UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Deepfake: explorando técnicas de detecção de manipulação digital de imagens de faces

Jorge de Jesus Gomes Leandro

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Jorge de Jesus Gomes Leandro

Deepfake: explorando técnicas de detecção de manipulação digital de imagens de faces

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Odemir Martinez Bruno

Versão original

São Carlos

2022

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Leandro, Jorge de Jesus Gomes
L437d DeepFake: explorando técnicas de detecção de manipulação digital de imagens de faces / Jorge de Jesus Gomes Leandro; orientador Odemir Martinez Bruno. -- São Carlos, 2022.
78 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2022.

1. Inteligência Artificial. 2. Visão Computacional. 3. Redes Neurais Convolucionais. 4. Deepfake. 5. Dispositivos Móveis. I. Martinez Bruno, Odemir , orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Jorge de Jesus Gomes Leandro

Deepfake: exploring techniques for the detection of digital manipulation in faces images

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Odemir Martinez Bruno

Original version

São Carlos

2022

Este trabalho é dedicado a todos os professores que cruzaram meu caminho ontem, contribuindo com seu conhecimento e inspiração para a construção do que me tornei hoje.

AGRADECIMENTOS

Pelo sopro de vida e a avidez por desafios que lhe imprimam propósito, sou grato ao Criador.

Por apontar-me o caminho de valores superiores que enobrecem a alma, sou grato aos meus pais, Dulce e Satiro.

Pelo suporte incondicional e a resignação com que sempre compreenderam os períodos de recolhimento que minha jornada nos impõe, sou grato a minha esposa Sheila e filha Julia.

Pelo apoio e motivação no ambiente corporativo da Motorola Mobility LLC, sou grato a Alexandre Olivieri, ex-diretor do Departamento de Desenvolvimento de Software. Pelo incentivo e suporte no ambiente corporativo da Microsoft Research, sou grato a William B. Dolan, PhD, *Partner Research Manager* na área de Inteligência Artificial.

"Once you stop learning you start dying."
Albert Einstein

RESUMO

Leandro, J. J. G. ***Deepfake*: explorando técnicas de detecção de manipulação digital de imagens de faces**. 2022. 78p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2022.

O presente documento aborda o tema de detecção de manipulação digital de imagens de faces, com ênfase na técnica *Deepfake*, um tópico relacionado à área de Inteligência Artificial. Avanços recentes em tecnologias de geração de *Deepfake*, catalizadas pelo poder de difusão de notícias falsas das redes sociais e a profusão de dispositivos móveis, resultam numa combinação com consequências preocupantes para todas as escalas da sociedade moderna. A maioria dos modelos estado da arte apresenta capacidade da ordem de milhões de parâmetros. Neste trabalho, técnicas para detecção de *Deepfake*, candidatas à implantação em dispositivos móveis, foram investigadas. Para tanto, foi proposta uma extensão da arquitetura `Meso-4` por um bloco incorporando o operador Filtro de `Sobel`, com parâmetros não-treináveis. Três configurações foram exploradas em um Estudo de Ablação sobre a base de dados *Deepfake*, conforme o paradigma de Aprendizado Supervisionado. O desempenho foi estimado por métricas usuais em problemas de classificação. Os resultados obtidos foram validados estatisticamente pelo *Teste de McNemar*. A latência e consumo de memória do modelo foram avaliados no dispositivo móvel. A arquitetura proposta `MesonetSobelConcat` produziu os melhores resultados, com acurácia binária em $0,961$ e AUC em $0,991$. O tempo médio de inferência no dispositivo medido resultou em torno de 108 milissegundos por *frame*, enquanto o consumo total de memória foi de quase 33 Mb. Os resultados obtidos indicam que a `MesonetSobelConcat` apresenta desempenho superior à `Meso-4` na detecção de *Deepfake* com taxas satisfatórias de acerto, mostrando-se também viável para implantação em dispositivos embarcados e móveis, com recursos limitados de *hardware*.

Palavras-chave: Inteligência Artificial. Visão Computacional. Redes Neurais Convolucionais. *Deepfake*. Dispositivos móveis.

ABSTRACT

Leandro, J. J. G. ***Deepfake*: exploring techniques for the detection of digital manipulation in faces images**. 2022. 78p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2022.

This document addresses the theme of detecting manipulation in digital images of faces, with an emphasis on the *Deepfake* technique, a topic related to the field of Artificial Intelligence. The recent progress in technologies for the generation of *Deepfake*, catalyzed by the power of fake news diffusion in social media, besides the profusion of mobile devices, result in a combination leading to major concerns for every scale in modern society. Most state-of-the-art models present a capacity in the order of millions of parameters. In the present work, candidate techniques for detecting *Deepfake* on mobile devices have been investigated. To that end, an extension to the `Meso-4` architecture has been proposed by incorporating the `Sobel` Filter block with non-learnable parameters. According to the Supervised Learning paradigm, three settings have been explored under an Ablation Study over the dataset *Deepfake*. Usual metrics for classification problems have been used to estimate the model performance. The *McNemar's Test* has statistically validated the obtained results. Latency and memory footprint have been evaluated on device. The proposed architecture `MesonetSobelConcat` yielded the best results with a binary accuracy of 0.961 and *AUC* of 0.991. The average time measured for inference on device was about 108 milliseconds, while the overall memory footprint measured was close to 33 Mb. The obtained results indicate that the `MesonetSobelConcat` overperforms `Meso-4` in detecting *Deepfake* with reasonable true positive rates and has shown to be feasible for deployment in embedded and mobile devices under limited *hardware* resources.

Keywords: Artificial Intelligence. Computer Vision. Convolutional Neural Networks. *Deepfake*. Mobile devices.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

Acc	*Accuracy* ou Acurácia
ADAM	Adaptive Moment Estimation
AGCN	Adaptive Gabor Convolutional Network
AUC	Area Under the Curve
API	Application Programming Interface
BN	Batch Normalization
CNN	Convolutional Neural Network
DFDC	Deepfake Detection Challenge
ED	Encoder-Decoder
FC	Fully Connected
FN	Falso Negativo
FP	Falso Positivo
GAN	Generative Adversarial Network
GCF	Gabor Convolutional Filter
GCN	Gabor Convolutional Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
HMN	Hierarchical Memory Network
ILM	Industrial Light and Magic
LSTM	Long Short Term Memory
MIT	Massachusetts Institute of Technology
MFC	Media Forensics Challenge
MSE	Mean Squared Error
NIST	National Institute of Standards and Technology

USP	Universidade de São Paulo
USPSC	Campus USP de São Carlos
Prec	*Precision* ou Precisão
Rec	*Recall* ou Revocação
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
ROC	Receiving Operating Characteristic
VAE	Variational Autoencoder
VGG	Architecture by the Visual Geometry Group
VFX	Visual Effects
ViT	Visual Transformer
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

LISTA DE SÍMBOLOS

x	Entrada
s	Identidade original
t	Identidade alvo
x_s	Imagem original
x_t	Imagem alvo
x_g	Imagem gerada
e	*Embedding*, *Encoding* ou Codificação
En	*Encoder* ou Rede Codificadora
De	*Decoder* ou Rede Decodificadora
E	Espaço Latente
D	Rede Discriminadora
D_a	Discriminador de amostra do domíno a
D_b	Discriminador de amostra do domínio b
G	Rede Geradora
z	Vetor amostrado de uma distribuição Normal
U_k	Camada oculta
H_{ab}	Gerador de amostra do domínio b
H_{ba}	Gerador de amostra do domínio a
N	Distribuição Normal
$\mathcal{L}$	*Loss* ou Função de Perda
max	O máximo de
log	O logaritmo de
∇	Operador Gradiente
G_x	Componente horizontal do gradiente de uma imagem
G_y	Componente vertical do gradiente de uma imagem

SUMÁRIO

1 INTRODUÇÃO

O recente advento de técnicas baseadas em Aprendizado Profundo (*Deep Learning*) para manipulação de imagens e geração de imagens falsas tem despertado o interesse do público em geral. Deve-se esse interesse tanto ao seu potencial para fins de entretenimento legítimo, quanto ao risco que representam, viabilizando a substituição de faces entre pessoas para aplicações questionáveis, como construção de farsas, notícias falsas, fraudes financeiras e pornografia falsa (TOLOSANA *et al.*, 2020).

Essa ameaça tem fomentado o interesse da comunidade em desenvolver técnicas para Detecção de *Deepfake* e manipulação facial em imagens estáticas e vídeo. Dentre suas principais motivações, figuram os efeitos devastadores consequentes ao compartilhamento de mídia falsa em redes sociais e plataformas de imagem e vídeo, livremente acessáveis por dispositivos móveis.

O presente projeto de pesquisa propõe a exploração, eventual simplificação e/ou aperfeiçoamento de técnicas e arquiteturas, tais como *Mesoscopic Features* (AFCHAR *et al.*, 2018) aliadas a Redes Neurais Convolucionais (*CNN*) (ROSSLER *et al.*, 2019), para detecção de imagens manipuladas e/ou sintéticas, considerando sua possível aplicação em dispositivos com restrições de recursos, como telefones celulares.

A popularização e o livre acesso à tecnologia de geração de imagens falsas, mediante redes neurais adversariais (*GAN*), permitem a qualquer pessoa a geração de imagens realistas de faces de indivíduos que não existem ou mesmo a substituição da identidade de indivíduos em imagens estáticas ou sequências de vídeo, uma técnica conhecida como *Deepfake* (TOLOSANA *et al.*, 2020).

Por um lado, essa tecnologia tem beneficiado setores como entretenimento, cinema, efeitos visuais (*VFX*) e captura de movimento com marcadores. Recentemente, foi amplamente divulgada a contratação do influenciador digital conhecido pelo pseudônimo de *Shamook* como artista sênior pela *ILM (Industrial Light and Magic)*, uma divisão de efeitos especiais da LucasFilm. Uma semana depois da veiculação de um episódio de *The Mandalorian*, objeto de duras críticas pela qualidade dos efeitos de Computação Gráfica na apresentação do rosto da personagem Luke Skywalker, *Shamook* lançou sua versão da mesma cena aperfeiçoada com tecnologia de *Deepfake*, tornando-se viral imediatamente (BBC-NEWS, 2021).

Por outro lado, tais técnicas de geração fotorrealistas também constituem ameaça para a sociedade de modo geral, haja vista seu uso potencial para manipulação da identidade de indivíduos, visando à geração de evidências forjadas, campanhas políticas e publicitárias maliciosas, entre outros. O emprego de *Deepfake* combinado a campanhas de desinformação

pode até mesmo surtir efeitos indesejáveis às eleições (QI *et al.*, 2020). Os prejuízos podem variar em escala, desde indivíduos, organizações e grupos até a sociedade em geral, assim como variar em velocidade, partindo de efeitos negativos imediatos a reputações, até a lenta corrosão da confiança pela circulação de pseudoevidências em imagens e vídeo. Esse perigo à integridade da informação pode acarretar consequências sobre a privacidade, aspectos legais, política, segurança e a erosão potencial da confiança.(CHU *et al.*, 2020)

Como agravante, considere a ubiquidade dos dispositivos móveis, como *smartphones*, permeando todos os aspectos do nosso quotidiano (WANG *et al.*, 2018) e facilitando a disseminação desse tipo de conteúdo. Inferência de modelos de Aprendizado Profundo nesse tipo de dispositivo apresentam vantagens em relacão à Computação em Nuvem, como (*i*) redução da banda de comunicação, (*ii*) redução de custo de computação na nuvem, (*iii*) redução do tempo de resposta e (*iv*) privacidade de dados (DENG, 2019).

Diante do exposto, seriam desejáveis modelos de Inteligência Artificial para detecção de *Deepfake*, capazes de atuar em dispositivos de recursos limitados como telefones celulares. Não obstante essa relevância, são muitos os desafios para conciliar a demanda de recursos necessários para inferência com redes neurais profundas e os recursos limitados em *smartphones*, uma vez que o desempenho de um modelo nesses dispositivos deve considerar não apenas a acurácia, mas também uma combinação de memória, latência e consumo de energia (DENG, 2019).

Neste projeto, pretende-se investigar arquiteturas de redes profundas para detecção de *Deepfake* viáveis para ambientes com recursos limitados.

Espera-se, outrossim, identificar ou adaptar arquiteturas existentes para detecção de imagens falsas e/ou manipuladas, com potencial para uso em dispositivos móveis. Em geral, a produtização de modelos de *Deep Learning* em dispositivos com recursos limitados constitui-se num grande desafio, uma vez que esses modelos costumam ser representados com milhões de parâmetros, demandando muito dos recursos de memória e processamento para inferência.

Diante do exposto, a seguinte questão de pesquisa emerge, a qual norteará este projeto:

Q1 *"Atualmente, existem modelos para detecção de imagens e/ou vídeos falsos ou manipulados de face com potencial para implantação e produtização em telefones celulares?"*

Definem-se os seguintes objetivos para o desenvolvimento deste trabalho, em busca de resposta à questão de pesquisa:

- Explorar modelos e técnicas de treinamento em laboratório para a tarefa de detecção de imagens e vídeos falsos de face, avaliando sua viabilidade para dispositivos com recursos limitados de *hardware*, em termos de números de parâmetros e tamanho.

Esse objetivo está relacionado à questão de pesquisa Q1.

- Analisar o desempenho obtido em relação aos modelos estado da arte em *benchmarks*. Esse objetivo está relacionado à questão de pesquisa Q1.

Pretende-se obter um modelo, ainda que adaptado, capaz de alcançar desempenho competitivo e passível de implantação em dispostivos com recursos limitados.

2 FUNDAMENTOS TEÓRICOS

Neste capítulo, são discutidos os impactos de *Deepfake*, listados tipos de ataque, descritas técnicas de criação e detecção, apresentadas as principais bases de dados e modelos estado da arte para detecção, assim como modelos candidatos para inferência em dispositivos com recursos limitados. Ainda, são introduzidos tópicos relevantes dos fundamentos das áreas de Redes Neurais e Aprendizado Profundo, essenciais ao contexto de *Deepfake*.

2.1 *Deepfake*

Deepfake é um conteúdo falso gerado por Inteligência Artificial, que parece verdadeiro sob o escrutínio de olhos humanos. Esse termo é derivado da combinação dos termos *Deep Learning* e *Fake*, em alusão à geração de conteúdo por redes neurais artificiais (MIRSKY; LEE, 2021). A manipulação de imagens de humanos está entre as aplicações mais frequentes, em que a mídia sintética resultante é produto da substituição de um rosto numa imagem ou video existente pelo de outra pessoa (ZHANG *et al.*, 2021), consequentemente exibindo uma reencenação com conteúdo fictício, como se fosse real (WEERAWARDANA; FERNANDO, 2021). Vale salientar que *Deepfake* não deve ser confundido com Aprendizado de Máquina Adversarial[1], posto que o objetivo deste último é o de enganar máquinas, enquanto o objetivo do primeiro é o de enganar seres humanos (MIRSKY; LEE, 2021).

