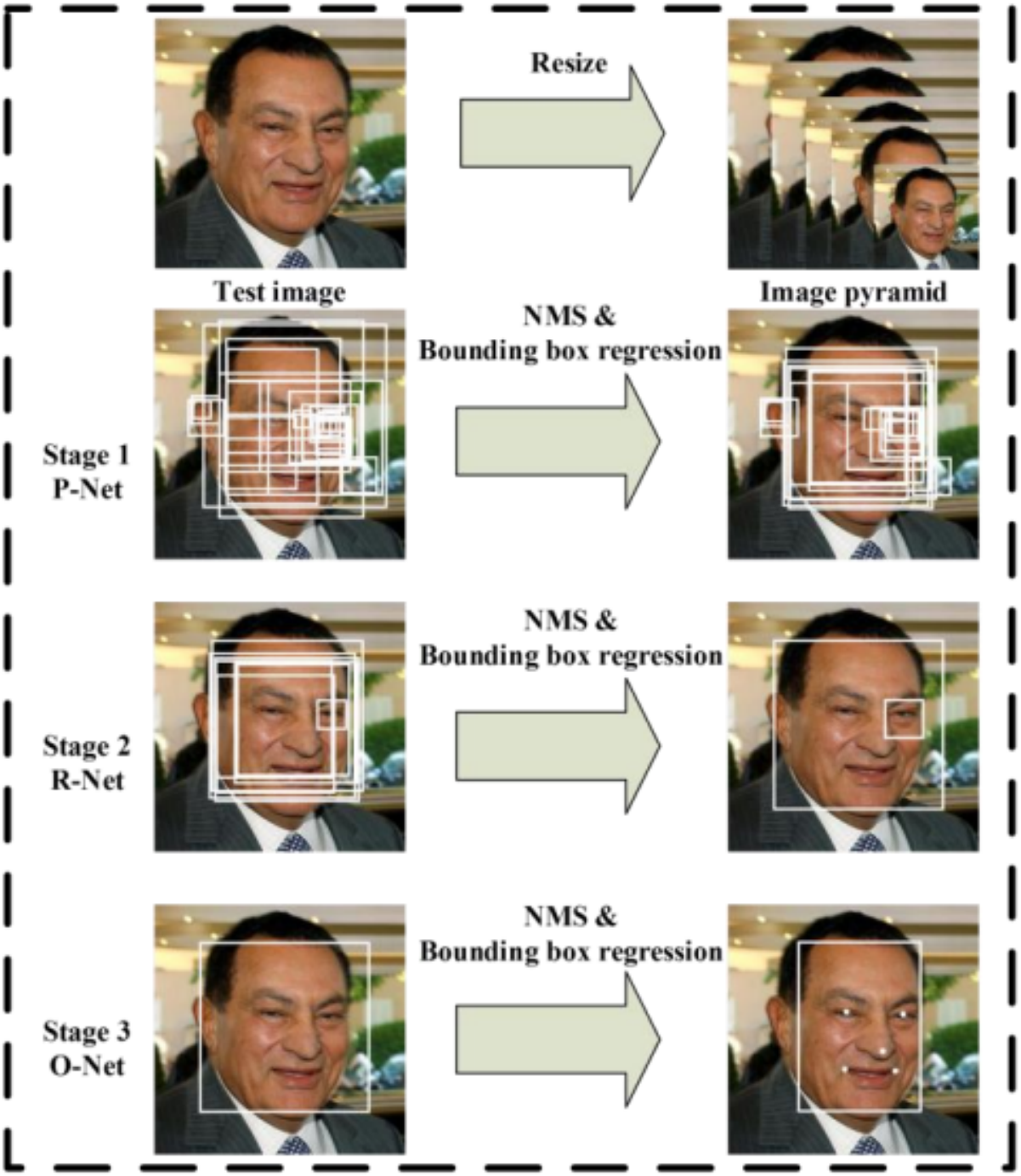

Figura 8: *Pipeline* da aplicação das redes de convolução do algoritmo MTCNN (Zhang *et al.*, 2016)

Em comparação ao algoritmo Haars *cascade*(sessão 2.1.1, o MTCNN requer um poder computacional maior, levando a maior tempo de processamento, e também tendo resultados mais precisos mesmo em situações mais adversas como com baixa iluminação ou imagens e baixa resolução. Sendo assim, esse algoritmo foi selecionado para a realização de testes das arquiteturas propostas.

2.2 5G

As evoluções das redes móveis são impulsionadas pela necessidade de avanços tecnológicos que atendam às demandas da sociedade (Gupta & Jha, 2015). Os principais avanços estão relacionados à melhoria da conectividade, maiores taxas de transferência, mais escalabilidade e a capacidade da rede de comportar o aumento na quantidade de dispositivos conectados, os quais têm constantemente aumentado ao longo das décadas. Cada geração trouxe melhorias significativas, preparando o caminho para a inovação contínua e o desenvolvimento de novas tecnologias. É possível acompanhar esses avanços a partir da Figura 9.

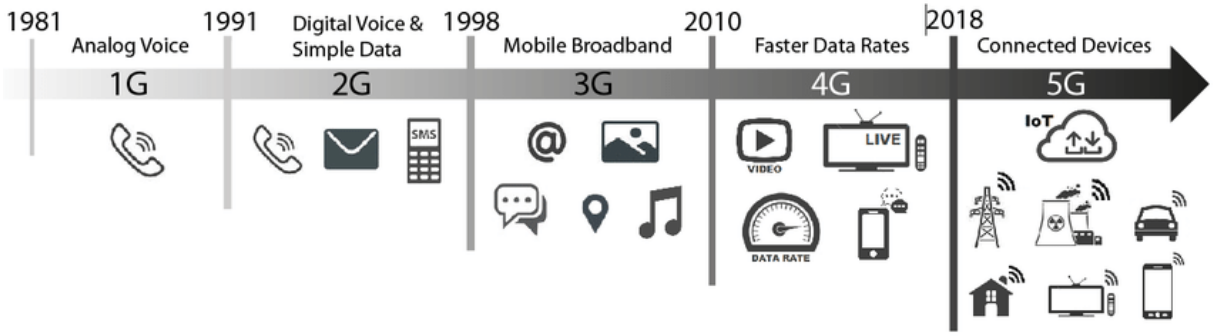

Figura 9: Evolução progressiva das tecnologias de redes móveis da primeira até a quinta geração. (Guevara & Auat Cheein, 2020)

Essa evolução é guiada por padrões e requisitos rigorosos estabelecidos por organizações internacionais.

A primeira geração das redes móveis, do inglês *First Generation of Mobile Networks* (1G) foi introduzido na década 80 e marca a era dos sistemas de telefonia analógica de transferência de voz. Na década de 90 foi introduzido a tecnologia digital com a segunda geração das redes móveis (*Second Generation of Mobile Networks* (2G)) trazendo avanços significativos na qualidade das transmissões de voz e redução de interferências, na segurança e trouxe também as mensagens de texto *Short Message Service* (SMS). Contudo, o maior salto na evolução das redes móveis em termos de conectividade e transferência de dados se deu no início dos anos 2000 com a chegada da terceira geração de redes móveis (*Third Generation of Mobile Networks* (3G)).

O 3G trouxe um aumento considerável nas taxas de *download* e *upload* em relação ao 2G, com taxas de até 2 *Mega bits per second* (Mbps), possibilitando o uso de internet móvel. Além disso, o 3G introduziu a capacidade de realizar videochamadas, acessar a *Web* e utilizar serviços de multimídia de *streaming* de áudio e vídeo. A quarta geração das redes móveis, do inglês *Fourth Generation of Mobile Networks* (4G), teve sua padronização *3rd Generation Partnership Project* (3GPP) na *release* 8, em 2008, com o LTE, com posteriores melhorias em *releases* subsequentes com o LTE-*Advanced* e LTE-*Advanced-Pro*. As redes 4G surgiram com o intuito de trazer melhorias às características introduzidas 3G com aumento de velocidade de dados, a baixa latência, a introdução da capacidade de *streaming* de vídeos e também o aumento

significativo na capacidade de conexão de dispositivos simultâneos.

Apesar dos avanços significativos, as demandas futuras exigiam o desenvolvimento de uma nova geração de redes móveis. Embora o 4G tenha melhorado a velocidade de dados em relação ao 3G, as aplicações em tempo real e projeções futuras demandam ainda maiores taxas de *download* e *upload*. Além disso, era necessário reduzir a latência para suportar aplicações que requerem respostas imediatas, como carros autônomos e realidade aumentada. O 5G foi projetado para gerenciar inúmeras conexões simultâneas e otimizar o consumo de energia dos dispositivos, tornando-o mais adequado para a expansão da Internet das Coisas (IoT). Com velocidades de dados significativamente maiores, latência ultra baixa e maior eficiência energética, o 5G facilita inovações tecnológicas e aplicações emergentes.

Entender como funciona a arquitetura 5G é importante para compreender os componentes 5G envolvidos nesse TCC. A arquitetura da rede 5G é mais flexível e escalável, permitindo a implantação de novas tecnologias e serviços de maneira mais eficiente. Além disso, o 5G utiliza de frequências que vão desde as bandas de baixa frequência (sub-1 GHz) até as de alta frequência (*Millimeter Wave* (mmWW)), permitindo um equilíbrio entre cobertura e capacidade. A arquitetura da rede 5G (Figura 10) pode ser dividida em dois componentes principais: Rede de Acesso via Rádio 5G (*Radio Access Networks* (RAN)) e a Rede de Núcleo 5G (*5G Core Network* (5GC)).