Essa tecnologia surgiu em 2017, quando um usuário do portal *Reddit*[2], sob o codinome *Deepfakes* (ZHANG *et al.*, 2021), utilizou uma fita de vídeo pública, motores de busca de imagens e o *framework Tensorflow*[3] para produzir videos pornográficos forjados com faces de celebridades (MIRSKY; LEE, 2021) e publicá-los em portais de mídia social (WEERAWARDANA; FERNANDO, 2021). No ano seguinte, o portal *Buzzfeed*[4] lançou um video *Deepfake*, produzido com o *software FakeApp* do usuário do *Reddit*, em que o ex-presidente Barack Obama fazia uma palestra a respeito e levantava questões concernentes a roubo de identidade, imitações e a propagação de desinformação em mídias sociais (MIRSKY; LEE, 2021).

Aplicações criativas e produtivas de *Deepfake* são realidade, por exemplo, dublagem artificial de filmes estrangeiros (ROETTGERS, 2019), reanimação de personagens históricas para fins educacionais e modelos digitais personalizadas para campanhas de

[1] *Adversarial Machine Learning*
[2] https://www.reddit.com/
[3] http://www.tensorflow.org/
[4] https://www.buzzfeed.com/

moda (DIETMAR, 2019). Ainda, frequentemente vídeos forjados retratam situações cômicas, em que um indivíduo aparece atuando de uma maneira que normalmente não faria. Apesar do fim aparentemente ingênuo, tais reencenações podem acarretar constrangimento ao seu alvo. Algumas aplicações extrapolam os limites da comédia, assumindo um viés malicioso e cobrindo um espectro de ilicitudes que se estende desde fraudes financeiras até videos pornográficos envolvendo celebridades (WEERAWARDANA; FERNANDO, 2021).

Os impactos sociais desse tipo de aplicação são nefastos, do que decorre uma série de implicações legais, uma vez que infringem direitos de imagem e direitos de propriedade intelectual, culminando com prejuízos de ordem econômica e ataque à reputação. No limite, vídeos falsificados sobre personagens políticos podem induzir uma crise na mídia, ameaçar a estabilidade social e a segurança nacional (ZHANG *et al.*, 2021).

2.2 Categorias de Ataques *Deepfake*

Mirsky and Lee (2021) identificaram quatro categorias de *Deepfake* dentro do contexto de falsificações visuais de faces humanas, a saber reconstituição[5], substituição[6], edição[7] e síntese[8], ressaltando não haver objetivos claros de ataque relacionados às duas últimas.

A seguir, descrevemos a categorização proposta por aqueles autores, salvo menção contrária, seguindo sua notação e denotando por s e t as identidades de origem e alvo, respectivamente, x_s e x_t como as imagens que representam aquelas identidades e x_g como a imagem gerada a partir das identidades s e t. A Figura 1 exemplifica pictoricamente a categorização ora descrita.

2.2.1 Reconstituição

Em *Deepfake* por Reconstituição, a imagem de origem x_s é usada para guiar a expressão, boca, olhar, posição da cabeça ou pose do corpo na imagem alvo x_t, como detalhado na Tabela 1.

2.2.2 Substituição

Em *Deepfake* por Substituição, o conteúdo da imagem alvo x_t é substituído pelo conteúdo da imagem de origem x_s, sob a restrição de preservar a identidiade de x_s, conforme detalhado na Tabela 2.

---

[5] *reenactment*
[6] *replacement*
[7] *editing*
[8] *synthesis*

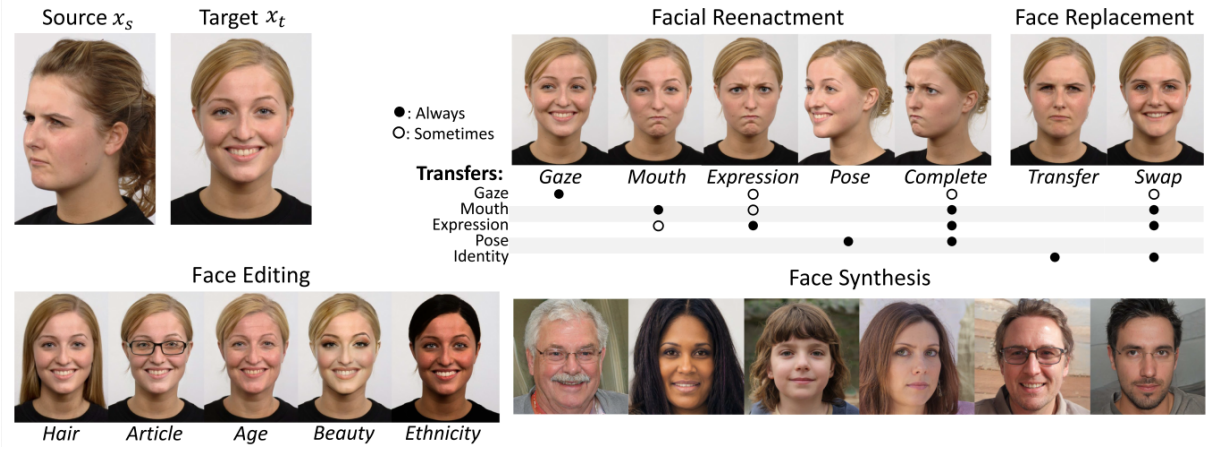


Figura 1 – Exemplos de *Deepfake* de faces humanas por reconstituição, substituição, edição e síntese.

Fonte: (MIRSKY; LEE, 2021).

Tabela 1 – *Deepfake* por Reconstituição.

Componente	Objetivo	Aplicações
Expressão	x_s guia a expressão em x_t	Cinema, *video games* e mídia educacional.
Boca	áudio ou boca em x_s guia a boca em x_t	Dublagem para outro idioma e edição.
Olhar	direção dos olhos e globos oculares em x_s guiam os correspondentes em x_t	Correção de fotografias ou manutenção do contato visual em entrevistas em vídeo.
Cabeça	a posição da cabeça em x_s guia a correspondente em x_t	Frontalização de faces.
Corpo	pose do corpo em x_s é transferida para o corpo em x_t	Síntese de pose humana.
Modelo de Ataque	Imitar uma identidade, controlando suas palavras para fomentar difamação, descrédito, desinformação, adulteração de evidências, falsas evidências, embaraço visando a chantagens e imitação em tempo real.	

Fonte: Elaborada pelo autor.

Nota: Categorização de Mirsky and Lee (2021).

2.2.3 Edição e Síntese

Deepfake por Edição e Síntese representam menores riscos de ataque, quando comparadas às abordagens anteriores, portanto não foram consideradas para os estudos de detecção elencados no levantamento de Mirsky and Lee (2021). Descrições e aplicações são sumarizadas na Tabela 3.

Tabela 2 – *Deepfake* por Substituição.

Técnica	Objetivo	Aplicações
Transferência	o conteúdo de x_s é transferido para x_t	Na indústria da moda, a transferência de faces de modelos em diferentes trajes.
Troca	o conteúdo de x_s substitui e é guiado por x_t	*Face Swap* usado para criar conteúdo cômico, trocando a face de um indivíduo pela de uma celebridade ou para anonimização, em lugar de borramento ou pixelização.
Modelo de Ataque	Pornografia de vingança, em que a face de uma atriz é subtituída pela da vítima, com fins de humilhaçao, difamação ou mesmo chantagem.	

Fonte: Elaborada pelo autor.

Nota: Categorização de Mirsky and Lee (2021).

Tabela 3 – *Deepfake* por Edição e Síntese.

Técnica	Objetivo	Aplicações
Edição	os atributos da imagem alvo x_t são alterados, removidos ou acrescentados	Mudança de características como roupas, cabelos, pelos faciais, idade, peso, beleza e etnia.
Síntese	uma imagem x_g é criada sem um alvo x_t	técnicas de síntese produzem faces e corpos sem direitos autorais para cinema e jogos, mas também podem ser usadas para criar personagens falsas.

Fonte: Elaborada pelo autor.

Nota: Categorização de Mirsky and Lee (2021).

2.3 Criação de *Deepfake*

O processo de geração de uma imagem *Deepfake* x_g por Reconstituição ou Substituição, de forma geral, pode ser organizado em três ou quatro estágios. Zhang *et al.* (2021) reconhecem três estágios nesse processo, a saber (*i*) Reconhecimento de Faces, (*ii*) Substituição de Faces e (*iii*) Pós-processamento de faces, enquanto que Mirsky and Lee (2021) dividem o processo em (1) Detecção e recorte de face, (2) Extração de representações intermediárias, (3) Geração guiada de uma face por outra face e (4) Fusão da face gerada sobre a imagem alvo, como exibido na Figura 2.

2.3.1 Detecção e Recorte de Faces

Inicialmente, algum sistema de detecção de faces pode ser utilizado para localizar a face na imagem, determinar seu tamanho e efetuar seu recorte.

2.3.2 Extração de Representações Intermediárias

Sobre a região localizada, operações tradicionais de Processamento de Imagens, tais como conversão para níveis de cinza, equalização de histograma, normalização, redução de ruído e filtragens podem ser aplicadas. Por fim, vetores de características visuais, caracte-

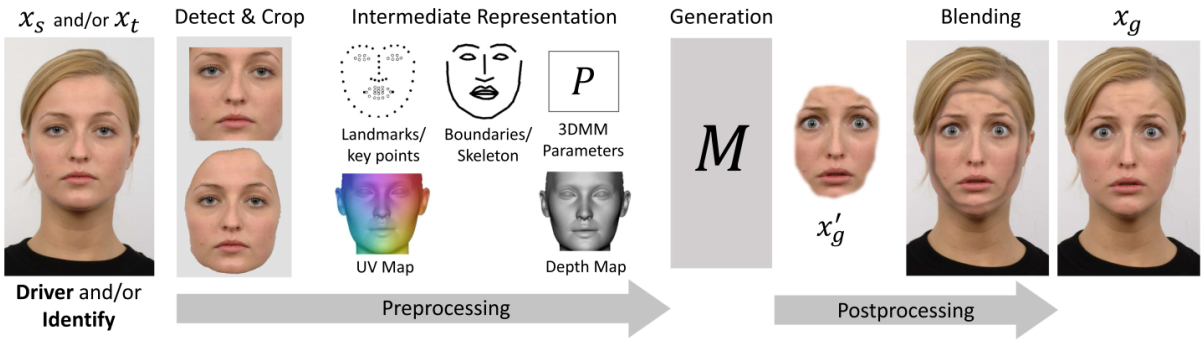


Figura 2 – Fluxo de Geração de *Deepfake* de faces humanas por reconstituição e substituição.

Fonte: (MIRSKY; LEE, 2021).

rísticas estatísticas ou características obtidas a partir de coeficientes de transformações podem ser obtidos como representação das faces, usadas finalmente para efetuar consultas e casamentos com padrões em bases de dados (ZHANG *et al.*, 2021).

2.3.3 Geração Guiada De Uma Face Por Outra Face

Neste estágio, uma face original é convertida em uma face forjada alvo. Para tanto, *Deepfake* utiliza tipicamente arquiteturas baseadas no modelo de *Autoencoder*, ilustrado na Figura 3 capaz de reconstruir imagens de entrada e composta por uma rede codificadora (*encoder*) e duas redes decodificadoras (*decoder*). A rede codificadora aprende os padrões de características comuns a todas as faces humanas, a partir das imagens originais e imagens alvo. As redes decodificadoras, por sua vez, identificam a individualidade de cada face, aprendendo a gerar as faces originais e faces alvo separadamente (ZHANG *et al.*, 2021; ZI *et al.*, 2020; LI *et al.*, 2020).

2.3.4 Fusão Da Face Gerada Sobre A Imagem Alvo

A fim de eliminar artefatos e corrigir distorções, como diferenças de tom de pele, diferenças de laminação, bordas da face e fundo complexo, são efetuadas operações de pós-processamento a fim de integrar adequadamente a face gerada à imagem de fundo da cena original (ZHANG *et al.*, 2021).

2.4 Redes Neurais

Em sua versão simplificada, um sistema *Deepfake* treina uma rede neural denominada *autoencoder* com imagens de um par de indivíduos, enquanto duas outras redes neurais chamadas decodificadores reconstroem as imagens originais. Durante a inferência,

os dois decodificadores trocam as faces, concluindo o processo de manipulação. Esse conceito é ilustrado de forma simplificada na Figura 3.

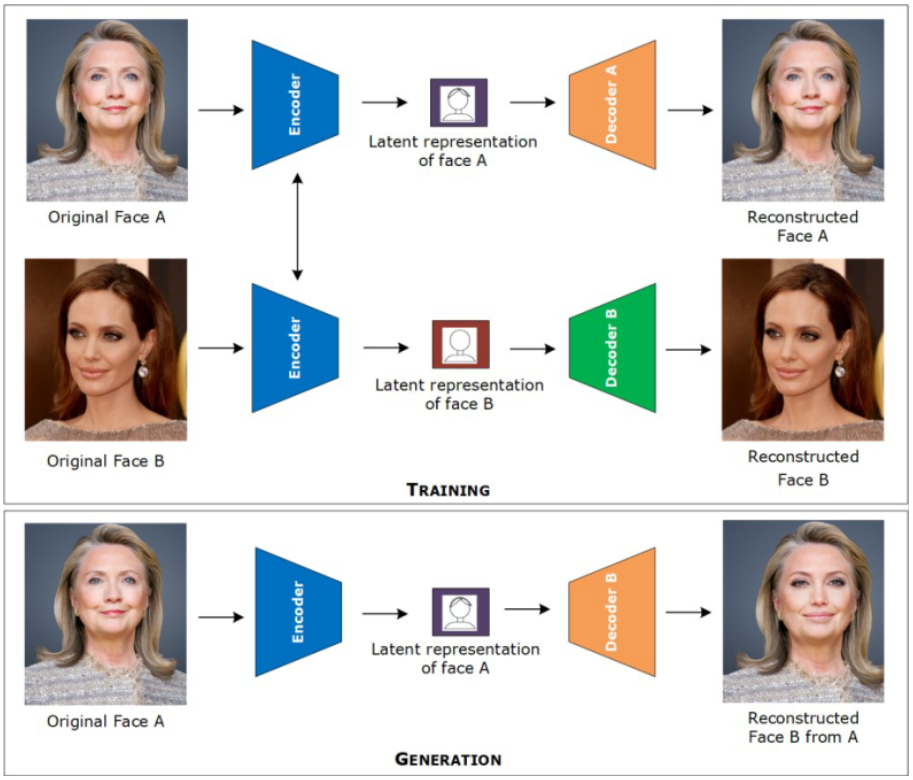


Figura 3 – Criação de *Deepfake* usando um *autoencoder* e um *decoder*.

Fonte: (MASOOD *et al.*, 2021).

Redes neurais artificiais são eficientes aproximadores de funções não-lineares (GOODFELLOW; BENGIO; COURVILLE, 2016), que podem ser usadas para diversas tarefas envolvendo predição e geração de conteúdo a partir de uma entrada. Essas redes são compostas por camadas de neurônios, os quais são interligados por sinapses. No processo de propagação progressiva[9], uma entrada n-dimensional x é propagada pelas camadas da rede e o resultado da operação é não-linearmente transformado por alguma função de ativação, como Sigmóide, Tangente Hiperbólica ou Unidade Linear Retificada (ReLU)[10]. Cada camada é ativada pela saída da camada anterior, sucessivamente, até que a última camada produza o resultado. Uma rede neural pode ser treinada conforme algum paradigma, como Aprendizado Supervisionado, Aprendizado Não-Supervisionado e Aprendizado Autossupervisionado, dentre outros. No contexto de aprendizado supervisionado, por exemplo, uma rede neural é treinada sobre uma base de dados pareada, ou seja, entrada e respectivo rótulo[11], enquanto é minimizada uma função de perda[12] diferenciável,

9 *forward propagation*
10 *Rectified Linear Unit*
11 também chamado anotação
12 *loss function or objective function*

que quantifica a proximidade entre valores alvo e valores preditos, mediante o processo denominado retropropagação[13], que implementa a Regra da Cadeia para diferenciação. Os pesos são apropriadamente atualizados por algoritmos de otimização baseados no Gradiente Descendente, a partir do gradiente determinado no passo anterior. Uma vez treinada, a rede aprende a aproximar a função desejada e realizar predições sobre dados desconhecidos, uma propriedade chamada de generalização (MIRSKY; LEE, 2021).

Para tanto, algumas condições previstas pela teoria de Aprendizado de Máquina devem ser satisfeitas: (i) o conjunto de treinamento deve ser estatisticamente representativo da população de interesse, o que é expresso matematicamente pela condição de suas amostras terem sido obtidas aleatoriamente a partir de uma mesma distribuição de probabilidades, sendo então independentes e identicamente distribuídas; (ii) assume-se alguma suposição *a priori*, denominada Viés Indutivo sobre o espaço de representações para uma família de funções (Viés de Restrição) e/ou sobre a estratégia de uma hipótese que minimize a função de perda (Viés de Preferência) (MITCHELL, 1997).

As arquiteturas mais utilizadas em Redes Neurais para geração de *Deepfake* são combinações de Redes Neurais Convolucionais[14], Redes Generativas Adversariais[15] (WEERAWARDANA; FERNANDO, 2021), Redes Neurais Recorrentes[16], Codificadores-Decodificadores[17] e arquiteturas baseads em *CycleGAN* e *pix2pix* (MIRSKY; LEE, 2021), como ilustrado na Figura 4.

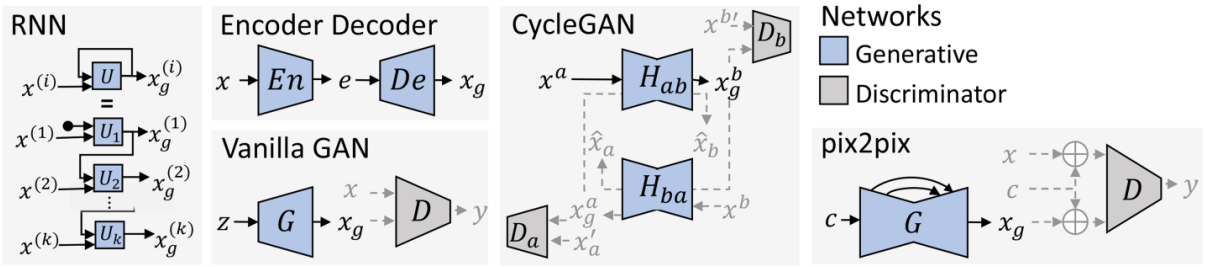


Figura 4 – Arquiteturas Básicas usadas para Geração de *Deepfake* de faces humanas. As linhas representam o fluxo de dados durante treinamento (cinza) e produção (preto). Notação: x: amostra de entrada, x_g: amostra gerada, U_k: camada oculta, En: Codificador, De: Decodificador, $e = En(x)$: codificação ou *embedding*, z: vetor aleatório (ruído), D: Discriminador, G: Gerador, H_{ab}: Gerador de amostra do domínio b, H_{ba}: Gerador de amostra do domínio a, D_a: Discriminador de amostra do domínio a, D_b: Discriminador de amostra do domínio b.

Fonte: (MIRSKY; LEE, 2021).

13 *backpropagation*
14 *Convolutional Neural Networks - CNN*
15 *Generative Adversarial Networks - GAN*
16 *Recurrent Neural Networks - RNN*
17 *Encoder-Decoder - ED*

2.4.1 Redes Neurais Convolucionais

Em Redes Neurais Convolucionais (*CNN*), aprendem-se os filtros que são deslocados sobre tensores durante o processo de convolução, extraindo-se padrões em diferentes níveis conceituais hierárquicos. Os pesos dos filtros aprendidos são compartilhados, permitindo a drástica redução do número de parâmetros em relação às arquiteturas baseadas em redes densas e o aumento dos tamanhos das redes, sem aumento em termos de quantidade de dados para treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). Operações da família de *Pooling* podem ser usadas para subamostrar ou superamostrar mapas de ativação, viabilizando a construção de arquiteturas do tipo Codificador-Decodificador para imagens (MIRSKY; LEE, 2021).

2.4.2 Redes Neurais Recorrentes

Redes Neurais Recorrentes[18] (*RNN*) são redes projetadas para lidar com dados sequenciais e comprimento variável (MIRSKY; LEE, 2021), podendo ser usadas para tarefas como geração de sequências, classificação de sequências e tradução de sequências, entre outras. Essas redes possuem memória levando em conta não somente a amostra atual, mas o estado oculto do sistema que incorpora passos anteriores para o processamento de uma amostra atual (MURPHY, 2022). Por sua natureza, redes *RNN* podem ser usadas para processar dados de vídeo e áudio. Como evoluções de *RNN*, as redes do tipo *LSTM*[19] e *GRU*[20](MIRSKY; LEE, 2021) estendem sua capacidade com células de memória e portas de controle.

2.4.3 Redes Codificadoras-Decodificadoras

Redes Codificadoras-Decodificadoras[21] (*ED*) são formadas por pelo menos duas redes, uma conhecida como a Codificadora, ou *Encoder En*, e a outra como a Decodificadora, ou *Decoder De*, com uma configuração tal que suas camadas estreitam-se em direção ao centro, o que confere a essa arquitetura a propriedade de resumir a entrada $De(En(x)) = x_g$. Chama-se *Encoding* ou *Embedding* a codificação obtida pela transformação $En(x) = e$, dada a distribuição de X e *Espaço Latente* ao espaço formado por $E = En(X)$. Quando a topologia de uma rede neural *ED* é simétrica e a mesma é treinada com o objetivo de reconstruir a entrada, ou seja $De(En(x)) = x$, chama-se essa rede de *Autoencoder*. Em particular, se um *Autoencoder* aprende a distribuição *a posteriori* do *Decoder*, dado o conjunto X, então trata-se de um tipo especial denominado *Variational Autoencoder* (*VAE*). Este último possui a propriedade de desentrelaçar os fatores de variação no espaço

[18] *Recurrent Neural Networks*
[19] *Long-Short Term Memory*
[20] *Gated Recurrent Unit*
[21] *Encoder-Decoder Networks*

latente, produzindo *embeddings* que se comportam melhor sob operações de interpolação ou modificações (MIRSKY; LEE, 2021).

2.4.4 Redes Generativas Adversariais

Uma Rede Generativa Adversarial[22] (*GAN*) é um *framework* de treinamento composto por duas redes, a Geradora G e a Discriminadora D, treinadas de maneira adversarial. A rede G gera amostras falsas x_g para enganar a rede D, que por sua vez aprende a distinguir entre amostras reais ($x \in X$) e amostras falsas ($x_g = G(z)$), onde $z \sim N$ é um vetor amostrado aleatoriamente de uma distribuição normal (MIRSKY; LEE, 2021). Durante o treinamento, a rede D busca maximizar a média do logaritmo das probabilidades de imagens reais e do logaritmo das probabilidades invertidas de imagens geradas falsas (Eq. 2.1) ou equivalentemente minimizar o negativo desta expressão. A rede G busca minimizar a média do logaritmo das probabilidades invertidas das predições de D de imagens geradas falsas (Eq. 2.2), o que equivale a minimizar o contrário do que D busca maximizar. Como essa última função satura, quando o desempenho de G não é satisfatório e D aprende corretamente a discriminar imagens falsas, os gradientes não são suficientes para atualizar os pesos. Por isso, prefere-se maximizar a média do logaritmo das probabilidade das predições de D de imagens geradas falsas, ou equivalentemente minimizar seu negativo (BROWNLEE, 2019).

$$\mathcal{L}_{adv}(D) = \max \log D(x) + log(1 - D(G(z))) \tag{2.1}$$

$$\mathcal{L}_{adv}(G) = \min \log(1 - D(G(z))) \tag{2.2}$$

Em particular, cumpre destacar duas arquiteturas baseadas em *GAN* que são amplamente empregadas na geração de *Deepfake*:

2.4.4.1 *Image-to-Image Translation* (*pix2pix*)

O *framework pix2pix* viabiliza transformações pareadas entre domínios. Durante o treinamento, a rede G gera uma imagem x_g dada uma imagem de contexto visual x_c, enquanto a rede D discrimina entre (x, x_c) e (x_g, x_c). A rede G é uma *U-Net*, ou seja, uma arquitetura de *Encoder-Decoder* com *CNN* (*ED CNN*) e conexões residuais[23] entre *En* e *De* para preservar detalhes da entrada, permitindo a geração de conteúdo em alta resolução (MIRSKY; LEE, 2021).

2.4.4.2 CycleGAN

O *framework CycleGAN* é uma evolução de *pix2pix*, que permite tradução de imagens não pareadas. Duas *GAN* atuam num ciclo, convertendo imagens entre os domínios

[22] *Adversarial Neural Network*
[23] *skip connections*

de entrada e saída, respectivamente, encorajando a consistência por meio da função de perda de consistência de ciclo[24] ($\mathcal{L}_{cyc}$).

2.5 Funções de Perda

Uma função de perda[25] estima de maneira única e global o erro incorrido na tomada de qualquer decisão ou ação disponível, sendo a solução ótima aquela que minimiza a função de perda (BISHOP, 2006).

Diferentes funções de perda são empregadas conforme a tarefa de interesse. Para tarefas de classificação, em que uma rede neural estima probabilidades para as diversas classes, comumente usa-se a função Entropia Cruzada[26]. Para tarefas de regressão, como é o caso de *Deepfake*, funções de perda como as normas $\mathcal{L}_1$ e $\mathcal{L}_2$ são muito usadas. Uma desvantagem dessas funções é a necessidade de amostras pareadas (MIRSKY; LEE, 2021), o que significa que elas devem ser alinhadas. Além disso, para aplicações como Reconstituição, em que x_s encontra-se em posição diferente de x_t, pode haver um deslocamento significativo, que será altamente penalizado pela $\mathcal{L}_2$.

Para transformações entre imagens não-pareadas, podem-se usar funções de perda associadas a um modelo denominado perceptual, frequentemente uma rede de reconhecimento de faces como a *VGG*. A função de perda perceptual, por exemplo, $\mathcal{L}_{\text{perc}}$, compara diretamente mapas de ativação em camadas escondidas daquele modelo e seu resultado pode ser interpretado como a medida da diferença semântica entre as contrapartidas de x_g. Alternativamente, emprega-se a função perda de casamento de características[27] $\mathcal{L}_{FM}$, que compara as saídas da última camada do modelo perceptual, portanto conceitos de alto nível semântico. A função de perda de conteúdo[28] $\mathcal{L}_{\mathtt{C}}$, por sua vez, compara os mapas de ativação apenas da imagem gerada x_g (MIRSKY; LEE, 2021).

2.6 Detecção de *Deepfake*

Por um lado, dado o grau de realismo que as falsificações produzidas por *Deepfake* alcançaram, sua identificação a olho nu constitui-se em tarefa deveras desafiadora. Por outro lado, o rampante aprimoramento das tecnologias de geração de *Deepfake* aliado ao seu crescente uso para fins excusos urgem o desenvolvimento de técnicas efetivas para sua detecção (WEERAWARDANA; FERNANDO, 2021).

Com vistas a essa demanda, a academia e setores da indústria têm buscado soluções efetivas, de maneira independente ou colaborativa, como a parceria entre a gigante *Meta*[29]

[24] *Cycle Consistency Loss*
[25] *Loss Function*
[26] *Cross Entropy*
[27] *Feature Matching Loss*
[28] *Content Loss*
[29] anteriormente *Facebook*

e algumas universidades, dentre as quais *MIT*, *University of Oxford*, *UC Berkeley* e *University of Maryland*. Dessa colaboração, resultou a criação da competição *Deepfake Detection Challenge - DFDC*[30] no ano de 2020, quando apresentaram um sistema de detecção que alcançou cerca de 82% de acurácia (WEERAWARDANA; FERNANDO, 2021). Outra iniciativa nesse sentido, foi o lançamento da competição *Media Forensics Challenge 2018 - MFC2018*[31] pelo *NIST*.

Zheng, Zhang and Thing (2019) publicaram uma extensa revisão sobre adulteração e sua detecção em imagens reais de maneira geral. O levantamento de Mirsky and Lee (2021) concentrou-se em manipulações realizadas em imagens de face, segundo o qual as técnicas de detecção podem ser agrupadas em técnicas específicas conforme artefatos ou abordagens indiretas.

2.6.1 Técnicas Específicas Conforme Artefatos

- **Fusão**[32] (Espacial): artefatos que surgem durante o estágio de fusão da face gerada à imagem original de fundo. Métodos baseados em detecção de bordas, medidas de qualidade e análise de frequência.

- **Ambiente**[33] (Espacial): artefatos relacionados à inconsistência entre o conteúdo gerado e o original podem ocasionar anomalias indicando conteúdo gerado, como resíduos relativos a processos de transformações geométricas[34], iluminação e variação de fidelidade. Em geral, esse tipo de artefato é identificado, usando *CNN* para comparar regiões de objeto e fundo ou usando *ED* para codificar partes da face e contexto e alimentar um classificador com a diferença entre essas codificações e a codificação de toda a imagem.

- **Análise Forense**[35] (Espacial): marcas únicas, padrões sutis como impressões digitais são deixados por *GAN*, detectáveis mesmo na presença de compressão e ruído. Nesse sentido, métodos baseados na análise da diferença entre o padrão de frequência de uma câmera pode ser usado para detecção de conteúdo falso colado. Outros métodos concentram-se nos resíduos, usando redes *ED* para codificar a imagem original e uma versão melhorada por bancos de filtros como o *LoG*[36], que alimentam uma *LSTM*, responsável por classificar uma sequência de vídeo a partir de quadros. Outras abordagens focam-se em imperfeições, em vez de resíduos, usando redes neurais

30 https://ai.facebook.com/datasets/dfdc/
31 https://www.nist.gov/itl/iad/mig/media-forensics-challenge-2018
32 *Blending*
33 *Environment*
34 *Warping*
35 *Forensics*
36 *Laplacian of Gaussian*

para enfatizar os ruídos e suprimir imperfeições, como um pré-processamento para o classificador.