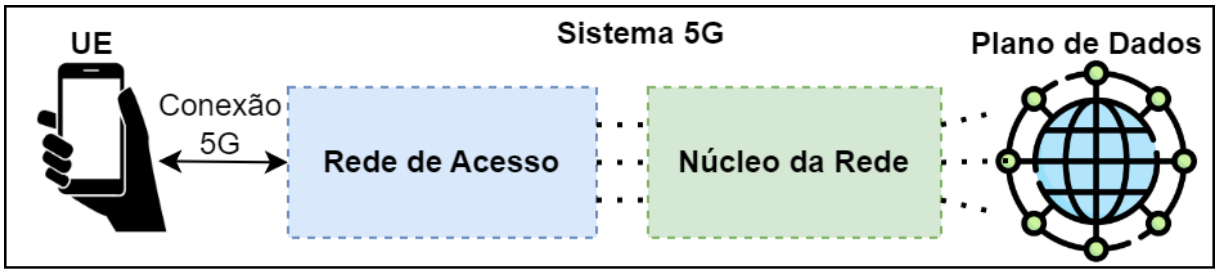

Figura 10: Arquitetura 5G alto nível

2.2.1 5G RAN

A RAN é a parte da infraestrutura de rede que conecta dispositivos móveis ao núcleo rede. Ela facilita a comunicação sem fio entre os dispositivos dos usuários e a rede, permitindo a transmissão e recepção de dados. Isso é realizado por meio de estações-base, chamadas nas redes 5G de *Next Generation Node B* (gNB) , as quais são compostas de antenas e equipamentos de rádio. A RAN utiliza uma ampla gama de frequências, incluindo bandas de baixa (sub-1 GHz), média (1-6 GHz) e alta frequência (mmWave, acima de 24 GHz). Cada uma dessas bandas oferece diferentes vantagens e desafios.

A arquitetura da RAN 5G é caracterizada pela flexibilidade e capacidade de virtualização, permitindo que as operadoras adaptem e otimizem recursos de forma dinâmica conforme a demanda de tráfego e serviços.

Além disso, a RAN incorpora tecnologias avançadas, como *Multiple Input Multiple Output* (MIMO) e *beamforming*, que melhoram a eficiência espectral e a capacidade da rede. O *Massive* MIMO permite que várias antenas transmitam e recebam dados simultaneamente, melhorando a cobertura e a capacidade, enquanto o *beamforming* direciona o sinal de rádio de forma precisa para dispositivos específicos, aumentando a eficiência energética e melhorando o desempenho da rede em áreas densamente povoadas.

2.2.2 Rede de Núcleo 5G

O 5GC é a parte central da infraestrutura de uma rede de telecomunicações, responsável por gerenciar a conexão, roteamento e entrega de dados entre dispositivos móveis e a internet. Ele é significativamente diferente das gerações anteriores devido à sua flexibilidade e eficiência. A rede central do 5G é projetada para ser totalmente baseada em software, o que facilita a implementação de funções de rede virtualizadas e o uso de *Software Defined Network* (SDN), uma abordagem que separa o plano de controle, responsável pela tomada de decisões sobre o tráfego, do plano de dados, que movimenta o tráfego. Isso é feito para permitir uma gestão centralizada e flexível da rede (Kreutz *et al.*, 2015).

Uma das principais inovações do núcleo da rede 5G é a organização de suas funções de rede em uma *Service Based Achitecture* (SBA), uma abordagem de arquitetura de redes que organiza a infraestrutura de telecomunicações em serviços modulares e independentes que se comunicam entre si por interfaces bem definidas (Moreira *et al.*, 2020). Cada função de rede pode ser acessada e compartilhada por meio de APIs padronizadas. Elas podem ser implementadas, escaladas e atualizadas de forma independente, proporcionando maior flexibilidade e eficiência operacional. Essa abordagem permite uma integração mais fácil de novos serviços e a capacidade de adaptar rapidamente a rede às mudanças nas demandas dos usuários e nos requisitos de serviço. Além disso, a SBA facilita a implementação de segurança avançada e gerenciamento de recursos de rede, garantindo a entrega confiável e eficiente de serviços críticos. A rede 5G possui diversas funções de rede (*Network Function* (NF)) que possuem cada um funções distintas como *Network Function* (AMF) para gerência a interação do usuário com a rede, *Session Management Function* (SMF) Gerencia sessões de dados, *User Plane Function* (UPF) Lida com o plano de usuário para encaminhamento de dados, etc. Na Figura 11 é possível ver essas NFs na arquitetura da rede e a comunicação entre elas, ilustrando como o SBA funciona em redes 5G.

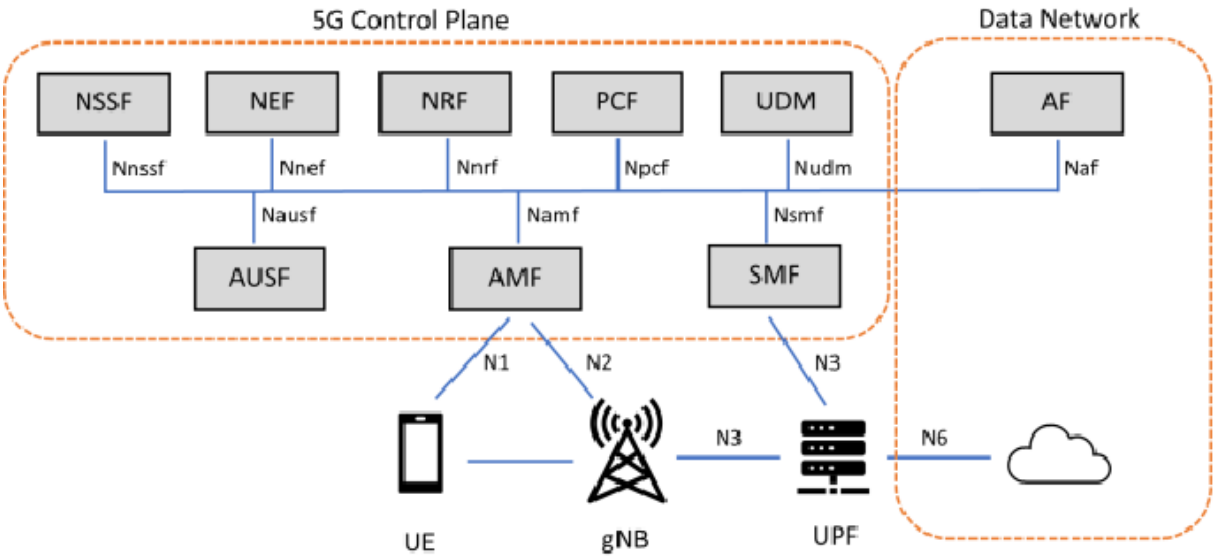

Figura 11: Arquitetura 5G baseada em serviço. (Bartolín-Arnau *et al.*, 2024)

2.3 PLATAFORMAS DE CÓDIGO ABERTO

Plataformas de código aberto como OpenAirInterface (OAI) (OpenAirInterface, 2024) e Open5GS (Open5GS, 2024) desempenham um papel crucial no desenvolvimento de redes móveis 5G. Elas oferecem uma alta relação custo-benefício e, devido à SBA, é possível implementar toda a rede usando diferentes plataformas de código aberto. Essas ferramentas de código aberto apoiam a criação de redes 5G escaláveis e eficientes, promovendo inovação e experimentação contínuas em ambientes de redes móveis. Elas permitem estudos e testes de novas arquiteturas propostas, orquestradores e aplicações de gerenciamento de rede (Mihai *et al.*, 2022).

2.4 ETSI MEC

O padrão MEC (ETSI MEC ISG, 2024) desenvolvido pelo ETSI (European Telecommunications Standards Institute (ETSI), 2024) estende as capacidades de computação em nuvem até a borda da rede celular, ela surgiu como uma evolução natural para atender à crescente demanda por latência ultrabaixa e alta largura de banda em aplicações modernas. O MEC é um sistema que traz os recursos de computação, armazenamento e rede mais próximos dos usuários e dos dispositivos finais, reduzindo a latência e melhorando a eficiência da rede (Nencioni *et al.*, 2023).

O MEC é uma tecnologia chave para a computação de borda no 5G, que promete suportar novos tipos aplicações, desde a IoT até a Realidade Aumentada (AR),*Virtual Reality* (VR) e veículos autônomos. A combinação do MEC com a infraestrutura 5G permite que os serviços sejam entregues com maior velocidade e confiabilidade, conceitos essenciais para aplicações em tempo real.

Essa proposta funciona através da implantação de servidores de borda nas proximidades

dos usuários finais. Esses servidores podem executar funções de rede virtualizadas (VNFs) e aplicativos de usuário, processando dados localmente. Um sistema MEC é composto por um conjunto de vários MEH. A Figura 12 ilustra arquitetura de referência geral MEC ETSI, ela e dividida em 2 níveis principais: o nível MEH e nível MEC *System*.