- **Comportamento**[37] (Temporal): padrões e anomalias no comportamento de algum alvo podem ser identificados e modelados a partir de grandes quantidades de dados em vídeos gravados. Outra abordagem sem vídeos de referência é buscar discrepâncias entre medidas sobre emoções extraídas das sequências de áudio e vídeo, usando redes Siamesas.

- **Fisiologia** (Temporal): o fato de que conteúdo gerado é destituído de sinais fisiológicos tem fomentado o surgimento de técnicas de detecção de *Deepfake* bem sucedidas, baseadas na monitoração de frequência cardíaca, pulso e padrões irregulares no piscar de olhos.

- **Sincronização** (Temporal): quando a sincronização em ataques com vídeos dublados é explorada, evidenciam-se inconsistências ao correlacionar a fala aos pontos fiduciais em torno da boca ou mesmo entre visemas (formato da boca) e fonemas falados, em particular aqueles em que a boca está totalmente fechada.

- **Coerência** (Temporal): artefatos decorrentes de incoerência temporal podem indicar a presença de conteúdo falso, por exemplo, usando uma *RNN* ou *LSTM* para detectar efeitos de *flicker* e *jitter* na região da face, monitorando o fluxo óptico ou estimando o erro de reconstrução do próximo *frame* por uma *LSTM*.

2.6.2 Abordagens Indiretas

Alternativamente, nesta categoria de estratégias, redes neurais são encarregadas de encontrar as características relevantes para análise, seguindo basicamente duas abordagens.

- **Classificação**: arquiteturas baseadas em *CNN* têm sido usadas efetivamente para detectar vídeos *Deepfake*, como redes Siamesas treinadas sobre exemplos de imagens reais e falsas. Na arquitetura de Redes de Memórias Hierárquicas[38](*HMN*), a região de face é codificada e processada por uma *GRU* bidirecional com um mecanismo de atenção. Um módulo de memória compara esse *encoding* aos vistos recentemente em memória e realiza uma predição. *Ensembles* de classificadores usando *CNN* produzem resultados mais robustos e redes *CNN* $3D$ que realizam convoluções espaço-temporais sobre múltiplos *frames* superam os métodos de detecção por *frames* individuais.

- **Detecção de Anomalias**: nessa abordagem, redes neurais são treinadas em dados reais e usadas em produção para detectar *Deepfake* como anomalias. Isso pode ser

[37] *Behavior*
[38] *Hierarchical Memory Network*

alcançado, medindo-se as ativações de uma rede para reconhecimento de faces, em vez de analisar *pixels* brutos. Em outra abordagem, redes do tipo *VAE* são treinadas para reconstruir imagens reais. Anomalias são detectadas computando o Erro Quadrático Médio (*MSE*) entre os componentes médios da imagem codificada e da imagem reconstruída. Também é possível comparar imagens de entrada com imagens reais projetadas no espaço latente por uma rede *ED*.

2.7 Alguns *Datasets* e *Benchmarks* em Manipulação de Imagens

Algumas bases de dados e respectivos *benchmarks* são listadas na Tabela 4, onde constata-se uma ampla gama de técnicas de manipulação digital de imagens de faces, em particular o *Deepfake*.

Tabela 4 – *Datasets*.

Dataset	Origem	Videos	Técnica de geração	Referência
`Celeb-DF`	*Youtube*	5.639	Basic Deepfake Maker	(LI *et al.*, 2020)
`DeepForensics-1.0`	*Youtube*	60.000	DF-VAE	(JIANG *et al.*, 2020)
`WildDeepfake`	*Youtube*	707	Encoder-Decoder	(ZI *et al.*, 2020)
`FaceForensics++`	*Youtube*	1.000	*Deepfakes*, *FaceSwap*, *Face2Face* e *NeuralTextures*	(ROSSLER *et al.*, 2019)
DFDC Preview	*Actors*	4.119	Unknown	(DOLHANSKY *et al.*, 2019)
UADFV	*Youtube*	49	FakeApp	(LI; CHANG; LYU, 2018)
Deepfake Dataset	*Internet*	19.456	*Deepfake*	(AFCHAR *et al.*, 2018)

Fonte: Elaborada pelo autor.

2.8 O Estado da Arte

Segundo Weerawardana and Fernando (2021), até a presente data, não há um método preciso para detecção de *Deepfake*, não obstante o progresso de métodos baseados em *Deep Learning*. Apresentamos o estado da arte em detecção de *Deepfake*, nas modalidades de Reconstituição e Substituição, exibindo os resumos recentemente publicados nos levantamentos de Mirsky and Lee (2021) e Tolosana *et al.* (2020), nas Figuras 8, 7 e 9.

Além desses e outros trabalhos listados em levantamentos e revisões na literatura, vale salientar o destaque que os primeiros colocados no desafio *Deepfake Detection Challenge - DFDC*[39], promovido pela *Meta*, têm recebido quando usados como referência, tal qual um *benchmark*. Em particular, Coccomini *et al.* (2022) empregaram o poder dos *ViT*, ou Transformers Visuais[40], combinados com a rede *EfficientNetB0* e reportaram resultados comparáveis aos de Das *et al.* (2021), o primeiro trabalho colocado naquele desafio.

[39] https://ai.facebook.com/datasets/dfdc/
[40] *ViT: Visual Transformers*

A despeito da acurácia desses modelos, índices de desempenho sobre latência e memória também devem ser considerados dentre os critérios para seleção de modelos candidatos à implantação em dispositivos com recursos limitados.

Model	**AUC**	**F1-score**	**# params**
ViT with distillation [18]	0.978	91.9%	373M
Selim EfficientNet B7 [37]†	0.972	90.6%	462M
Convolutional ViT [39]	0.843	77.0%	89M
Efficient ViT (our)	0.919	83.8%	109M
Conv. Cross ViT Wodajo CNN (our)	0.925	84.5%	142M
Conv. Cross ViT Eff.Net B0 - Avg (our)	0.947	85.6%	101M
Conv. Cross ViT Eff.Net B0 - Voting (our)	0.951	88.0%	101M

† Uses an ensemble of 6 networks.

Figura 5 – Comparação de métodos estado da arte para detecção *Deepfake* usando *ViT* e *EfficientNet*.

Fonte: (COCCOMINI *et al.*, 2022).

Como mostrado na Figura 5, o menor dos modelos para detecção de *Deepfake*, nessa comparação, chega a um total de 89 milhões de parâmetros (COCCOMINI *et al.*, 2022). A complexidade do modelo medida em número de parâmetros treináveis sugere a quantidade mínima necessária de memória de *GPU*. A memória total alocada pode ser estimada pela quantidade de memória necessária para o número de parâmetros de um modelo e para o processamento de dados em lote. Durante a inferência, computações intermediárias para ativações e mapas de características[41] demandam grande quantidade de memória, o que inviabiliza muitos modelos candidatos para implantação em dispositivos com memória limitada (LIU *et al.*, 2020). A complexidade em termos de custo computacional pode ser medida não somente pelo número de *FLOP*[42], ou multiplicações-adições (BIANCO *et al.*, 2018), mas também pelo custo de acesso à memória (MA *et al.*, 2018), ou *MAC*[43].

41 *feature maps*
42 *Floating-Point Operations*
43 *Memory Access Cost*

2.9 A arquitetura `MesoNet`

Na contramão dos modelos pesados, emerge o modelo denominado `MesoNet` proposto por Afchar *et al.* (2018), figurando entre os métodos estado da arte na compilação de Tolosana *et al.* (2020) (ver Figura 7 e Figura 8). Naquele trabalho, `MesoNet` é apresentada em duas versões de arquitetura minimalista, a saber, `Meso-4` e `MesoInception-4` com 27.977 e 28.615 parâmetros treináveis respectivamente, reportando altas taxas de acerto de detecção, ou seja, acima de 98% e 95% para as técnicas de geração denominadas *Deepfake* e *Face2Face*.

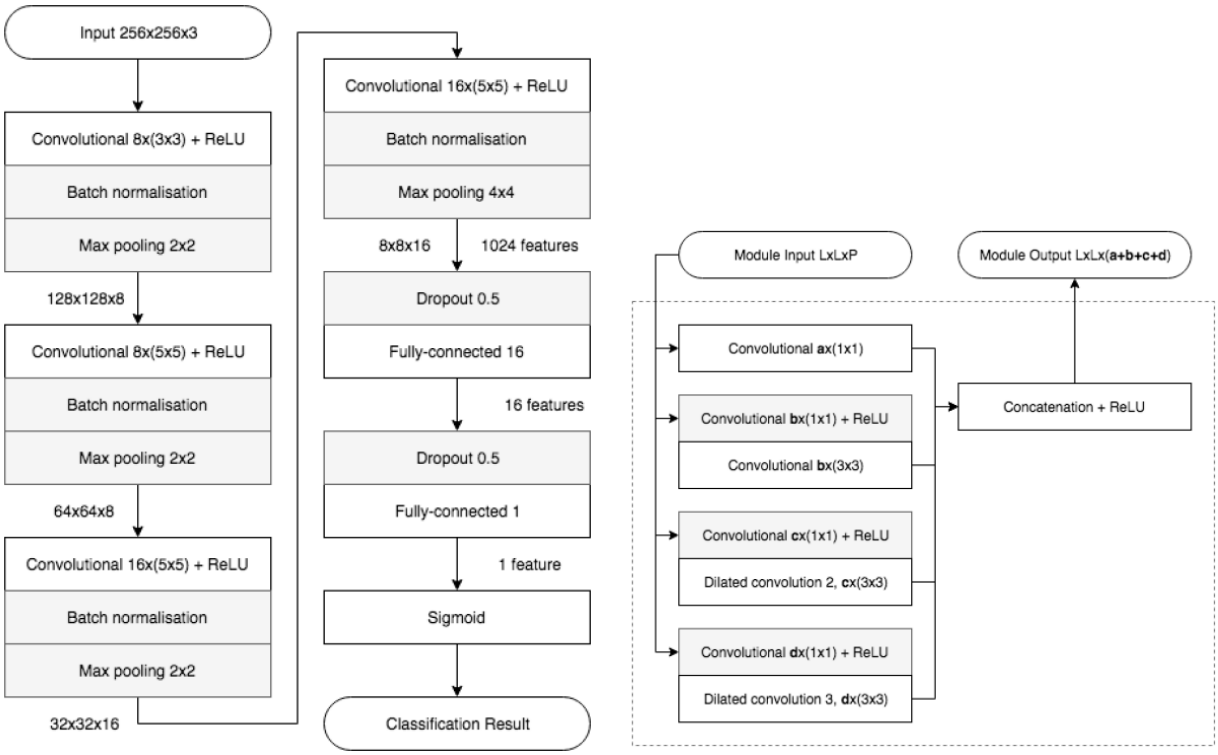
Afchar *et al.* (2018) observaram que, por um lado, o processo de compressão degrada o ruído em imagens oriundas de vídeos falsificados, inviabilizando a análise baseada em ruído com características de baixo nível, a que se referem como microscópicas. Por outro lado, a análise em nível macroscópcio, ou semântico, é extremamente desafiadora até mesmo para olhos humanos, em particular para imagens de rosto. Considerando o exposto, aqueles autores propuseram um método de análise em nível intermediário, isto é, a análise em nível mesoscópico, mediante uma rede neural profunda, mas com número reduzido de camadas: a arquitetura `MesoNet`. As arquiteturas representadas na Figura 6 foram reportadas nesse estudo como as que obtiveram melhor desempenho para a tarefa de classificação de imagens falsas, a partir da simplificação gradual de arquiteturas mais complexas, sob a restrição de manter o mesmo desempenho.

Em sua versão básica, ilustrada na Figura 6(a), a `Meso-4` é constituída por quatro blocos com camadas de convolução e um bloco com camadas densas, usando a função de ativação *ReLU* para introduzir não-linearidade às transformações, Normalização em *Batch*[44] para mitigar o Desaparecimento do Gradiente[45] e *Dropout* como estratégia de regularização, a fim de evitar o superajuste[46] do modelo.

No mesmo trabalho, Afchar *et al.* (2018) propuseram a arquitetura alternativa denominada `MesoInception-4` (Figura 6(b)), substituindo os dois primeiros blocos de camadas convolucionais da arquitetura básica por uma variante do bloco conhecido como *Inception*, originário da *InceptionNet* (SZEGEDY *et al.*, 2015), com convoluções dilatadas[47] de filtro reduzido, a fim de introduzir informação multiescala, evitando o nível semântico. Neste estudo, ficou demonstrado que a substituição de mais de duas camadas pelo bloco *Inception* não melhora o desempenho.

---

44 *Batch Normalization*
45 *Vanishing Gradient*
46 *Overfitting*
47 *Dilated Convolutions*



(a) (b)

Figura 6 – Arquiteturas `MesoNet`: (a) `Meso-4` e (b) `MesoInception-4`.

Fonte: (AFCHAR *et al.*, 2018).

Study	Features	Classifiers	Best Performance	Databases (Generation)
Afchar *et al.* (2018) [53]	Mesoscopic Level	CNN	Acc. = 83.2%	FF++ (Face2Face, LQ)
			Acc. = 93.4%	FF++ (Face2Face, HQ)
			Acc. = 96.8%	FF++ (Face2Face, RAW)
			Acc. ≃ 75%	FF++ (NeuralTextures, LQ)
			Acc. ≃ 85%	FF++ (NeuralTextures, HQ)
			Acc. ≃ 95%	FF++ (NeuralTextures, RAW)
Rössler *et al.* (2019) [6]	Image-related Steganalysis	CNN	Acc. ≃ 91%	FF++ (Face2Face, LQ)
			Acc. ≃ 98%	**FF++ (Face2Face, HQ)**
			Acc. ≃ 100%	**FF++ (Face2Face, RAW)**
			Acc. ≃ 81%	FF++ (NeuralTextures, LQ)
			Acc. ≃ 93%	**FF++ (NeuralTextures, HQ)**
			Acc. ≃ 99%	**FF++ (NeuralTextures, RAW)**
Matern *et al.* (2019) [57]	Visual Artifacts	Logistic Regression, MLP	AUC = 86.6%	FF++ (Face2Face, RAW)
Nguyen *et al.* (2019) [58]	Image-related	Autoencoder	EER = 7.1%	FF++ (Face2Face, HQ)
			EER = 7.8%	FF++ (NeuralTextures, HQ)
Stehouwer *et al.* (2019) [7]	Image-related	CNN + Attention Mechanism	**AUC = 99.4%** **EER = 3.4%**	**FF++ (Face2Face, -)**
Amerini *et al.* (2019) [101]	Inter-Frame Dissimilarities	CNN + Optical Flow	Acc. = 81.6%	FF++ (Face2Face, -)
Sabir *et al.* (2019) [60]	Image + Temporal Information	CNN + RNN	**Acc. = 94.3**	**FF++ (Face2Face, LQ)**

Figura 7 – Comparação entre métodos estado da arte de detecção *Deepfake* por Reconstituição.

Fonte: (TOLOSANA *et al.*, 2020).