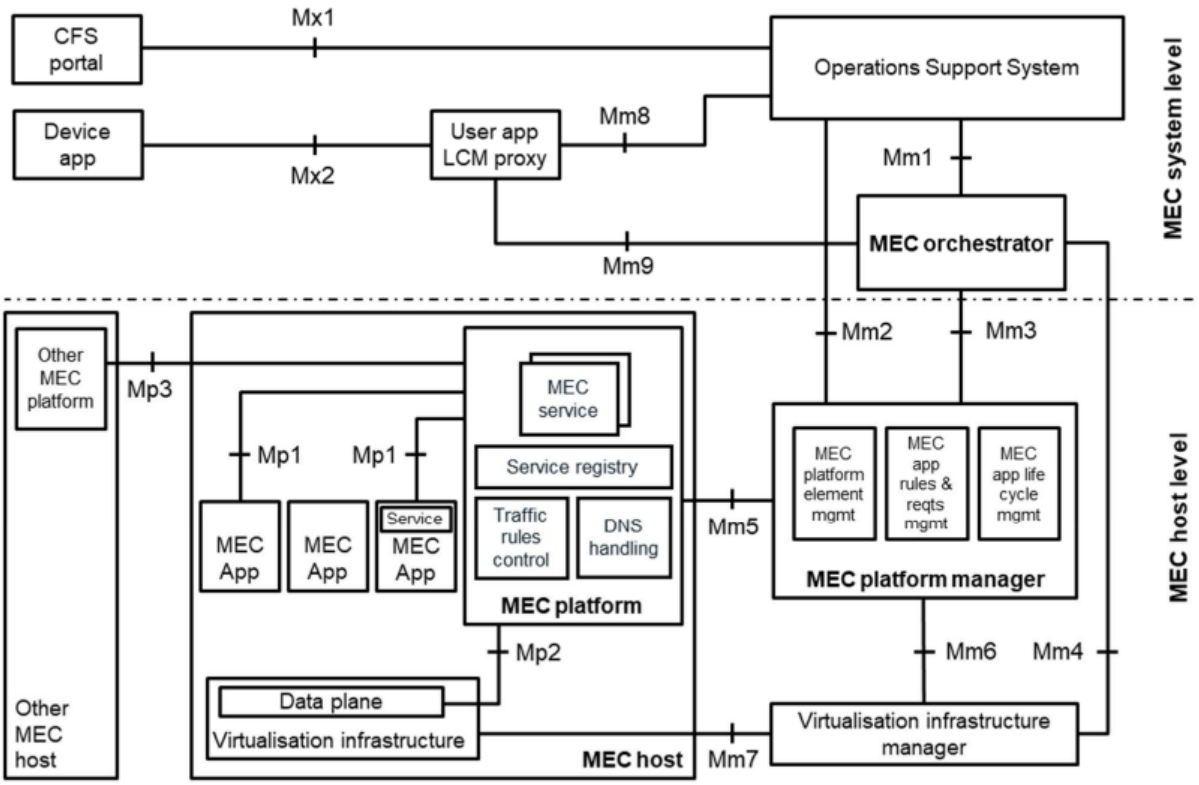

Figura 12: Arquitetura MEC Etsi (Borcoci *et al.*, 2018)

2.4.1 Nível MEH

Cada MEH possui um host virtualizado para fornecer recursos de computação, armazenamento e rede para executar Aplicação MEC (MEC App) e Plataforma MEC (MEP).

A MEP fornece o ambiente de execução para aplicações de borda. Ela gerencia recursos de computação e armazenamento e oferece serviços de suporte como orquestração e gerenciamento de aplicações. As aplicações podem utilizar serviços MEC existentes ou registrar novos serviços MEC. A plataforma exibe centralmente os serviços disponíveis e seus *endpoints* de forma padronizada. As MEC App são aplicações desenvolvidas para executar no MEH e utilizar sua proximidade aos usuários finais para fornecer serviços com baixa latência. Essas aplicações podem ter vários objetivos, funcionando essencialmente como APIs que visam explorar as vantagens da localização na borda, podem também consumir ou fornecer serviço MEC. As MEC App devem registrar seus serviços na MEP para serem acessíveis aos usuários da rede e outras aplicações MEC. O gerenciamento do MEH é composto pelo MEC Platform Manager (MEPM) e pelo *Virtualization/Virtualized Infrastructure Manager* (VIM). O MEPM gerencia o ciclo de vida

das aplicações MEC, regras e requisitos, enquanto o VIM gerencia a alocação e monitoramento dos recursos virtuais, transmitindo relatórios de falhas e desempenho ao MEPM.

2.4.2 Nível MEC *System*

Este nível é composto por *MEC Orchestrator* (MEO), *Operator's Operations Support System* (OOS) e o proxy de *Life-Cycle Management* (LCM). O MEO é o componente principal desse nível, e supervisiona todo o sistema MEC. Esse componente gerencia pacotes de aplicação, verifica integridade e conformidade, seleciona os MEH apropriados para a instanciação de aplicações MEC e aciona a instanciação, terminação e relocação das aplicações conforme necessário. O OOS recebe solicitações de MEC App via proxy de LCM de aplicação do usuário, permitindo que clientes selecionem e solicitem aplicações MEC ou recebam informações de nível de serviço sobre as aplicações provisionadas.

3

TRABALHOS RELACIONADOS

Este capítulo apresenta o estado da arte em termos de uso de plataformas MEC e Redes 4G/5G para implantações críticas que requerem processamento em tempo real, como visão computacional e realidade aumentada. A Tabela 1 resume os trabalhos relacionados, destacando a camada em que os dados são processados, o tipo de aplicação utilizada, o ambiente de avaliação e o objetivo do estudo.

Tabela 1: Resumo de trabalhos relacionados

Artigo	Camada	Tipo de aplicação	Ambiente	Objetivo
Araújo *et al.* (2023)	MEC	Visão Computacional	*Testbed*	Processamento remoto para cenários de *Vehicle-to-everything* (V2X)
Motlagh *et al.* (2017)	MEC / Local	Visão Computacional	*Testbed*	Análise de desempenho e eficiência energética em Veículo Aéreo Não Tripulado (VANTs)
Ren *et al.* (2019)	Local / Nuvem / Borda	Realidade Aumentada	Simulado	Análise de desempenho e eficiência energética em aplicativos de AR
Este Projeto	MEC / Nuvem	Computer Visions	*Testbed* 5G	Processamento remoto de algoritmos de visão computacional com MEC e 5g

O trabalho apresentado por (Araújo *et al.*, 2023) demonstra um protótipo para detecção de objetos em Comunicação Cooperativa de Veículo-para-Tudo (CV2X) usando dados de vídeo. A demonstração utiliza o núcleo 5G da Capgemini (Capgemini, 2023) acoplado a um servidor de borda para alcançar comunicação de baixa latência. O sistema utiliza o algoritmo de detecção de objetos YOLO em servidores de borda para processar vídeos capturados por veículos em um ambiente 5G. Embora a plataforma proprietária utilizada na demonstração ofereça vantagens, ela não realiza uma avaliação de desempenho quantitativa e carece de uma análise comparativa com abordagens alternativas, incluindo computação em nuvem.

A proposta apresentada por (Motlagh *et al.*, 2017) investiga o potencial de Veículos Aéreos Não Tripulados (VANTs) equipados com dispositivos IoT para vigilância de reconhe-

cimento facial em altitudes elevadas. O estudo demonstra a viabilidade de transferir tarefas de processamento de dados de vídeo para servidores de computação de borda móvel (MEC), reduzindo o consumo de energia e o tempo de processamento a bordo dos VANTs. O trabalho apresenta um protótipo para reconhecimento facial utilizando o algoritmo de Histograma de Padrão Binário Local (LBPH) do OpenCV em um ambiente de teste. Os resultados mostram que a transferência para MEC economiza energia e reduz o tempo de processamento de reconhecimento facial. No entanto, a pesquisa está limitada a ambientes de teste 4G, o que restringe a exploração do verdadeiro potencial do processamento MEC de baixa latência em ambientes 5G.

Com base no estado-da-arte apresentado, nenhum artigo na literatura atual foi encontrado que considere de forma conjunta redes 5G, visão computacional e integração de MEC/Nuvem em um *testebed* real. Além disso, nossa proposta se beneficia de um rádio 5G de alto desempenho, enquanto reduz os custos gerais da implantação usando soluções de código aberto tanto para MEC quanto para o núcleo 5G, e também a viabilidade de integrar um componente proprietário com componentes de código aberto em uma única infraestrutura fim a fim. Portanto, a abordagem proposta apresentada na última linha da Tabela 1 utiliza a arquitetura MEC e Nuvem na camada de processamento, empregando aplicações de visão computacional por meio de um *testbed* 5G real.

4

PROPOSTA

Esta seção apresenta os componentes arquiteturais e a aplicação utilizada para capturar métricas de rede e validar a arquitetura MEC proposta em comparação com a arquitetura em nuvem. O projeto foi desenvolvido seguindo dois tópicos principais: a implementação de uma aplicação para análise de sentimentos e o design de um ambiente integrado 5G-MEC para processamento de aplicações de visão computacional.

4.1 AMBIENTE INTEGRADO 5G-MEC PARA PROCESSAMENTO DE APLICAÇÕES DE VISÃO COMPUTACIONAL

O ambiente proposto é construído utilizando plataformas de código aberto para o núcleo 5G e MEC, oferecendo a flexibilidade e as capacidades necessárias para a interoperabilidade com o rádio proprietário de alto desempenho. Integrar o 5G com a MEP nos permitirá explorar os benefícios de baixa latência, alta largura de banda e confiabilidade oferecidos por esta nova geração de redes celulares. A arquitetura proposta incluirá a implantação de servidores MEC próximos aos usuários finais para processamento na borda, reduzindo a necessidade de transmitir dados para a plataforma em nuvem e, consequentemente, diminuindo a latência. O uso de plataformas de código aberto garante replicabilidade, facilitando implementações e expansões futuras.

A Figura 13 ilustra a arquitetura fim a fim proposta. As especificações de canda componente de alto nível utilizado na arquitetura estão especificadas na Tabela 2. Para a rede 5G, foi utilizada a plataforma de código aberto open5GS para o núcleo da rede e uma solução proprietária, o Huawei BBU 5900, para a rede de acesso. A integração do ambiente de núcleo virtualizado com a plataforma proprietária foi realizada através da interface N2 descrita no padrão 3GPP. Esta interface permite a comunicação entre a RAN e o núcleo da rede, e o N2 padroniza o formato das mensagens trocadas. Ao realizar as devidas configurações de parâmetros da rede 5G tanto no núcleo como na RAN, é possível realizar a comunicação de forma padronizada desses componentes.

Para o componente ETSI MEC, a proposta adotou a plataforma de código aberto OAI. A aplicação MEC, desenvolvida utilizando o *framework* Flask (Armin Ronacher, 2010) da

linguagem Python (Python Software Foundation, 2024), foi registrada na plataforma MEC ao ser inicializada, tornando todos os seus *endpoints* disponíveis para os usuários. A arquitetura também integra funcionalidades de SDN e SBA, permitindo uma implantação flexível e escalável de serviços MEC. Ainda na Figura 13, o núcleo da rede tem uma interface direta com a plataforma MEP chamada Mp2. Ela permite que o MEP se comunique diretamente com o núcleo, com menos saltos do que teria se acessado via internet. Também possibilita que os serviços MEC acessem informações relacionadas ao núcleo de forma mais direta. Adicionalmente, a interface Mp1 fornece a comunicação entre o MEP e as MEC App. Esta arquitetura reduz o número de saltos para que os pedidos de aplicação do cliente acessem os serviços fornecidos pelo MEP em comparação com as soluções baseadas em nuvem, resultando em menor latência.

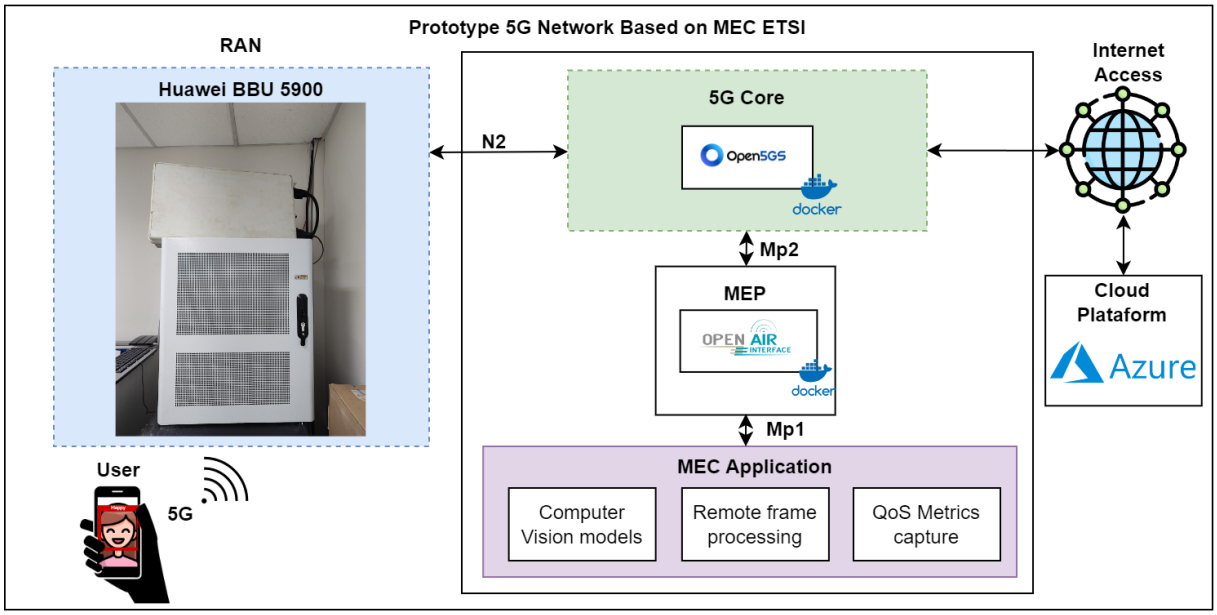

Figura 13: Arquitetura 5G experimental com núcleo da rede e plataforma MEC open-source e rede de acesso proprietária

A Figura 14 mostra a interface gráfica da MEP que centraliza os serviços MEC. Esta *Application Programming Interface* (API) é chamada Swagger (SmartBear Software, 2024), essa plataforma é um conjunto de ferramentas de software *open-source* desenvolvido pela SmartBear e oferece uma maneira padrão para descrever a estrutura de uma API, facilitando a compreensão, teste e interação com a API. Nela é incluído todos os *endpoints* da API MEC, permitindo a execução direta dos endpoints a partir do navegador. Ainda na figura, os desenvolvedores de MEC App podem visualizar os comandos para registrar e cancelar o registro de serviços MEC, bem como o comando para descobrir todos os serviços MEC disponíveis. Requisições HTTP também podem ser executadas através do navegador na interface Swagger. Na Figura 15, é possível ver a resposta a uma requisição de descoberta de serviço, exibindo os *endpoints* de serviços disponíveis para serem acessados na plataforma, uma descrição de cada *endpoint* e como acessar cada serviço.

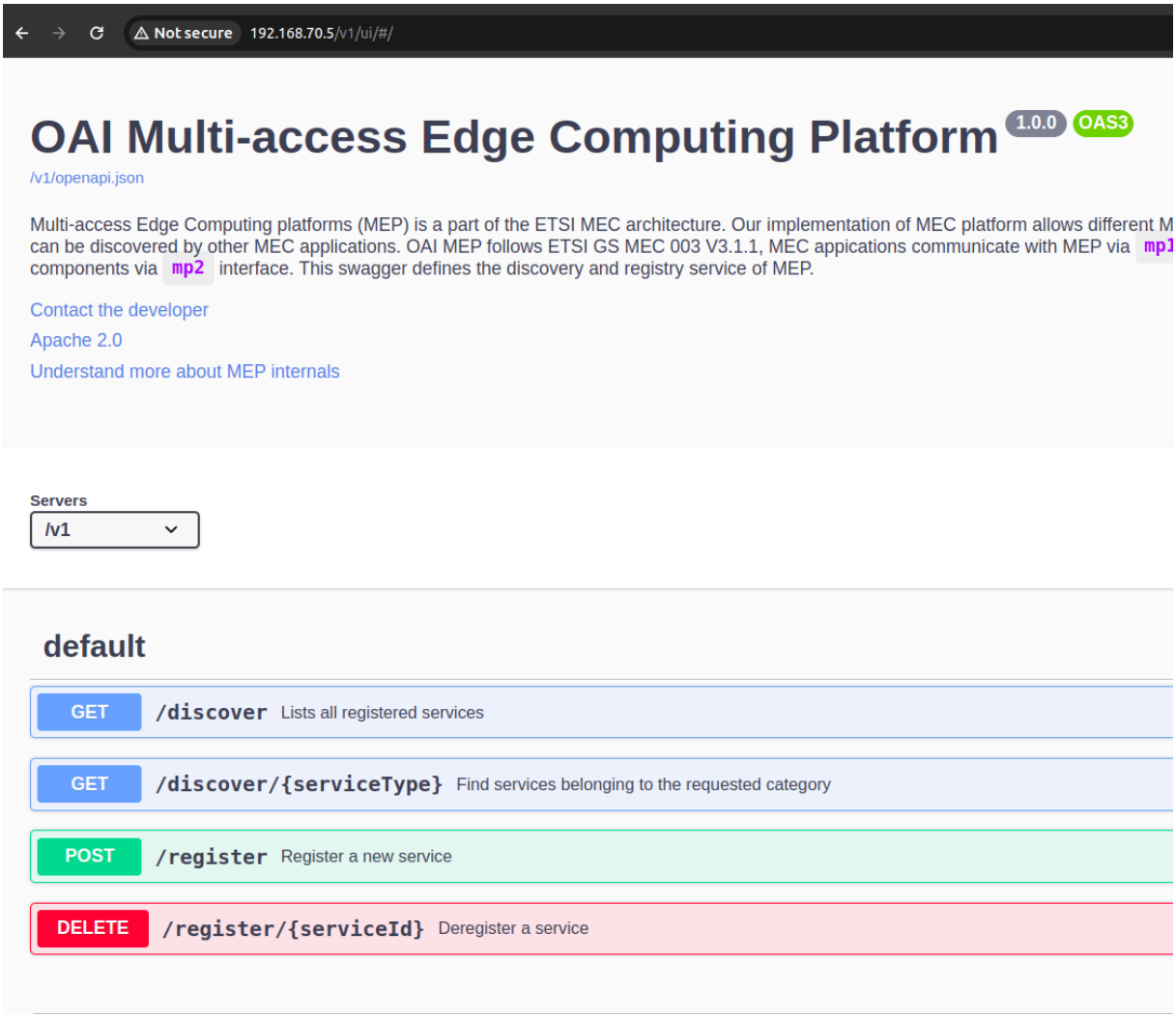

Figura 14: Swagger da plataforma MEC implementada pelo OAI

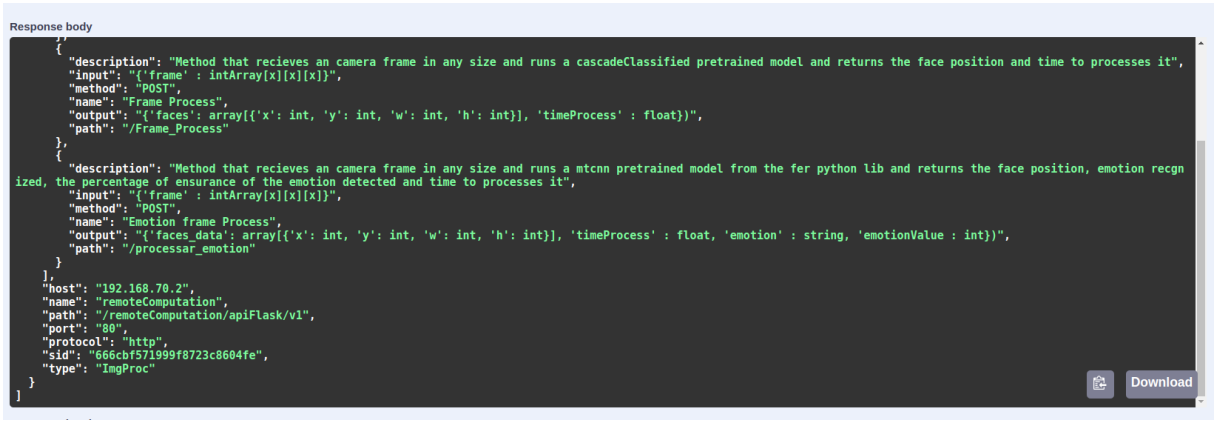

Figura 15: Resposta de uma requisição de descoberta de serviços na plataforma MEC através da interface gráfica do Swagger.

Finalmente, a aplicação em nuvem foi hospedada na plataforma Microsoft Azure. A aplicação em nuvem tem as mesmas funcionalidades e *endpoints* da aplicação MEC, exceto pelo local de hospedagem.

Tabela 2: Configuração do ambiente do projeto.