Study	Features	Classifiers	Best Performance	Databases
Zhou *et al.* (2018) [52]	Image-related Steganalysis	CNN SVM	*AUC = 85.1%*	*UADFV*
			AUC = 83.5%	*DeepfakeTIMIT (LQ)*
			AUC = 73.5%	*DeepfakeTIMIT (HQ)*
			AUC = 70.1%	*FF++ / DFD*
			AUC = 55.7%	***Celeb-DF***
Afchar *et al.* (2018) [53]	Mesoscopic Level	CNN	Acc. = 98.4%	Own
			AUC = 84.3%	*UADFV*
			AUC = 87.8%	*DeepfakeTIMIT (LQ)*
			AUC = 62.7%	*DeepfakeTIMIT (HQ)*
			Acc. ≃ 90.0%	FF++ (DeepFake, LQ)
			Acc. ≃ 94.0%	FF++ (DeepFake, HQ)
			Acc. ≃ 98.0%	FF++ (DeepFake, RAW)
			Acc. ≃ 83.0%	FF++ (FaceSwap, LQ)
			Acc. ≃ 93.0%	FF++ (FaceSwap, HQ)
			Acc. ≃ 96.0%	FF++ (FaceSwap, RAW)
			AUC = 53.6%	*Celeb-DF*
Korshunov and Marcel (2018) [1]	Lip Image - Audio Speech Image-related	PCA+RNN PCA+LDA, SVM	**EER = 3.3%** **EER = 8.9%**	**DeepfakeTIMIT (LQ)** **DeepfakeTIMIT (HQ)**
Güera and Delp (2018) [54]	Image + Temporal Information	CNN + RNN	Acc. = 97.1%	Own
Yang *et al.* (2019) [55]	Head Pose Estimation	SVM	AUC = 89.0%	UADFV
			AUC = 55.1%	*DeepfakeTIMIT (LQ)*
			AUC = 53.2%	*DeepfakeTIMIT (HQ)*
			AUC = 47.3%	*FF++ / DFD*
			AUC = 54.8%	*Celeb-DF*
Li *et al.* (2019) [56]	Face Warping Artifacts	CNN	**AUC = 97.4%**	**UADFV**
			AUC = 99.9%	**DeepfakeTIMIT (LQ)**
			AUC = 93.2%	**DeepfakeTIMIT (HQ)**
			AUC = 79.2%	*FF++ / DFD*
			AUC = 53.8%	*Celeb-DF*
Rössler *et al.* (2019) [6]	Image-related Steganalysis	CNN	Acc. ≃ 94.0%	FF++ (DeepFake, LQ)
			Acc. ≃ 98.0%	**FF++ (DeepFake, HQ)**
			Acc. ≃ 100.0%	**FF++ (DeepFake, RAW)**
			Acc. ≃ 93.0%	FF++ (FaceSwap, LQ)
			Acc. ≃ 97.0%	**FF++ (FaceSwap, HQ)**
			Acc. ≃ 99.0%	**FF++ (FaceSwap, RAW)**
Matern *et al.* (2019) [57]	Visual Artifacts	Logistic Regression MLP	AUC = 85.1%	Own
			AUC = 70.2%	*UADFV*
			AUC = 77.0%	*DeepfakeTIMIT (LQ)*
			AUC = 77.3%	*DeepfakeTIMIT (HQ)*
			AUC = 78.0%	*FF++ / DFD*
			AUC = 48.8%	*Celeb-DF*
Nguyen *et al.* (2019) [58]	Image-related	Autoencoder	*AUC = 65.8%*	*UADFV*
			AUC = 62.2%	*DeepfakeTIMIT (LQ)*
			AUC = 55.3%	*DeepfakeTIMIT (HQ)*
			AUC = 76.3%	*FF++ / DFD*
			EER = 15.1%	FF++ (FaceSwap, HQ)
Stehouwer *et al.* (2019) [7]	Image-related	CNN + Attention Mechanism	AUC = 99.4% EER = 3.1%	DFFD
Dolhansky *et al.* (2019) [51]	Image-related	CNN	**Precision = 93.0%** **Recall = 8.4%**	**DFDC Preview**
Agarwal and Farid (2019) [59]	Facial Expressions and Pose	SVM	AUC = 96.3%	Own (FaceSwap, HQ)
Sabir *et al.* (2019) [60]	Image + Temporal Information	CNN + RNN	**AUC = 96.9%** **AUC = 96.3%**	**FF++ (DeepFake, LQ)** **FF++ (FaceSwap, LQ)**

Figura 8 – Comparação entre métodos estado da arte de detecção *Deepfake* por Substituição.

Fonte: (TOLOSANA *et al.*, 2020).

			Type	Modality			Content			Method			Eval. Dataset							Performance*		
			Reenactment	Replacement	Image	Video	Audio	Feature	Body Part	Face Image	Model	Indicates Affected Area	Input Resolution	DeepfakeTIMIT [86]	DFFD [149]	FaceForensics [130]	FaceForensics++ [131]	FFW [82]	Celeb-DF [101]	Other Deepfake DB	Custom	ACC	EER	AUC
Classic ML	[187]	2017		●	●	●				●	SVM-RBF		250x250								●	92.9		
	[4]	2017		●	●	●				●	SVM		*							●			18.2	
	[178]	2018	●	●	●	●				●	SVM		*							●				0.97
	[86]	2018	●	●		●				●	SVM		128x128	●									3.33	
	[42]	2019		●	●					●	SVM, Kmeans...		1024x1024								●	100		
	[8]	2019	●	●	●	●				●	SVM		*	●									13.33	
	[6]	2019	●	●		●				●	SVM		*								●			0.98
Deep Learning	[111]	2018	●	●	●	●				●	CNN		256x256								●	99.4		
	[97]	2018	●	●				●	●		LSTM-CNN		224x224								●			0.99
	[119]	2018	●	●	●	●				●	Capsule-CNN		128x128			●						99.3		
	[17]	2018		●	●	●				●	ED-GAN		128x128								●	92		
	[39]	2018	●	●	●					●	CNN		1024x1024							●				0.81
	[63]	2018	●	●		●				●	CNN-LSTM		299x299								●	97.1		
	[106]	2018	●	●	●					●	CNN		256256								●	94.4		
	[33]	2018	●	●	●	●				●	CNN AE		256x256			●					●	90.5		
	[3]	2018	●	●		●				●	CNN		256x256								●			0.99
	[132]	2019	●	●		●				●	CNN-LSTM		224x224				●					96.9		
	[118]	2019	●	●	●	●				●	CNN-DE	●	256x256			●	●					92.8	8.18	
	[38]	2019		●	●	●				●	CNN		-								●	98.5		
	[41]	2019	●	●	●	●				●	CNN AE GAN	●	256x256				●				●	99.2		
	[149]	2019	●	●	●	●				●	CNN+Attention	●	299x299		●								3.11	0.99
	[98]	2019	●	●	●	●				●	CNN		128x128							●	●			0.99
	[101]	2019		●		●				●	CNN		*						●					0.64
	[52]	2019	●	●	●	●				●	CNN+HMN		224x224			●	●	●				99.4		
	[92]	2019	●	●	●	●				●	FCN		256x256				●					98.1		
	[177]	2019	●	●	●	●				●	CNN		128x128								●	94.7		
	[161]	2019	●	●	●	●				●	CNN		224x224							●		86.4		
	[153]	2019	●	●	●					●	CNN		1024x1024								●			94
	[30]	2019	●	●		●				●	CNN		128x128			●					●	96		
	[99]	2019	●	●	●	●				●	CNN	●	224x224	●									93.2	
	[11]	2019	●	●		●				●	CNN		224x224				●					81.6		
	[?]	2019	●			●	●	●	●		LSTM		*								●		22	
	[47]	2019	●			●	●	●	●		LSTM-DNN		*								●		16.4	
	[25]	2019	●			●				●	CNN		256x256								●	97		
	[180]	2019	●	●	●			●		●	CNN		128x128			●		●			●	99.6	0.53	
	[166]	2019	●	●	●	●				●	SVM+VGGnet		224x224				●					85		
	[94]	2019	●	●	●	●		●			CNN	●	64x64		●		●		●	●				99.2
	[95]	2020	○	●	●					●	HRNet-FCN	●	64x64		●		●			●			20.86	0.86
	[96]	2020	●	●	●					●	PP-CNN		-	●		●				●				0.92
	[123]	2020	●	●	●					●	ED-CNN		299x299				●		●	●				0.99
	[108]	2020	●	●		●				●	ED-LSTM		224x224				●		●					
	[167]	2020	●	●	●					●	CNN ResNet		224x224				●				●	Avrg.	Prec.=	0.93
	[64]	2020	●	●	●					●	AREN-CNN		128x128			●					●	98.52		
	[110]	2020	●	●		●	●	●		●	ED-CNN		*	●	●					●				0.92
	[5]	2020	●	●		●	●		●		CNN		128x128							●		89.6		
	[10]	2020	●	●		●				●	LSTM		256x256				●					94.29		
	[69]	2020	●	●	●					●	Siamese CNN		64x64								●	TPR=0.91		
	[129]	2020	●	●	●					●	Ensemble		224x224				●					99.65		1.00
	[36]	2020	●	●		●				●	*		112x112						●			98.26		99.73
	[81]	2020	●	●	●					●	OC-VAE		100x100				●					TPR=0.89		
	[51]	2020	●	●	●					●	ABC-ResNet		224x224								●		?	
Statistics & Steganalysis	[85]	2018		●		●				●	PRNU		1280x720								●	*TPR=1*	*FPR=*	*0.03*
	[150]	2019		●		●				●	Statistics	●	-								●			
	[107]	2019	●	●	●					●	PRNU		*								●	90.3		

*Only the best reported performance, averaged over the test datasets, is displayed to capture the 'best-case' scenario.

Figura 9 – Comparação entre métodos estado da arte de detecção *Deepfake* por Reconstituição.

Fonte: (MIRSKY; LEE, 2021).

2.10 A arquitetura `MesoNet` e as Abordagens Híbridas em Inteligência Artificial

As vantagens e desvantagens das abordagens de Visão Computacional com Aprendizagem de Máquina Clássica e de Aprendizagem Profunda são bem conhecidas. Os modelos de Aprendizagem Profunda costumam produzir resultados com maior acurácia, além de serem muito versáteis. Entretanto, esses modelos demandam recursos computacionais de alto custo (WALSH *et al.*, 2019) e, em sua maioria, são modelos do tipo caixa-preta (MOLNAR, 2022), representando um desafio, no que diz respeito a sua Interpretabilidade[48]. Os métodos de Visão Computacional Clássicos, por sua vez, geralmente são constituídos por modelos caixa-branca (transparentes) otimizados para desempenho e eficiência energética.

Abordagens híbridas combinam Visão Computacional Clássica e Aprendizagem Profunda, explorando os pontos fortes de ambas e têm demonstrado a capacidade de melhorar o desempenho de métodos de Visão Computacional Clássica e lidar com problemas não apropriados para Aprendizagem Profunda (WALSH *et al.*, 2019).

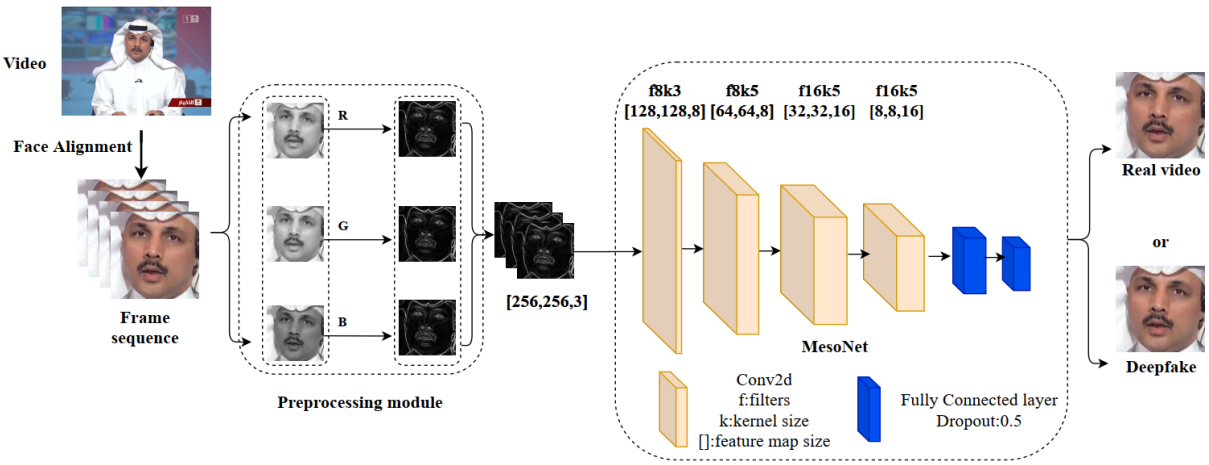


Figura 10 – Módulo de Pré-processamento sobre a entrada para a `Meso-4`.

Fonte: (Xia *et al.*, 2022).

Seguindo essa vertente e combinando técnicas específicas com abordagens indiretas detalhadas na Seção 2.6, Xia *et al.* (2022) propuseram uma adaptação à `MesoNet`, conforme Figura 10, acoplando a esta um módulo de pré-processamento para remover componentes de baixa frequência, retendo altas frequências, haja vista a perda de informação nas características da imagem em consequência dos algoritmos de compressão de vídeo. De acordo com esse estudo, o método foi extensivamente testado sobre as bases de dados `FaceForensics++` e `Celeb-DF`, superando outros métodos estado da arte sobre esta última base de imagens.

Em suma, o módulo de pré-processamento na proposta de Xia *et al.* (2022) é composto pela detecção de faces presentes na imagem de entrada, seguidas de uma filtragem

[48] *Interpretability* ou *Explainability*

passa-alta, com o objetivo de manter a informação com alto poder discriminatório de textura, a partir da observação de que, em geral, a área do rosto é suave em imagens de face produzidas por *Deepfake*. Para tanto, o seguinte operador foi definido:

$$p_{i,j} = \sqrt{(m_{i,j} - m_{i+1,j})^2 + (m_{i,j} - m_{i,j+1})^2} \tag{2.3}$$

onde, $i \in 1, \ldots, H, j \in 1, \ldots, W$, $m_{i,j}$ são os valores dos *pixels* na imagem original, $p_{i,j}$ são os valores dos *pixels* na imagem processada, H e W representam altura e largura da imagem, respectivamente.

Como resultado do pré-processamento, em áreas suaves na imagem, a diferença entre *pixels* será pequena, enquanto a diferença será grande em áreas fortemente texturizadas.

Motivado pela observação de que a Equação 2.3 é similar à aproximação de um operador diferencial, no presente trabalho foi proposta a exploração de um dos operadores diferenciais conhecidos na literatura de Processamento de Imagens e Visão Computacional, que exerça função similar e produza resultados competitivos ou superiores.