	Componente	**Especificação**
Servidor MEC/5GC	CPU RAM *Docker* *Docker Compose* Sistema Operacional	Intel Xeon Silver 4314 64GB 26.1.0 2.27.0 Ubuntu 20.04.6 LTS
Core	Platforma *Release* 3GPP Modo de Operação	Open5GS *Release* 17 5G SA
RAN	Platforma Radio Banda	Proprietária Huawei BBU 5900 n78
MEC	Platforma *Release*	OpenAirInterface ETSI GS MEC 003 V3.1.1
Servidor Nuvem	Platforma Num. vCPU RAM Sistema Operacional Localização	Microsoft Azure 2 vCPU 8 GB Ubuntu 20.04.6 LTS Brazil South (Zone 3)
Usuários	UE Sim Card	Motorola Edge 30 Ultra/Neo Sysmocom - S1J1.

4.2 APLICAÇÃO DE ANÁLISE DE SENTIMENTO

Para demonstrar a eficiência do ambiente 5G-MEC proposto para *offloading* de aplicações de visão computacional, foi implementada uma aplicação de análise de sentimento como um caso de uso. Esta aplicação visa detectar e classificar emoções humanas a partir de imagens de vídeo capturadas em tempo real da câmera de um dispositivo móvel, com o processamento do algorítimo de detecção sendo feito de forma remota tanto na borda da rede pela aplicação MEC quanto, em outro cenário, em um servidor remoto pela aplicação em nuvem.

A aplicação Android foi desenvolvida em JavaScript (Brendan Eich, 1995) utilizando o framework de aplicações Android React Native (Facebook, Inc., 2024) para realizar a captura de cada quadro da câmera do dispositivo e os enviar via solicitações HTTP para servidores remotos. Essa captura dos quadros atuais da câmera foi possível com a utilização da biblioteca de câmera de react-native-vision-camera (Vision Camera Team, 2024). Esses servidores processam as imagens usando algoritmos de reconhecimento de emoções e retornam os valores coletados e calculados referentes às métricas de *Quality of Service* (QoS) da iteração atual entre o servidor remoto e a aplicação, a localização dos rostos encontrados no quadro e também a emoção detectada em cada rosto. Com base nesse retorno, a aplicação Android exibe todos os rostos detectados e a emoção de cada rosto detectado na tela. A Figura 16 mostra o visual da aplicação durante a detecção de um rosto. É possível ver a localização do rosto na tela, o sentimento

detectado e também as métricas que o servidor capturou durante o quadro atual. No algoritmo 1 é possível ver a lógica em alto nível da aplicação.

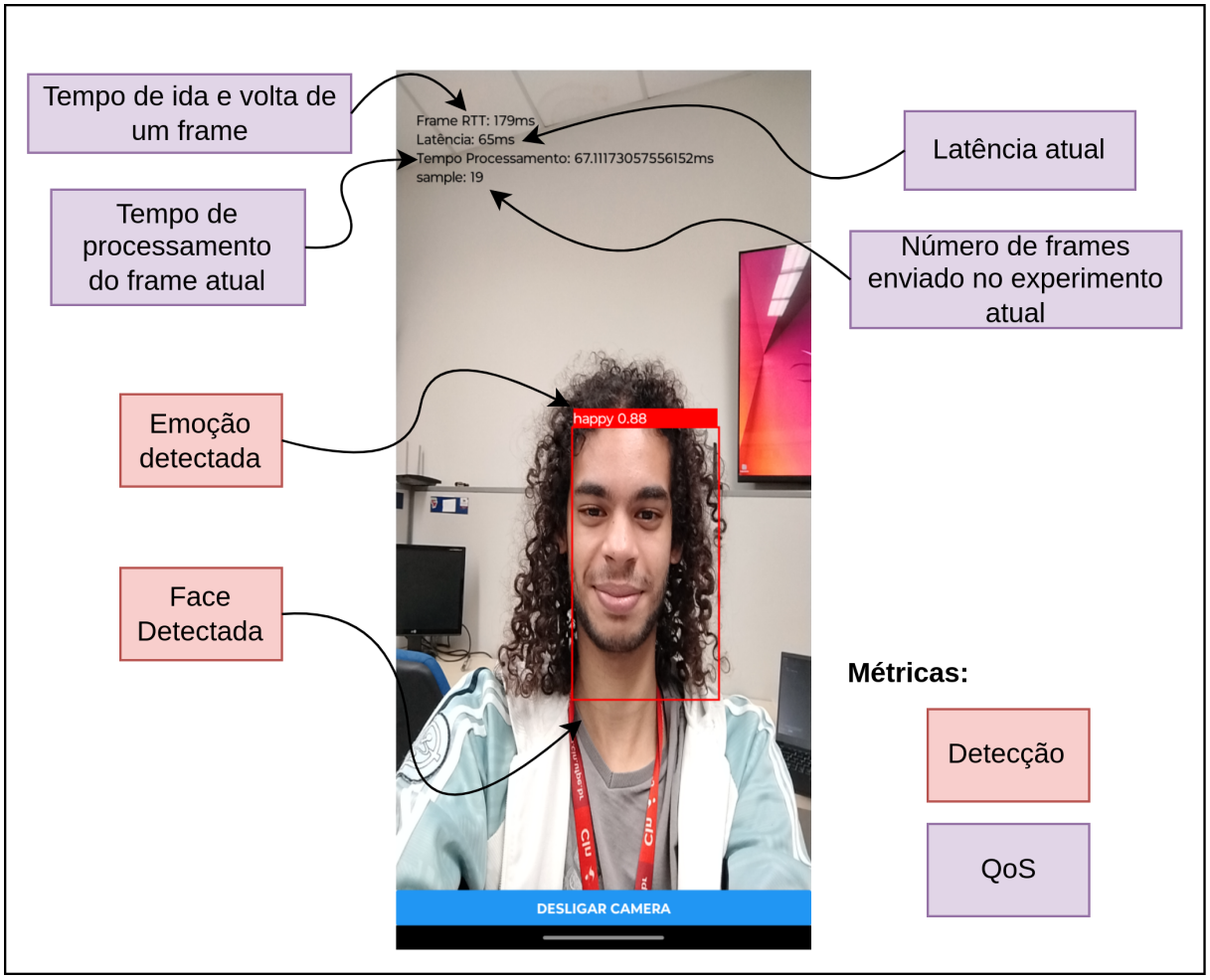

Figura 16: Tela da aplicação de visão computacional de dispositivo móvel demostrando o funcionamento da aplicação em um frame.

Para essas aplicações de processamento remoto, foi utilizado um servidor Pyhton implementado no framework Flask, empregando a princípio um modelo de detecção de faces utilizando o algoritmo Haar *Cascade Classifier* para validar a proposta. Em seguida, para aumentar a carga de processamento o algorítimo anterior foi descartado e foi utilizado a biblioteca *Facial Expression Recognition* (FER) (Shenk, 2021) para a implementação desse servidor remoto. Esta biblioteca utiliza a técnica MTCNN para a detecção de rostos e possui um modelo pré-treinado baseado em CNNs para classificar emoções em seis categorias: ‘medo’, ‘neutro’, ‘feliz’, ‘triste’, ‘raiva’ e ‘desgosto’. A escolha desse algoritmo é motivada por sua relevância nas pesquisas atuais e aplicações práticas, como medir o interesse em ambientes educacionais e terapias para crianças com Transtorno do Espectro Autista (TEA) Liu *et al.* (2019).

Esses servidores são hospedados em duas plataformas diferentes, sendo a primeira na borda da rede na MEC e em um servidor remoto hospedado na Microsoft Azure (Microsoft Corporation, 2024). As aplicações, por fim, capturam várias métricas da rede enquanto estão

Algoritmo 1: Algoritmo da aplicação android

```
Result: Captura de emoções em tempo real usando a câmera
Inicialize o programa;
while o programa está rodando do
    Comece a capturar a câmera;
    while há novos frames da câmera do
        Envie o frame atual para o servidor remoto;
        Espere pela resposta do servidor;
        if resposta recebida then
            Desenhe um quadrado na tela com a localização (x, y, height, weight);
            Escreva na posição (x, y) a emoção identificada;
            Escreva na tela as métricas de desempenho (Rtt, Tempo Processamento,
             Tempo Resposta );
Ends;
```

fazendo o *offloading* da carga, essas métricas servem para avaliar o desempenho da aplicação de visão computacional nas plataformas utilizadas.

5

RESULTADOS

Neste capítulo serão apresentados os resultados obtidos neste trabalho. Na Seção 5.1 descrevem-se os cenários utilizados, enfatizando os parâmetros empregados nas configurações, o número de execuções e as arquiteturas empregadas. Posteriormente, a Seção 5.2 oferece uma análise de desempenho da arquitetura proposta, comparando-a com a solução baseada em processamento em nuvem computacional remota.

5.1 CENÁRIOS AVALIADOS

Este capítulo apresenta a avaliação de desempenho da proposta, foi seguida a metodologia de avaliação idealizada por Raj jain (Jain, 1991). Com isso, primeiramente, o objetivo do sistema foi delimitado como o estudo do impacto da localização física de um servidor de *offloading* para aplicações de visão computacional em tempo real. Os serviços e as respostas que o sistema deve ser capaz de responder são os de receber imagens e responder com a localização de rostos e emoções nessa imagem.

Os cenários foram avaliados com base nas seguintes métricas:

- **RTT**: O RTT foi medido como o tempo de ida e volta de uma requisição feita pelo telefone ao servidor ao enviar um pacote de 40 bytes. Esta métrica ajuda a isolar a latência da rede da latência de processamento. A métrica foi calculada em milissegundos.

- **Tempo de Processamento do Algoritmo de Reconhecimento de Emoções**: Este tempo refere-se ao período necessário para o servidor remoto processar a imagem recebida e determinar a emoção presente no quadro e a localização dos rostos na tela. A métrica foi calculada em milissegundos.

- **Tempo de Resposta**: Este é o tempo desde o envio da imagem do telefone até o recebimento da resposta do servidor no dispositivo do usuário. Esta métrica é crucial para entender a latência total envolvida no processo de reconhecimento de emoções. A métrica foi calculada em milissegundos.

- **Vazão (*Throughput*)**: A vazão foi calculada dividindo-se o tamanho dos dados enviados e recebidos pelo tempo de resposta. Esta métrica é importante para avaliar a eficiência da rede em termos do volume de dados transmitidos por unidade de tempo. A métrica foi calculada em Megabits por segundo.

Os parâmetros avaliados podem ser divididos em dois tipos: os Parâmetros de Sistema e os Parâmetros de Carga de Trabalho. Os parâmetros de sistema são os parâmetros descritos na Tabela 2 e os de Carga de Trabalho são os dispositivos utilizados no experimento, sendo eles o Motorola Edge 30 Ultra e o Motorola Edge 30 Neo.

Por fim, os fatores de estudos quem podem ser vistos na tabela 3 são basicamente os parâmetros variados durante as realizações dos testes, sendo eles: Número de Dispositivos Simultâneos e Plataforma Remota de *Offloading*.

Fatores de Estudo	**Valores**
Número de dispositivos simultâneos	1,2
Plataforma Remota de *Offloading*	MEC, *Cloud*

Tabela 3: Fatores de estudos da avaliação de desempenho e seus respectivos valores.

5.2 AVALIAÇÃO DE DESEMPENHO DA REDE

Com base nos resultados obtidos a partir de testes realizados com os dispositivos Motorola Edge 30 Ultra e Motorola Edge 30 Neo, foram revelados alguns *insights* sobre a comparação de desempenho entre processamento baseado em nuvem e MEC. A seguir, discutimos cada métrica avaliada e os impactos observados em cada cenário.

Para cada cenário, os resultados foram apresentados seguindo um padrão específico. "Cloud1" representa o resultado obtido no cenário de nuvem remota e utilizando um dispositivo. "Cloud2" representa o resultado obtido no cenário de nuvem remota e utilizando dois dispositivos. O mesmo padrão se aplica a "MEC1" e "MEC2", correspondendo à avaliação na plataforma MEC usando um e dois dispositivos, respectivamente. Finalmente, para alcançar um intervalo de confiança de 95%, 100 amostras foram capturadas para cada cenário.

5.2.1 RTT

Como mostrado na Figura 17, o RTT médio foi substancialmente menor nos cenários onde o processamento foi realizado no MEC (MEC1 e MEC2) em comparação com o processamento baseado em nuvem (Cloud1 e Cloud2). O RTT mais baixo observado no MEC (82,0 ms e 76,9 ms) reflete a vantagem de processar dados mais próximos do usuário final, reduzindo assim a latência da rede. Em termos percentuais, o MEC1 apresentou uma melhoria de aproximadamente 43,91% em relação ao Cloud1, enquanto o MEC2 apresentou uma melhoria de aproximadamente 55,52% em relação ao Cloud2. Este resultado confirma a expectativa de que o MEC pode

fornecer uma resposta mais rápida devido à sua proximidade com os dispositivos móveis, o que é crucial para aplicações de visão computacional que requerem processamento em tempo real.

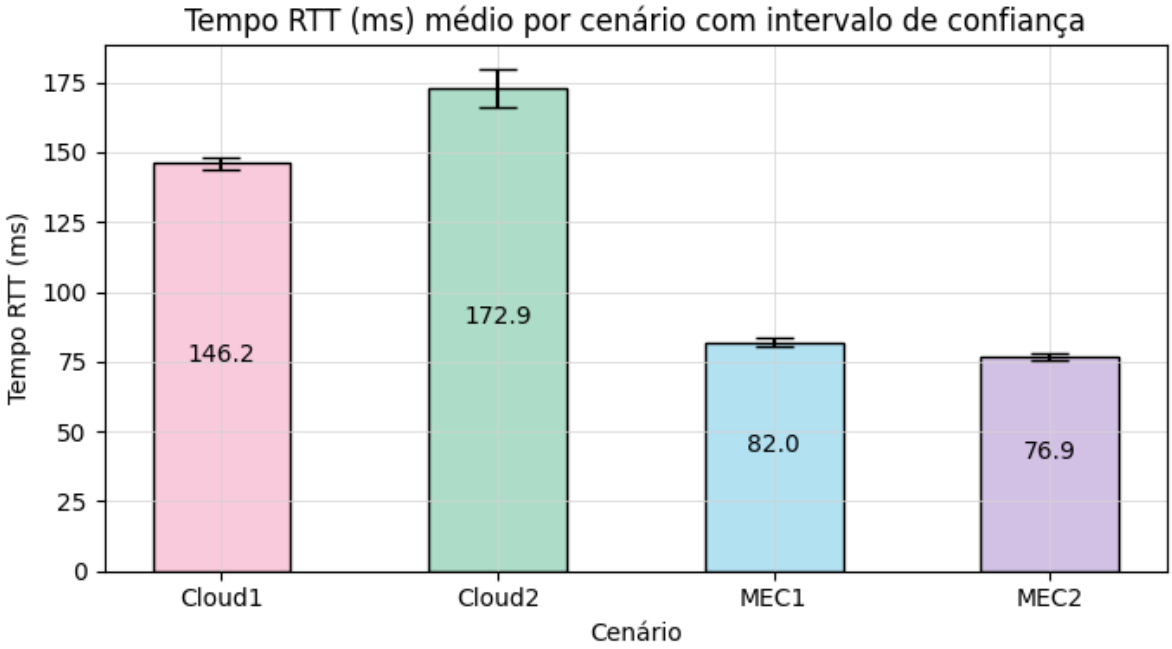

Figura 17: RTT médio para os cenários MEC e Nuvem medidos para um e dois dispositivos.

5.2.2 Tempo de Processamento

Os tempos de processamento do algoritmo de reconhecimento de emoções, apresentados na Figura 18, demonstram que o MEC é significativamente mais eficiente (54,2 ms e 78,0 ms) em comparação com a nuvem (164,7 ms e 287,8 ms). Isso se traduz em melhorias notáveis de 67,5% e 73,1% para o MEC em comparação com a computação em nuvem em cada cenário, respectivamente. Nota-se que a diferença entre tempos de processamento na nuvem é significantemente elevada quando o número de dispositivos aumenta. Isso se dá devido às limitações de hardware do servidor de alocação na nuvem. Porém, com ajuda da sessão 5.2.1 é possível inferir que o tempo que o dado leva na rede para chegar o servidor remoto não se altera tanto com o número de dispositivos, mas o congestionamento de rede se mostra um limitante para os testes propostos.

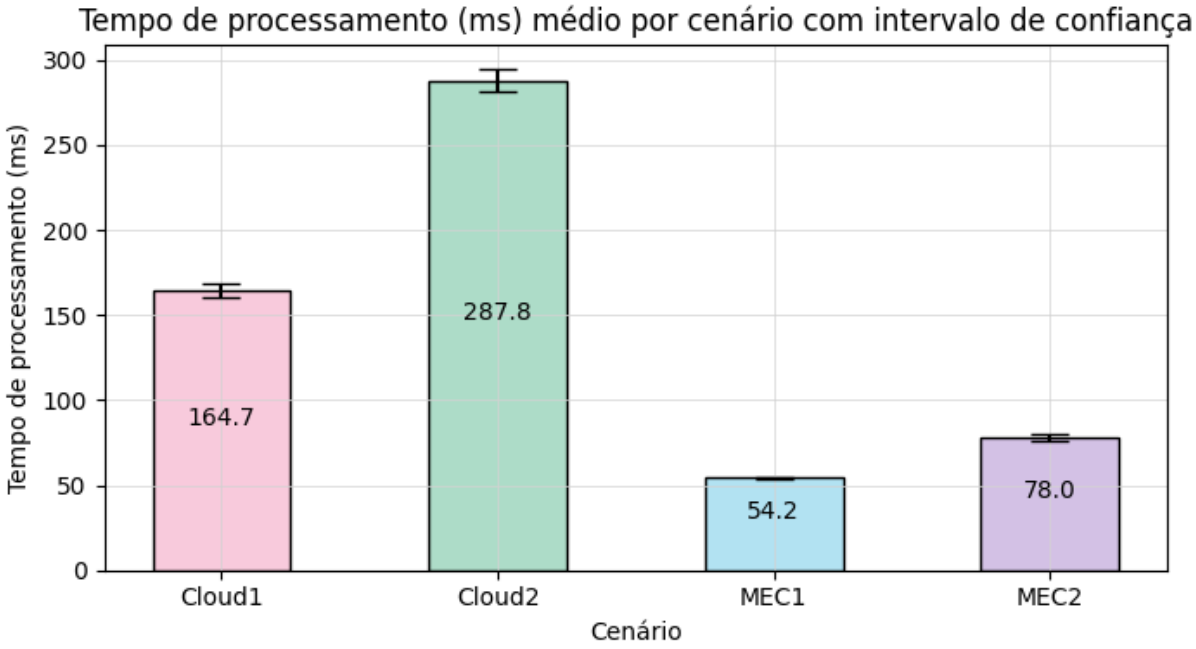

Figura 18: Tempo de processamento médio para os cenários MEC e Nuvem medidos para um e dois dispositivos.

5.2.3 Tempo de Resposta

O tempo total de resposta, que inclui tanto o RTT quanto o tempo de processamento, também favorece o MEC, como mostrado na Figura 19. Com tempos de resposta de 206,1 ms e 258,0 ms para o MEC, em comparação com 717,2 ms e 1483,9 ms para a nuvem, fica claro que o MEC oferece desempenho superior. Especificamente, o MEC mostra uma melhoria de desempenho de aproximadamente 71,3% e 82,6% sobre a nuvem. Esta métrica é particularmente importante para aplicações de visão computacional, onde atrasos perceptíveis podem comprometer a funcionalidade e a utilidade da aplicação. A menor latência total observada no MEC destaca sua capacidade de fornecer respostas rápidas e eficientes, cruciais para aplicações sensíveis ao tempo e críticas.