3 DESENVOLVIMENTO: MATERIAIS E MÉTODOS

Neste capítulo, serão descritos o método e as arquiteturas propostas, discutidos os critérios para a escolha da base de dados para treinamento e teste dos modelos, listadas as métricas de avaliação dos modelos em termos de taxas de acerto e erro, bem como desempenho no dispositivo móvel. Ainda, serão apresentadas as tecnologias utilizadas para o desenvolvimento deste trabalho e os resultados obtidos.

3.1 *Introdução*

O objetivo de estudar e adaptar arquiteturas viáveis para dispositivos com restrições de recursos de *hardware*, detalhado no Capítulo 1, o sucesso de abordagens híbridas de Visão Computacional e Aprendizagem Profunda, aliados aos resultados promissores reportados sobre a arquitetura `Meso-4` (AFCHAR *et al.*, 2018) e sua variante (Xia *et al.*, 2022), motivaram a consideração da arquitetura `Meso-4` como forte candidata em resposta à questão de pesquisa enunciada neste trabalho.

Baseado na observação de que o operador proposto por Xia *et al.* (2022) e definido na Equação 2.3 assemelha-se à aproximação de um operador diferencial por diferenças de primeira ordem, conjecturamos:

Conjectura 3.1.1 *Operadores diferenciais que atuam como detectores de bordas podem oferecer resultados equivalentes ou melhores que o operador proposto por Xia et al. (2022).*

Em Processamento Digital de Imagens, o operador Gradiente é conhecido justamente pela propriedade de enfatizar altas frequências, sendo aproximado por diferenças discretas de primeira ordem em algoritmos de detecção de bordas baseados em gradiente (JÄHNE, 2002). Diversos algoritmos para aproximação desses operadores foram propostos, amplamente conhecidos como detectores de bordas, tais como os filtros de *Roberts*, *Canny*, *Prewitt* e `Sobel` (GONZALES; WINTZ, 1987).

Conforme a aplicação, esta propriedade do operador Gradiente apresenta vantagens e desvantagens. Por um lado, Petrou and Petrou (2010) apontam a inadequação de máscaras de `Sobel` para detecção de bordas em imagens com altos níveis de ruído, sugerindo o poder de ênfase que o operador de `Sobel` tem sobre conteúdo com altas frequências, tais como texturas, o que é plenamente adequado para os propósitos deste trabalho. Por outro lado, Ye *et al.* (2022) propuseram a `DuFeNet`, uma arquitetura de rede neural convolucional, que emprega a informação de gradiente para aprender características de bordas para a tarefa de classificação de imagens. O bloco `Ramo de Gradiente`, recebe a saída do filtro de `Sobel` como entrada, que é processada por quatro camadas convolucionais e concatenada

com a saída do `Ramo de Textura`. Esse trabalho concluiu que o gradiente aumenta o viés de forma e melhora as habilidades de aprendizagem do modelo, como estratégia para abordar o viés de textura em *CNN*, apontado em (GEIRHOS *et al.*, 2018).

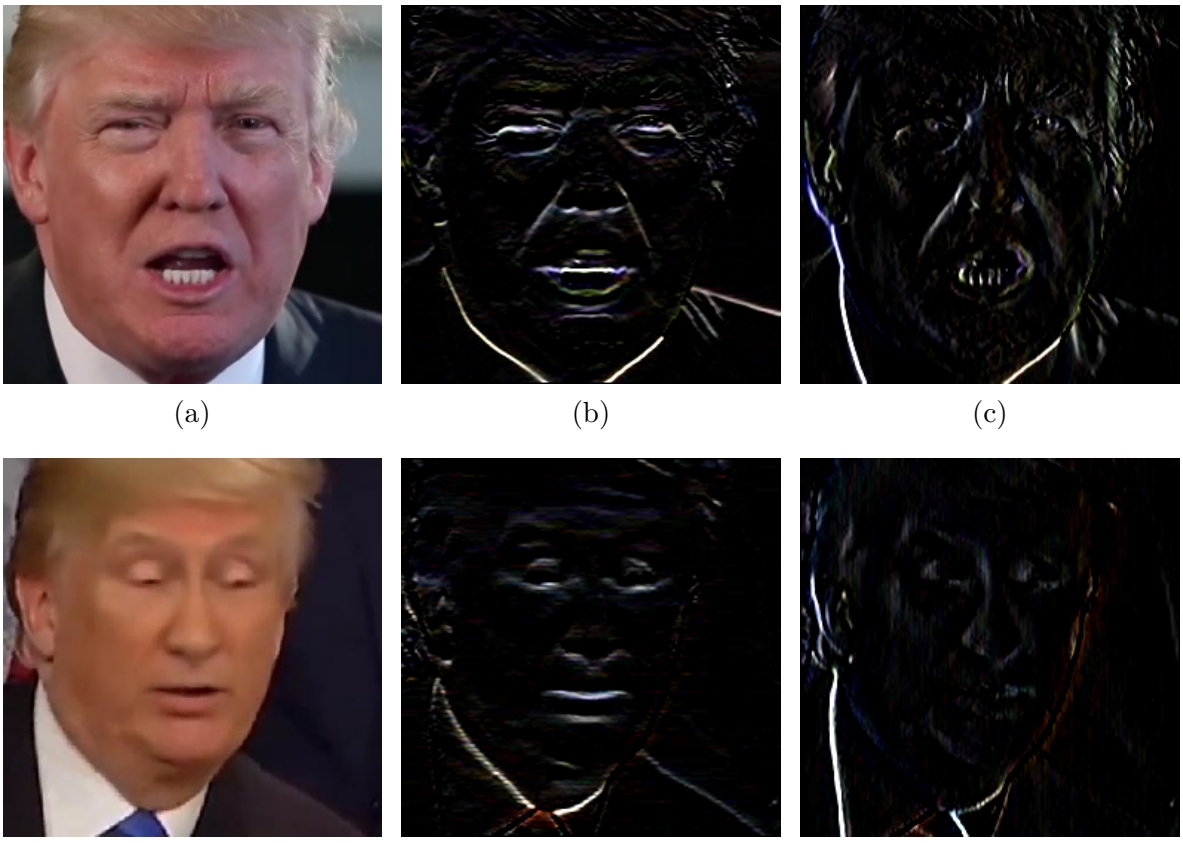
Saliente-se que Kong *et al.* (2021) contestaram a conclusão do trabalho de Geirhos *et al.* (2018) com evidências experimentais de que redes convolucionais não possuem intrinsicamente viés de textura ou forma, o que pode mudar com o viés interno dos dados. Kong *et al.* (2021) mostraram também que as *CNN* obtêm seu conhecimento de forma preguiçosa, no sentido de obter conhecimento de alto nível (forma) somente quando o conhecimento de baixo nível (textura) não for suficiente para satisfazer os requisitos da tarefa.

Diante do exposto, propomos um método para investigação de arquiteturas nas próximas seções, que integrem o conhecimento sobre o `Gradiente`, provendo suporte experimental para a Conjectura 3.1.1.

3.2 *Método Proposto*

Nesta seção, a utilização do operador `Sobel`, como primeira camada não-treinável na arquitetura `Meso-4` é proposta, seguindo a abordagem híbrida de Visão Computacional e Aprendizado Profundo, discutida anteriormente.

A Figura 11 fornece indícios perceptuais que suportam a Conjectura 3.1.1, haja vista a similaridade visual entre os resultados da aplicação do operador de `Sobel` e o resultado do operador de Xia *et al.* (2022), mostrado na Figura 10.



(a) (b) (c)

(d) (e) (f)

Figura 11 – Aplicação do Filtro de `Sobel`. A região de face na imagem falsa é mais suave, enquanto a região de face na imagem real é mais texturizada. O filtro de `Sobel` enfatiza essas diferenças. (a) Imagem real original. (b) Componente horizontal x da filtragem pelo Operador de `Sobel` sobre a imagem real. (c) Componente vertical y da filtragem pelo Operador de `Sobel` sobre a imagem real. (d) Imagem falsa original. (e) Componente horizontal x da filtragem pelo Operador de `Sobel` sobre a imagem falsa. (f) Componente vertical y da filtragem pelo Operador de `Sobel` sobre a imagem falsa.

Fonte: Elaborado pelo autor.

Em processamento de imagens, a magnitude do gradiente é usada para implementar derivadas de primeira ordem. Considere uma imagem $I(x, y)$, seu gradiente definido como

o campo vetorial (GONZALES; WINTZ, 1987):

$$\nabla \mathbf{I} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \tag{3.1}$$

e a magnitude do gradiente:

$$\nabla \mathbf{I} = \left\| \nabla I \right\| = \left[G_x^2 + G_y^2 \right]^{\frac{1}{2}} = \left[\left(\frac{\partial I}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} \right)^2 \right]^{\frac{1}{2}} \tag{3.2}$$

Analogamente, a Magnitude do Gradiente de `Sobel` sobre imagens é definida como:

$$G = \sqrt{G_x^2 + G_y^2} \tag{3.3}$$

onde o operador de `Sobel` calcula as componentes G_x e G_y da derivada de primeira ordem da imagem $I(x, y)$ pelas aproximações:

$$\mathbf{G_x} = \begin{bmatrix} 1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I(x, y) \qquad \mathbf{G_y} = \begin{bmatrix} 1 & 0 & 1 \\ 2 & 0 & 2 \\ 1 & 0 & 1 \end{bmatrix} * I(x, y) \tag{3.4}$$

Observe que o filtro da magnitude de `Sobel` é uma combinação não-linear dos operadores de `Sobel` nas direções x e y.

O *framework* `Tensorflow`, atualmente em sua versão 2.8.2, oferece uma implementação para `Sobel` em seu pacote para tratamento de imagens, a saber `tf.image.sobel_edges`[1]. O uso desse operador para a construção de uma camada e seu acoplamento à arquitetura `Meso-4` será detalhado na Seção 3.6.

[1] https://www.tensorflow.org/api_docs/python/tf/image/sobel_edges

3.3 *Arquiteturas*

Para integrar o bloco `Sobel` à rede `Meso-4`, foram propostas três arquiteturas, ilustradas nos diagramas da Figura 12. Na figura, veem-se as representações das três redes convolucionais projetadas e estudadas, suas componentes, entradas e saídas. O acoplamento do bloco `Sobel` à (a) rede original `Meso-4` foi arquitetado em três configurações distintas, são elas: (b) como camada inicial, (c) como camada somada à entrada e (d) como camada concatenada à entrada.

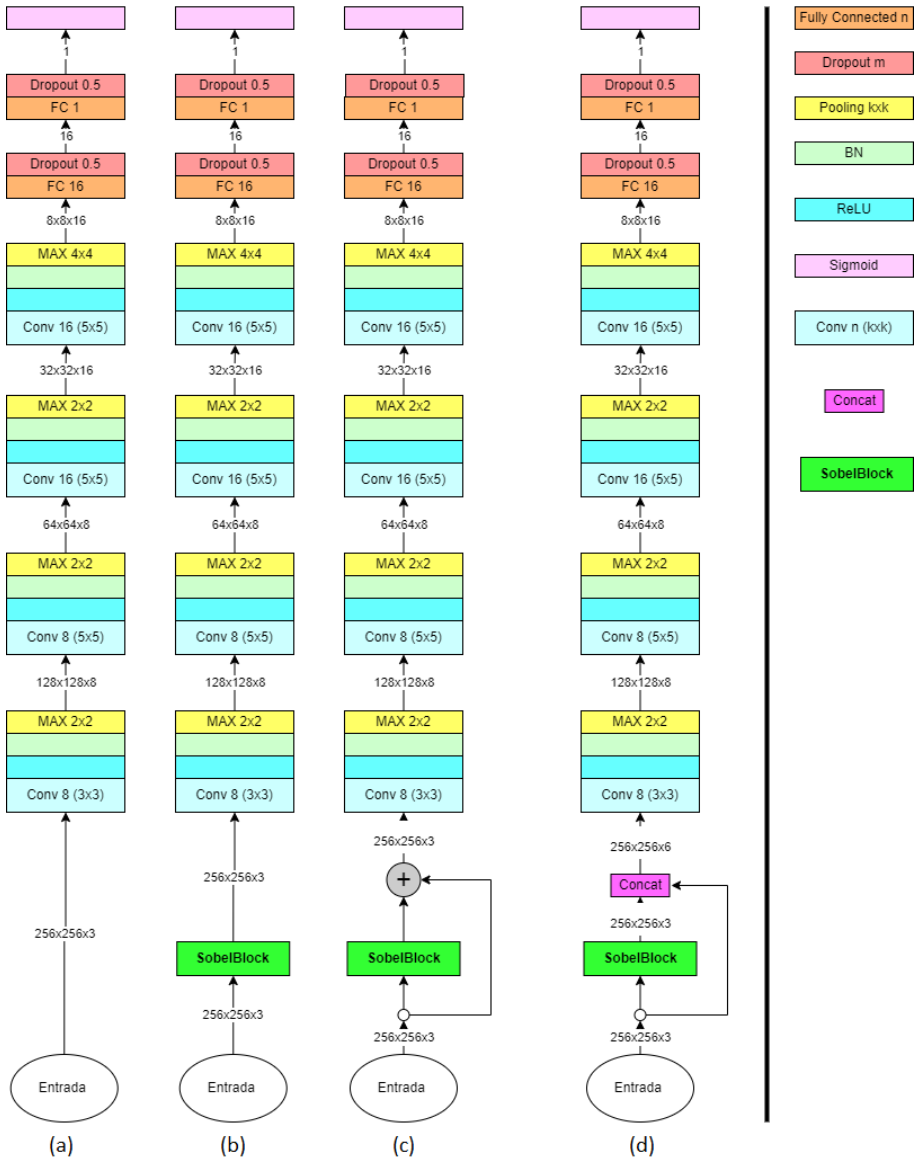


Figura 12 – Arquitetura `Meso-4` e arquiteturas propostas incorporando a camada `Sobel`. Da esquerda para a direita, (a) `Meso-4`: arquitetura original, (b) Arquitetura `MesoNetSobel`: camada `Sobel` na entrada da rede, (c) `MesoNetSobelAdd`: camada `Sobel` somada à entrada da rede, (d) `MesoNetSobelConcat`: camada `Sobel` concatenada à entrada da rede.

Fonte: Elaborado pelo autor.

3.4 *Dados*

Algumas das bases de dados mais populares na literatura foram listadas na Tabela 4, da Seção 2.7, revelando a ampla variedade em termos de origem de conteúdo, quantidade de vídeos e diferentes técnicas de falsificação.

Para a escolha da base de dados a ser usada no presente trabalho, os seguintes critérios foram considerados:

- tamanho;
- facilidade de acesso;
- licença;
- tempo estimado para treinamento, validação e teste;
- recursos oferecidos pelo serviço de hospedagem e execução em nuvem disponível durante este estudo.