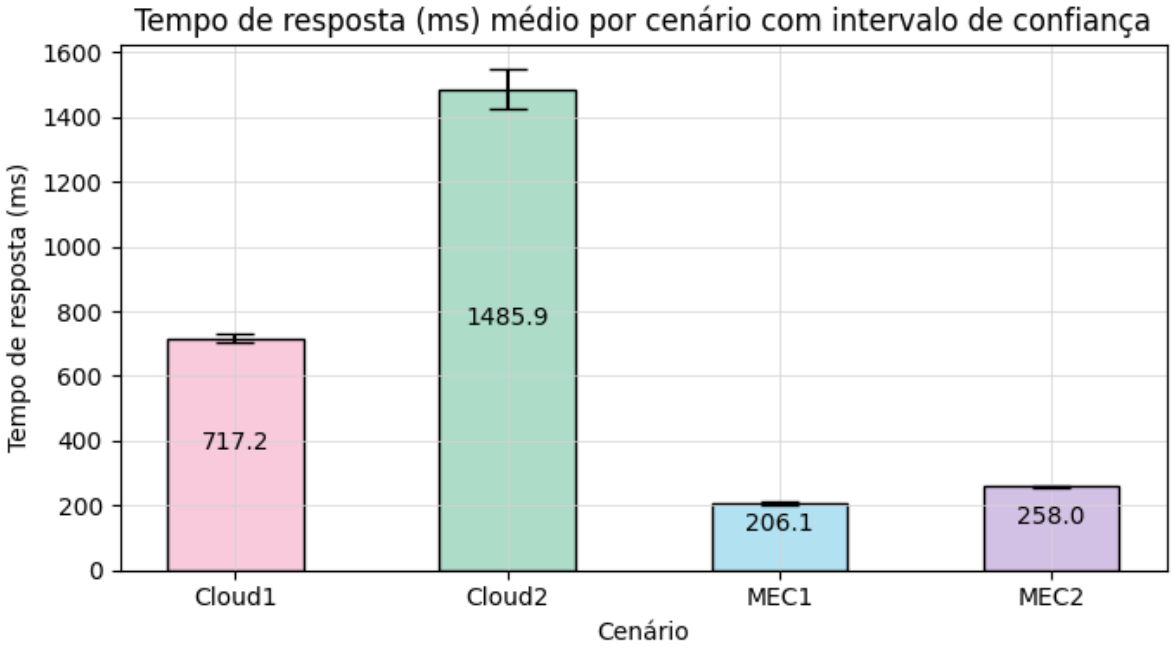

Figura 19: Tempo de resposta médio para os cenários MEC e Nuvem medidos para um e dois dispositivos.

Apesar do tempo de processamento ter aumentado em cerca de 120 ms devido ao hardware limitado da arquitetura utilizada, observando-se o tempo total de resposta mostra que ela não foi o fator mais relevante durante todo o processo de *offloading*.

Os impactos do tempo de resposta alto podem ser observados na Figura 20. Um tempo de resposta elevado resulta em atrasos na detecção de rosto e emoção, o que pode levar a erros significativos. Por exemplo, a detecção do rosto pode ocorrer em um local incorreto, identificando erroneamente outra pessoa próxima ao alvo e interpretando incorretamente a emoção na imagem, como detectar felicidade onde ela não existe.

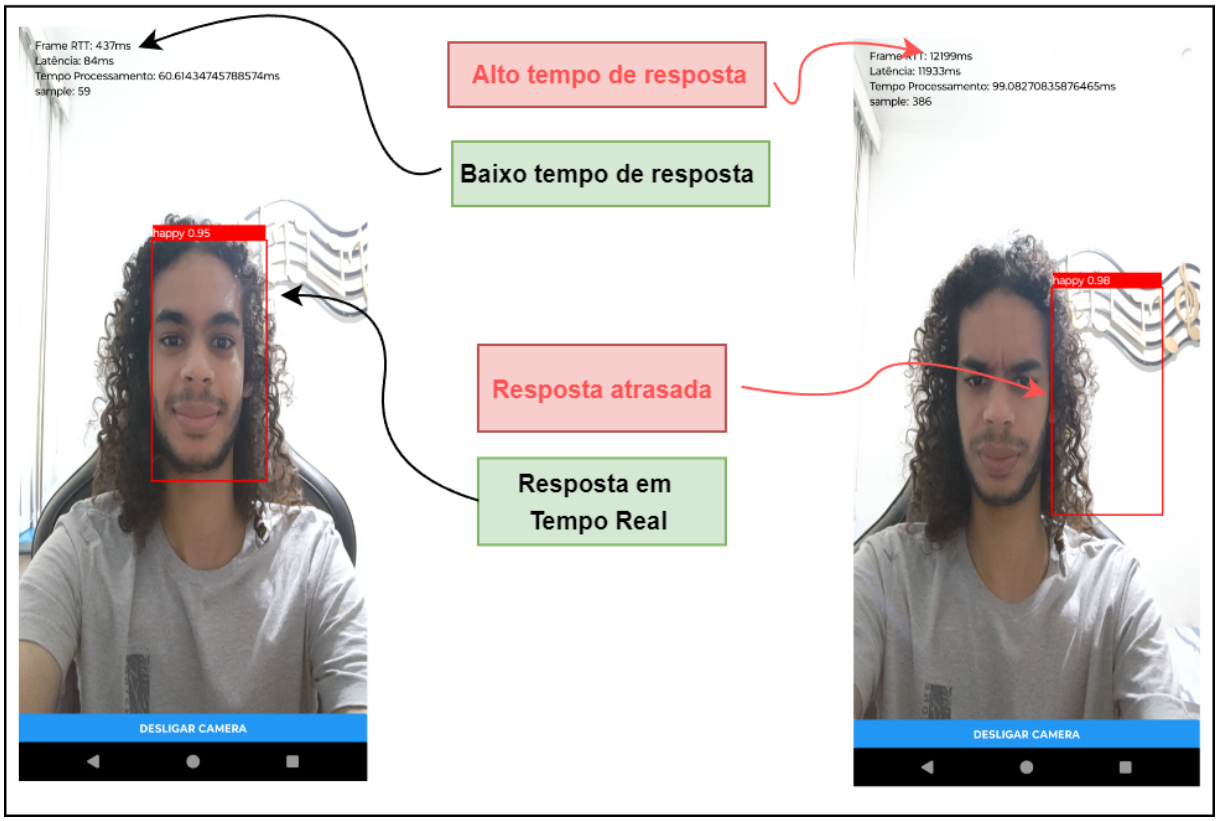

Figura 20: Impactos de alto tempo de resposta em uma aplicação de Visão computacional com processamento remoto em tempo real.

Esses erros podem ter repercussões críticas durante o experimento dado. Por exemplo, em uma aplicação que avalia o engajamento dos alunos durante uma aula, os atrasos podem fazer com que o algoritmo confunda um aluno com outro devido a mudanças de posição ou interprete erroneamente se os alunos têm dificuldade de compreensão em partes da aula onde, na verdade, não têm, e vice-versa.

5.2.4 Vazão

Finalmente, a Figura 21 mostra que a taxa de transferência média foi significativamente maior nos cenários MEC (3,6 Mbps e 2,9 Mbps) em comparação com a nuvem (1,0 Mbps e 0,6 Mbps). O MEC mostra uma melhoria de desempenho de, aproximadamente, 260% e 383,33% sobre a nuvem. Este aumento na taxa de transferência indica que o MEC pode transmitir e processar volumes maiores de dados mais rapidamente, fazendo melhor uso da largura de banda disponível. Este resultado é crucial para aplicações de visão computacional que dependem de streaming contínuo e de alta qualidade, pois uma taxa de transferência maior permite que mais dados sejam transmitidos em menos tempo, melhorando a eficiência geral do sistema.

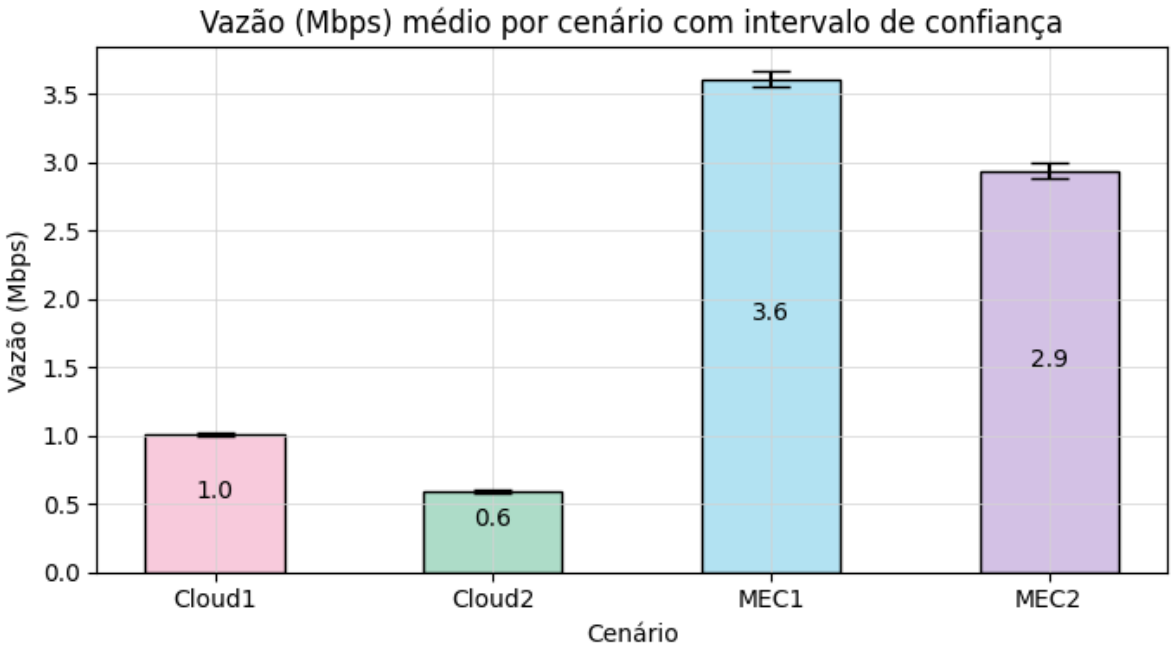

Figura 21: Vazão média para os cenários MEC e Nuvem medidos para um e dois dispositivos.