Apesar da imensa maioria das bases de dados serem passíveis de liberação para pesquisa no âmbito acadêmico e sem fins lucrativos, mediante solicitação formal via cadastro, identificação do solicitante e sua instituição, esse processo pode tornar-se um tanto burocrático, demorado e com nuances, por vezes, restritivas. A título de exemplo, citamos a base de dados `FaceForensics++`, cujos autores liberam seu código sob a licença *MIT*, porém mantêm seus dados sob um termo próprio de uso[2], em que a cláusula 6 estabelece o seguinte termo:

> *If Researcher is employed by a for-profit, commercial entity, Researcher's employer shall also be bound by these terms and conditions, and Researcher hereby represents that he or she is fully authorized to enter into this agreement on behalf of such employer.*

Analogamente, o acesso ao *DFDC Preview*, por sua vez, está condicionado à abertura de uma conta *AWS* e aos próprios termos de uso, que inclui uma cláusula semelhante à mencionada anteriormente:

> *If you are agreeing to be bound by the Agreement on behalf of your employer or other entity, you represent and warrant to Facebook that you have full legal authority to bind your employer or such entity to this Agreement. If you do not have the requisite authority, you may not accept the Agreement or access the Materials on behalf of your employer or other entity.*

[2] `FaceForensics` *Terms of Use*

Obviamente, entre questões legais, estratégias de negócio e/ou interesses comerciais, são diversas as razões pelas quais uma empresa pode oferecer objeções a tais condições e eleger seu colaborador como representante, para apoiar seu projeto pessoal.

Considerando o exposto, assim como os extensivos testes da arquitetura `Meso-4` com bases de dados como as supracitadas nos estudos de Afchar *et al.* (2018) e Xia *et al.* (2022), limitamos o escopo deste estudo à comparação de desempenho entre o modelo de base `Meso-4` e as arquiteturas aqui propostas incluindo o bloco `Sobel` para o problema de detecção de imagens falsificadas por técnicas de *Deepfake*. Para tanto, optou-se pela base de dados *Deepfake Dataset*[3], criada por Afchar *et al.* (2018) para o trabalho de desenvolvimento da `Meso-4`.

Essa base pode ser considerada pequena em tamanho, com 19.456 imagens ocupando cerca de 184Mb em arquivo compactado, com acesso via *link* para *download* direto, sem a necessidade de cadastro ou identificação, com código e base de dados liberados sob a licença permissiva *Apache* 2.0[4]. Com essa base de dados, os tempos para treinamento, validação e teste nas máquinas virtuais equipadas com *GPU* da plataforma em nuvem `Google Colab Pro`, somam cerca de duas horas e meia. Esse tempo pode ser considerado razoável para os propósitos de investigação das diversas configurações de arquitetura propostas neste trabalho.

Amostras reais e falsas dessa base de imagens são mostradas na Figura 13.

(a) (b)

Figura 13 – Amostras (a) Reais e (b) Falsas da Base de Dados *Deepfake Dataset* usadas para treinamento, validação e teste dos modelos estudados neste trabalho.

Fonte: Elaborado pelo autor.

A referida base foi construída a partir de vídeos gerados por *Deepfake* em diversos

[3] https://github.com/DariusAf/MesoNet
[4] *Apache License 2.0*

níveis de compressão, com duração de 2 segundos a 3 minutos e resolução (contagem de *pixels*) mínima de 854×480 *pixels*, coletados de diferentes plataformas na *internet*, buscando um balanceamento entre níveis de qualidade de resolução diferentes. A base é constituída por 19.456 recortes quadrados de face de dimensões variadas, totalizando $220Mb$. A quantidade de imagens, assim como a divisão empregada para os subconjuntos de Treinamento, Validação e Teste são detalhadas na Tabela 5.

Tabela 5 – *Deepfake Dataset.*

Classe	Treinamento	Validação	Teste
Real	7.416	1.854	2.238
Falsa	5.036	1.259	1.693

Fonte: Elaborada pelo autor.

Com respeito à Aumentação de Dados[5], visando à reprodutibilidade do original e sua comparação a nossa arquitetura, seguiu-se a mesma estratégia adotada naquele trabalho (AFCHAR *et al.*, 2018), isto é, um *pipeline* composto por módulos com variações de 20% em *zoom*, variações de 15 graus em orientação, *flip* horizontal, variações de 20% em brilho, deslocamento de canais para variações de cor, além da normalização usual no intervalo $[0, 1]$.

[5] Data Augmentation

3.5 *Métricas para Avaliação Experimental*

Detecção de *Deepfake* é um problema típico na categoria de tarefas de classificação binária e, como tal, foi avaliado com métricas clássicas, definidas sobre a Matriz de Confusão (Tabela 6), tais como Acurácia Binária[6] (Eq. 3.5), Precisão[7] (Equação 3.6), Revocação[8] (Equação 3.7) e *F1-score* (Equação 3.8). A Área Sob a Curva (AUC)[9] *ROC*[10] também foi empregada nessa avaliação.

Tabela 6 – Definição da Matriz de Confusão.

		Predição	
		Positivo	Negativo
Real	Positivo	VP	FN
	Negativo	FP	VN

Fonte: Elaborada pelo autor.

$$Acc = \frac{VP + VN}{VP + VN + FP + FN} \tag{3.5}$$

$$Prec = \frac{VP}{VP + FP} \tag{3.6}$$

$$Rec = \frac{VP}{VP + FN} \tag{3.7}$$

$$F1 = 2 \times \frac{Prec \times Rec}{Prec + Rec} \tag{3.8}$$

6 *Binary Accuracy*
7 *Precision*
8 *Recall*
9 *Area Under the Curve*
10 *Receiving Operating Characteristic*

3.6 *Tecnologias Utilizadas*

Conforme ilustração na Figura 14, os métodos aqui descritos foram implementados com a linguagem `Python` em *notebooks* do tipo `Jupyter`, hospedados pelo serviço `Google COLAB PRO` e executados em suas máquinas virtuais equipadas com *GPU*. O *framework* denominado `Tensorflow`, versão 2.8.2, com *API* de alto nível `Keras` foi usado para escrever todo o código implementando os modelos, o arcabouço para treinamento, a validação e teste. Empregou-se o `Tensorboard` para monitoramento das seções de treinamento em tempo real. Todos os experimentos foram registrados e rastreados por meio da plataforma `MLflow` integrada a um servidor remoto no `DagsHub`.

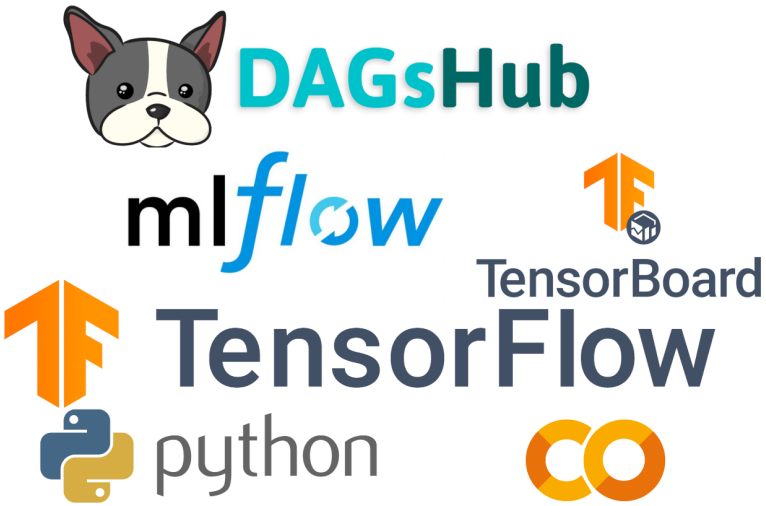


Figura 14 – Tecnologias utilizadas: linguagem `Python`, `Google COLAB PRO`, `Tensorflow 2.8.2`, `Keras`, `Tensorboard`, `MLflow` e `DagsHub`.

Fonte: Elaborado pelo autor.

`MLflow` é uma plataforma de código aberto para gerenciamento de ciclos de vida de ponta-a-ponta em Aprendizagem de Máquina, incluindo os estágios de experimentação, reprodutibilidade, implantação e registro central de modelo (ZAHARIA *et al.*, 2018). Apesar dos benefícios dessa ferramenta, sua (*i*) complexidade de configuração de servidor remoto, eventualmente com controle de acesso, (*ii*) os custos de serviços de nuvem, (*iii*) a falta de flexibilidade de sua interface gráfica para comparação de métricas e parâmetros entre experimentos ou exibição de diversos gráficos na mesma página e (*iv*) os inconvenientes que se impõem para compartilhar o estado de um projeto, como repositório, servidor, e especificação de um experimento estão entre as principais lacunas no `MLflow` (LOUSKY, 2021).

`DagsHub` é uma plataforma *web* baseada em ferramentas de código aberto, otimizada para Ciência de Dados, que pode ser integrada ao `MLflow`, com configuração mínima e preenchendo aquelas lacunas, além de servir como repositório de código, *pipeline* de dados e todos os experimentos do `MLflow` (LOUSKY, 2021).

A implementação de modelos seguiu a *API* Imperativa do `Tensorflow` (TUNG, 2021), também conhecida como *API* de Derivação de Classes[11], recomendada em sua documentação para aplicações em que se faz necessário o desenvolvimento de modelos customizados. Como linhas de base[12], essa implementação explorou (*i*) parte do código oficial[13] do repositório de Afchar *et al.* (2018) e (*ii*) parte do código não-oficial[14] disponibilizado no repositório de Agarwal (2021).

Por um lado, o código em (*i*) implementa blocos, ou conjuntos de camadas e o próprio modelo `Meso-4`, seguindo vagamente a *API* Imperativa, pois não se encontra em plena conformidade com o padrão recomendado pela documentação do `Tensorflow 2`, uma vez que não inclui o método `CALL()`. Esse código implementa algumas funções para a geração da base de imagens, mas não disponibiliza um arcabouço para treinamento e avaliação. Por outro lado, o código em (*ii*) implementa um arcabouço para treinamento e avaliação, no entanto emprega a *API* Funcional, reproduzindo apenas a arquitetura original.

Isto posto, foram necessárias reimplementações, modificações e refatorações de código para obter-se uma implementação do modelo original `Meso-4` descrito em (AFCHAR *et al.*, 2018), usando a *API* Imperativa do `Tensorflow 2`, que permitisse facilmente sua customização para consequente derivação das arquiteturas propostas no presente estudo, como ilustrado em trechos de código na Figura 15, bem como *logging* de métricas e parâmetros para rastreamento dos experimentos no `MLflow`.

[11] *Layers and Models Subclassing*
[12] *baselines*
[13] https://github.com/DariusAf/MesoNet
[14] https://github.com/MalayAgr/MesoNet-DeepFakeDetection

```
[ ] class Meso4SobelConcat(Classifier):
        def __init__(self):
          super(Meso4SobelConcat, self).__init__()
          self.sobelblock = SobelBlock()
          self.convblock1 = ConvBlock(filters=8, kernel_size=3, padding_conv='same', padding_pool='same', pool_scale=2, activation_function = 'relu')
          self.convblock2 = ConvBlock(filters=8, kernel_size=5, padding_conv='same', padding_pool='same', pool_scale=2, activation_function = 'relu')
          self.convblock3 = ConvBlock(filters=16, kernel_size=5, padding_conv='same', padding_pool='same', pool_scale=2, activation_function = 'relu')
          self.convblock4 = ConvBlock(filters=16, kernel_size=5, padding_conv='same', padding_pool='same', pool_scale=4, activation_function = 'relu')
          self.denseblock = DenseBlock(dense_units=16, dropout_rate=0.5, alpha=0.1, activation_function='sigmoid')

        def build(self, input_tensor):
            input_tensor = Input(shape = tf.shape(input_tensor))

        def call(self, inputs_tensor):
          edges = self.sobelblock(inputs_tensor)
          norm_edges = tf.sqrt(tf.reduce_sum(tf.square(edges), axis=-1))
          x01 = Concatenate(axis=-1)([norm_edges, inputs_tensor])
          x1 = self.convblock1(x01)
          x2 = self.convblock2(x1)
          x3 = self.convblock3(x2)
          x4 = self.convblock4(x3)
          y = self.denseblock(x4)

          return y
```

(a)

```
[ ] class SobelBlock(Layer):
      def __init__(self, ):
        super(SobelBlock, self).__init__()

      def build(self, input_tensor):
        input_tensor = Input(shape = tf.shape(input_tensor))

      def call(self, input_tensor):
        sobel = tf.image.sobel_edges(input_tensor)
        return sobel
```

(b)

Figura 15 – Trechos do código implementado, usando a *API* Imperativa do `Tensorflow 2`. (a) Classe para o modelo `MesoNetSobelConcat`, um dos modelos propostos, derivado do modelo `Meso-4`, (b) Classe para o Bloco `Sobel`.

Fonte: Elaborado pelo autor.

3.7 *Configurações e Treinamento*

As rodadas de treinamento foram realizadas em até 50 épocas, com a opção de parada antecipada[15], após uma espera com paciência[16] de 10 épocas sem melhoria na métrica Acurácia Binária, usada para monitorar a convergência no treinamento do modelo. Para avaliação do modelo treinado, foi separado um conjunto de teste com 20% do total de exemplos da base de dados. O restante foi dividido na proporção 80/20 entre treinamento e validação, respectivamente.

O processo de otimização foi executado com lotes[17] de 64 amostras, sobre a função de perda Entropia Cruzada Binária[18] pelo algoritmo *ADAM* em sua configuração padrão de parâmetros, isto é, $\beta_1 = 0.9$ e $\beta_2 = 0.999$, partindo de uma taxa de aprendizagem[19]

15 *Early Stopping*
16 *Patience*
17 *Batches*
18 *Binary Cross Entropy*
19 *Learning Rate*

inicial de 0.001, com cronograma[20] de redução definido como Decaimento Exponencial[21] com uma taxa de decaimento[22] de 0.10 a aproximadamente cada 2400 iterações[23].

Os experimentos e rodadas de treinamento foram rastreados pela plataforma de gerenciamento de ciclos de vida em Aprendizado de Máquina `MLflow`[24] no servidor remoto `DagsHub`[25]. Uma visão do painel no `MLflow` é exibida na Figura 16, demonstrando a facilidade proporcionada por essa ferramenta para o acompanhamento de tempo de treinamento, anotação automática dos resultados das métricas de avaliação em teste e comparação lado a lado entre rodadas com modelos e configurações diferentes.