6

CONCLUSÃO

As aplicações de visão computacional ganharam significativa atenção nos últimos anos devido aos avanços nas tecnologias de processamento de imagens e à crescente demanda por soluções que utilizam inteligência artificial para interpretar e responder a dados visuais. Seja na monitorização de vigilância ou na análise comportamental, a visão computacional está impulsionando avanços importantes em vários setores. No entanto, essas aplicações necessitam frequentemente de alto poder de processamento e respostas rápidas, que podem ser comprometidas pela latência associada ao processamento remoto tradicional e pelo poder computacional limitado dos dispositivos móveis modernos.

Este trabalho de conclusão de curso elaborou e implementou uma arquitetura 5G-MEC para *offloading* de aplicações em tempo real que requerem alta carga de trabalho. Além disso, desenvolveu uma aplicação para análise de sentimentos utilizando visão computacional.

Nesse contexto, a abordagem conjunta 5G-MEC pode reduzir a latência e melhorar a eficiência dessas aplicações. A Computação em Borda de Acesso Múltiplo permite o processamento e armazenamento mais próximo do usuário final, enquanto o 5G fornece uma infraestrutura de rede com alta velocidade, baixa latência e maior capacidade para conexões simultâneas de dispositivos. A presente proposta introduziu uma arquitetura MEC no contexto das redes 5G para consolidar aplicações de visão computacional eficientes para dispositivos moveis.

Os resultados obtidos demonstram que o uso de MEC juntamente com redes 5G proporciona ganhos substanciais de latência e taxa de transferência para aplicações de visão computacional. O MEC reduz a latência, tornando-o uma solução superior à computação em nuvem para aplicações que exigem processamento em tempo real com baixo tempo de resposta. Embora as métricas de tempo de processamento também sejam influenciadas pelo poder computacional dos servidores de borda e nuvem, os resultados revelam que a diferença de desempenho mais significativa ocorre devido ao tempo de transferência de dados entre o dispositivo e o servidor. Devido à sua proximidade com os dispositivos e aos requisitos reduzidos de transferência de dados, o MEC oferece um tempo de transmissão significativamente menor em comparação com a computação em nuvem.

Para trabalhos futuros, é essencial expandir o número de usuários e analisar o consumo de recursos computacionais e energia. Além disso, os esforços devem se concentrar em aprimorar

a aplicação para melhorar a taxa de transmissão.

REFERÊNCIAS

Amazon Web Services (2023). O que é visão computacional? https://aws.amazon.com/pt/what-is/computer-vision/. Acesso em: 4 jul. 2024.

Araújo, M., Silva, J., Santos, P. M., Singh, H., Gunjal, D., Fonseca, J., Duarte, P., Mendes, B., Barbosa, R., Steenkiste, P., Sabamoniri, S., Lam, L., Pereira, J., & Kurunathan, H. (2023). Demo: Object detection under 5g-edge mobility. In *2023 IEEE 24th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 343–345.

Armin Ronacher (2010). *Flask*. A lightweight WSGI web application framework in Python.

Babcock, N. (2021). Computer vision pipeline architecture: A tutorial. https://www.toptal.com/computer-vision/computer-vision-pipeline. Acesso em: 4 jul. 2024.

Baena, C., Fortes, S., Penaherrera-Pulla, O. S., Baena, E., & Barco, R. (2024). Gaming in the cloud: 5g as the pillar for future gaming approaches. *IEEE Communications Magazine*, 1–7.

Bartolín-Arnau, L., Vera-Pérez, J., Sempere-Payá, V., & Silvestre-Blanes, J. (2024). Are european initiatives related to local spectrum allocation for private 5g networks ready for use in industrial cases? *IEEE Access*, PP:1–1.

Borcoci, E., Vochin, M., & Obreja, S. (2018). Mobile edge computing versus fog computing in internet of vehicles.

Brendan Eich (1995). *JavaScript*. A high-level, interpreted programming language commonly used in web development.

Capgemini (2023). Solutions for the 5g edge revolution. Accessed: 2024-08-13.

ETSI MEC ISG (2024). Multi-access Edge Computing (MEC). "https://www.etsi.org/technologies/multi-access-edge-computing". [Accessed: 13-Jul-2024].

European Telecommunications Standards Institute (ETSI) (2024). ETSI Official Website. "https://www.etsi.org/". [Accessed: 13-Jul-2024].

Facebook, Inc. (2024). *React Native*. A framework for building native apps using React.

Goel, A., Tung, C., Lu, Y.-H., & Thiruvathukal, G. K. (2020). A survey of methods for low-power deep learning and computer vision. In *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, 1–6.

Guevara, C. & Auat Cheein, F. (2020). The role of 5g technologies: Challenges in smart cities and intelligent transportation systems. *Sustainability*, 12.

Gupta, A. & Jha, R. (2015). A survey of 5g network: Architecture and emerging technologies. *Access, IEEE*, 3:1206–1232.

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley.

Janai, J., Güney, F., Behl, A., & Geiger, A. (2020). Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1–3):1–308.

Kadir, K., Kamaruddin, M., Nasir, H., Safie, S., & Bakti, Z. (2014). A comparative study between lbp and haar-like features for face detection using opencv. 335–339.

Kim, M., Lee, D. G., & Kim, K.-Y. (2015). System architecture for real-time face detection on analog video camera. *International Journal of Distributed Sensor Networks*, 2015:1–11.

Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.

Lin, L., Liao, X., Jin, H., & Li, P. (2019). Computation offloading toward edge computing. *Proceedings of the IEEE*, 107(8):1584–1607.

Ling, Z., Yang, S., Gou, F., Dai, Z., & Wu, J. (2022). Intelligent assistant diagnosis system of osteosarcoma mri image based on transformer and convolution in developing countries. *IEEE Journal of Biomedical and Health Informatics*, 26(11):5563–5574.

Liu, J., He, K., Wang, Z., & Liu, H. (2019). A computer vision system to assist the early screening of autism spectrum disorder. In Sun, F., Liu, H., & Hu, D., editors, *Cognitive Systems and Signal Processing*, 27–38.

Mao, Y., You, C., Zhang, J., Huang, K., & Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys Tutorials*, 19(4):2322–2358.

Marsh, B. (2016). Multivariate analysis of the vector boson fusion higgs boson.

Microsoft Corporation (2024). *Microsoft Azure*. A cloud computing service created by Microsoft providing a range of cloud services, including those for computing, analytics, storage, and networking.

Mihai, R., Craciunescu, R., Martian, A., Li, F. Y., Patachia, C., & Vochin, M.-C. (2022). Open-source enabled beyond 5g private mobile networks: From concept to prototype. In *2022 25th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 181–186.

MordorIntelligence (2024). Tamanho do mercado de visão computacional e análise de ações – tendências e previsões de crescimento (2024 – 2029). `https://www.mordorintelligence.com/pt/industry-reports/computer-vision-market`. Accessed: 2024-06-23.

Moreira, J. B., Mamede, H., Pereira, V., & Sousa, B. (2020). Next generation of microservices for the 5g service-based architecture. *International Journal of Network Management*, 30(6):e2132. e2132 nem.2132.

Morín, D. G., Pérez, P., & Armada, A. G. (2022). Toward the distributed implementation of immersive augmented reality architectures on 5g networks. *IEEE Communications Magazine*, 60(2):46–52.

Motlagh, N. H., Bagaa, M., & Taleb, T. (2017). Uav-based iot platform: A crowd surveillance use case. *IEEE Communications Magazine*, 55(2):128–134.

Nencioni, G., Garroppo, R. G., & Olimid, R. F. (2023). 5g multi-access edge computing: A survey on security, dependability, and performance. *IEEE Access*, 11:63496–63533.

Open5GS (2024). Open5gs: Open source 5g core network. https://open5gs.org/. Accessed: 2024-06-23.

OpenAirInterface (2024). Openairinterface: 5g software alliance for democratising wireless innovation. https://openairinterface.org/. Accessed: 2024-06-24.

OpenCV (2017). Face detection using haar cascades. https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html.

Paneru, S. & Jeelani, I. (2021). Computer vision applications in construction: Current state, opportunities challenges. *Automation in Construction*, 132:103940.

Python Software Foundation (2024). Python Programming Language – Official Website. https://www.python.org/. [Accessed: 13-Jul-2024].

Rasheed, A., Zafar, B., Rasheed, A., Ali, N., Sajid, M., Dar, S. H., Habib, U., Shehryar, T., & Mahmood, M. T. (2020). Fabric defect detection using computer vision techniques: A comprehensive review. *Mathematical Problems in Engineering*, 2020(1):8189403.

Ren, J., He, Y., Huang, G., Yu, G., Cai, Y., & Zhang, Z. (2019). An edge-computing based architecture for mobile augmented reality. *IEEE Network*, 33(4):162–169.

Shenk, J. (2021). Fer. A Python library for facial emotion recognition using deep learning techniques.

SmartBear Software (2024). *Swagger*. A set of open-source tools for designing, building, documenting, and consuming RESTful APIs.

Verre, C. (2021). Visão computacional: o que é, como funciona e aplicações. https://blog.singularityubrazil.com/blog/visao-computacional/. Acesso em: 4 jul. 2024.

Viola, P. & Jones, M. (2001). 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*, 1:I–I.

Vision Camera Team (2024). *React Native Vision Camera*. A guide and documentation for the React Native Vision Camera library.

Yang, X., Zhang, Q., Yang, X., Peng, Q., Li, Z., & Wang, N. (2018). Edge detection in cassini astronomy image using extreme learning machine. *MATEC Web of Conferences*, 189:06007.

Zhang, C. & Zhang, Z. (2010). *A Survey of Recent Advances in Face Detection*.

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503.