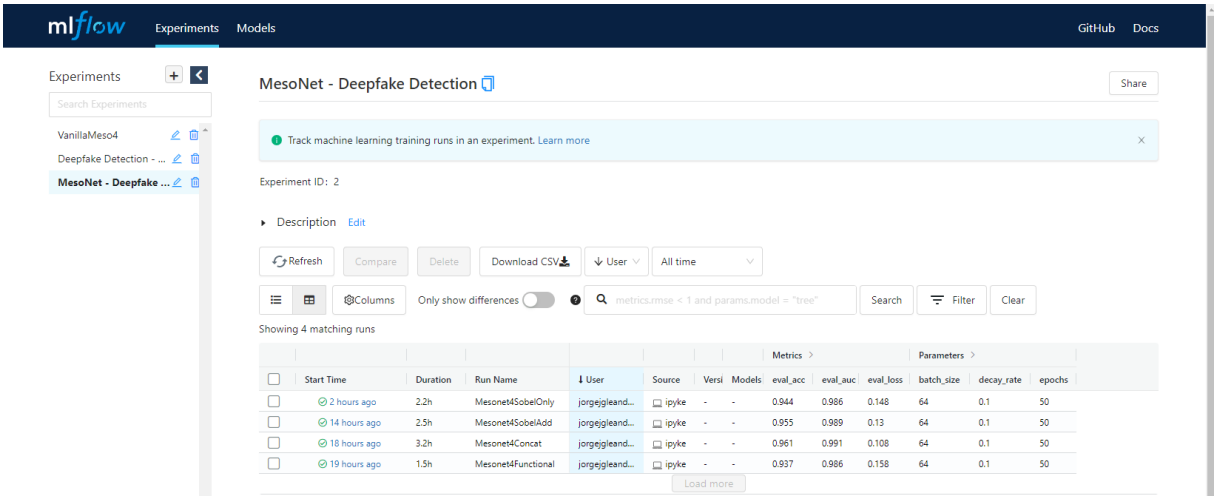


Figura 16 – `MLflow` *Dashboard*: rastreamento de experimentos.

Fonte: Elaborado pelo autor

Pelas curvas de treinamento na Figura 18, observam-se instabilidades durante a validação para épocas na porção média do período de treinamento. No entanto, nota-se convergência no final de cada período, sem aparente ocorrência de superajuste[26] do modelo. Observe que a lacuna entre as curvas de aprendizagem em treinamento e validação não aumenta no final da rodada de treinamento. Ainda, constata-se visualmente que o treinamento para a arquitetura `MesoNetSobelConcat` discorreu de forma mais estável que as demais, durante os processos de validação.

20 *Schedule*
21 *Exponential Decay*
22 *Decay Rate*
23 *Decay Steps*
24 https://mlflow.org/
25 https://dagshub.com
26 Overfitting

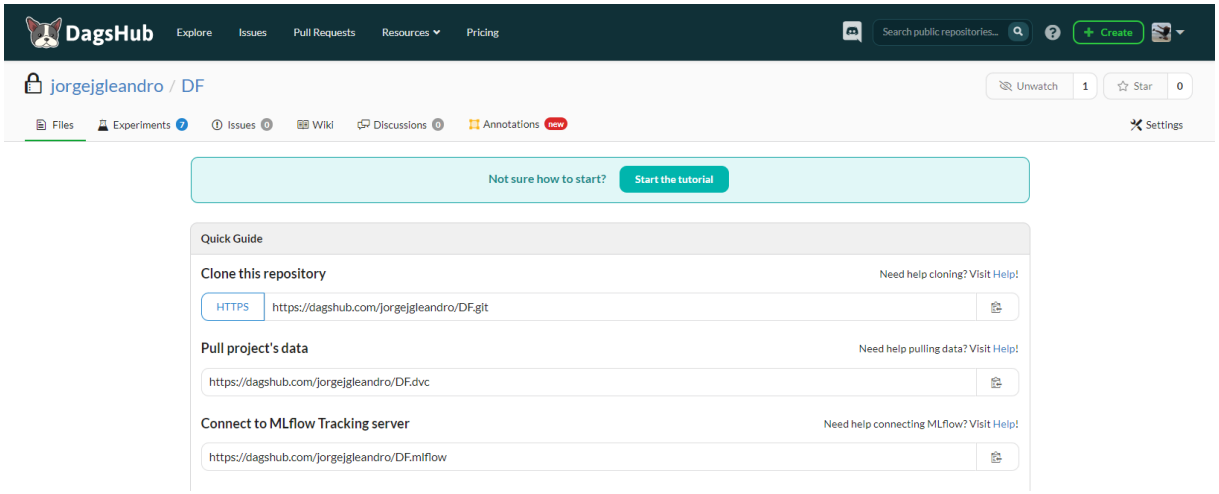


Figura 17 – `DagsHub`.

Fonte: Elaborado pelo autor.

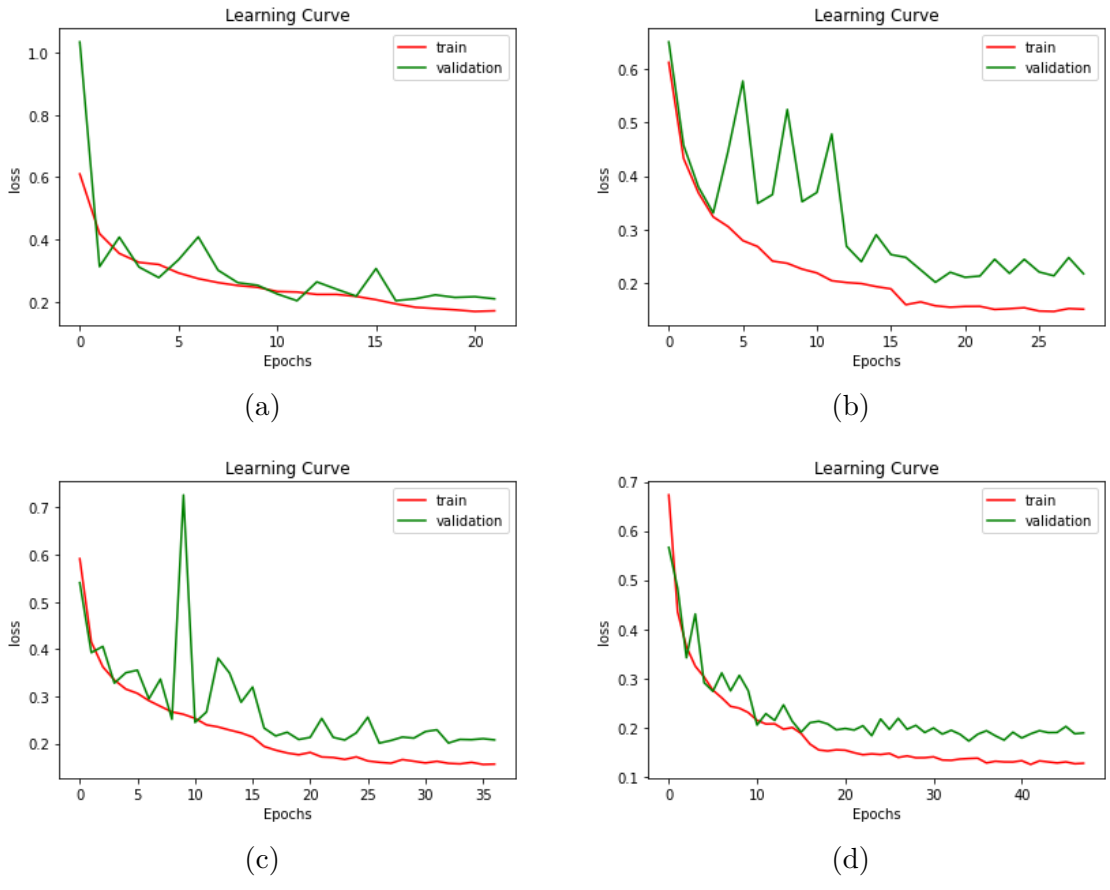


Figura 18 – Curvas de Treinamento para os modelos comparados. (a) `Meso-4`, (b) `MesoNetSobel`, (c) `MesoNetSobelAdd`, (d) `MesoNetSobelConcat`.

Fonte: Elaborado pelo autor.

3.8 *Resultados*

Nesta seção, são apresentados resultados de medições de desempenho em termos de taxas de acerto e erro, bem como desempenho em termos de velocidade de inferência e memória consumida no dispositivo.

3.8.1 *Estudo de Ablação*

No contexto da avaliação de Redes Neurais Artificiais, Estudos de Ablação se referem à avaliação de desempenho de modelos de Inteligência Artificial, estimando a contribuição individual de componentes, mediante sua remoção (MEYES *et al.*, 2019).

No presente estudo, os resultados reportados na Tabela 7 foram obtidos por meio de um Estudo de Ablação, assim como no trabalho de Xia *et al.* (2022), avaliando a contribuição do bloco `Sobel` para a melhoria da arquitetura `Meso-4` em diferentes configurações. O desempenho das configurações é avaliado em termos de níveis de acerto e erro, traduzidos nos valores observados para as métricas definidas sobre a tabela de Confusão.

Tabela 7 – Resultados: Estudo de Ablação comparando a contribuição do bloco `Sobel` sobre a `Meso-4` em diferentes configurações.

Arquitetura	Acurácia	Precisão	Revocação	F1-Score	AUC
`Meso-4`	0,937	0,931	0,962	0,946	0,986
`MesoNetSobel`	0,944	0,949	0,954	0,951	0,986
`MesoNetSobelAdd`	0,955	0,963	0,959	0,961	0,989
`MesoNetSobelConcat`	**0,961**	**0,962**	**0,971**	**0,966**	**0,991**

Fonte: Elaborada pelo autor.

3.8.2 *Resultados de Desempenho no Dispositivo*

Além do critério de desempenho em termos das métricas de avaliação definidas sobre a Matriz de Confusão, para atender a questão de pesquisa enunciada na Seção 1, os modelos candidatos devem satisfazer outrossim critérios objetivos que viabilizem sua implantação[27] em dispositivos embarcados ou móveis, portanto com restrições de recursos de *hardware*.

Para tanto, o modelo foi devidamente convertido para um formato compatível com a *runtime* do `TensorFlow Lite`, ou `TFLite`, responsável por executar operações de inferência com o modelo no ambiente do *Android*. Ainda, o `TFLite` disponibiliza ferramentas que implementam técnicas de compressão de modelo, como Quantização e Poda[28]. O emprego

[27] Deployment
[28] Pruning

destas técnicas pode diminuir o tamanho do modelo em Mb, ou mesmo em número de parâmetros, tornando-o eventualmente mais rápido. Estas técnicas não foram exploradas neste estudo.

A seguir, com o modelo convertido, avaliou-se o mesmo por meio de uma ferramenta de *Benchmark*, disponibilizada pelo `TFLite`. Trata-se de uma ferramenta binária nativa para execução em linha de comando, para estimar:

- tempo de inicialização;
- tempo de inferência no estado de aquecimento[29];
- tempo de inferência em operação estável;
- uso de memória durante o tempo de inicialização;
- memória total usada.

A Figura 19 exibe a captura de tela do terminal em que a ferramenta de *Benchmark* do `TFLite` foi executada. Os dados relevantes da saída dessa ferramenta foram organizados na Tabela 8.

O modelo convertido foi testado num dispositivo telefone celular, da marca `Motorola`, modelo `Edge 20 Lite`, com `Android` 11. Todos os tempos são medidos em microssegundos e correspondem aos tempos de carregamento (*Init*), inicialização do modelo (*Warmup*) e primeira inferência (*First*). O tempo *Inference (avg)* diz respeito ao tempo médio de 50 rodadas de inferência. *Memory (Mb)* informa a variação no consumo de memória antes e depois do modelo ter sido carregado e inicializado na memória *RAM*. O total de parâmetros foi de 28.289.

Tabela 8 – *Benchmark* do Modelo no Dispositivo - Algumas métricas relevantes para avaliação da viabilidade de implantação do modelo considerado em produto.

Model: MesonetSobelConcat - Device: Motorola Edge 20 Lite - Android: 11				
Timing (μs) - CPU				Memory (Mb)
Init	*First*	*Warmup (avg)*	*Inference (avg)*	
46,296	86,695	127,116	108,472	32.969

Fonte: Elaborada pelo autor.

A fim de diminuir o tamanho do modelo em número de parâmetros, tornando-o mais leve, mais rápido e sem perda de acurácia, foi proposta e testada a variante `MesonetSobelConcatDSC`, em que todas as camadas de Convolução foram substituídas

[29] *Warmup state*

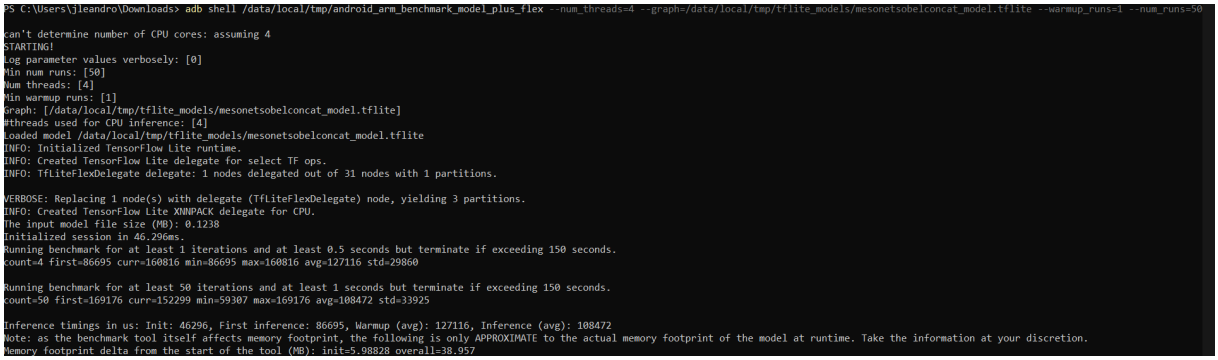


Figura 19 – Saída da ferramenta de *Benchmark* do `TFLite` rodando num telefone celular `Motorola Edge 20 Lite` com `Android 11`.

Fonte: Elaborada pelo autor.

por camadas do tipo *Depthwise Separable Convolution*. Esta camada, introduzida por Chollet (2017), realiza convoluções independentes por canal do mapa de características de entrada, seguidas de uma convolução 1×1 ponto-a-ponto (CHOLLET, 2021). O total de parâmetros da versão adaptada com as camadas *Depthwise Separable Convolution* foi de 18.007. Os resultados do desempenho no dispositivo desta arquitetura modificada são apresentados na Tabela 9.

Tabela 9 – *Benchmark* do Modelo com *Depthwise Separable Convolution* no Dispositivo - Algumas métricas relevantes para avaliação da viabilidade de implantação do modelo considerado em produto.

Model: MesonetSobelConcatDSC - Device: Motorola Edge 20 Lite - Android: 11				
Timing (μs) - CPU				Memory (Mb)
Init	*First*	*Warmup (avg)*	*Inference (avg)*	
$52,328$	$71,163$	$66,499.6$	$70,926.8$	$11,89$

Fonte: Elaborada pelo autor.

4 ANÁLISE DE RESULTADOS

Nesta seção, são analisados e discutidos os resultados apresentados na Seção 3.8.

4.1 Taxas de Detecção

Os resultados reportados na Tabela 7 suportam a Conjectura 3.1.1. As taxas de detecção são medidas mediante métricas usuais para problemas de classificação binária. Observe que os valores medidos para todas as métricas, ou seja, Acurácia, Precisão, Revocação, *F1-Score* e *AUC* para a arquitetura `MesoNetSobelConcat` superaram os valores para as respectivas métricas observadas para a arquitetura básica `Meso-4`. Lembramos que os valores medidos referem-se à detecção em nível de *frame*. Vale salientar que para detecção de *Deepfake* em nível de vídeo, Afchar *et al.* (2018) testaram uma técnica de agregação de *frames*, computando a média das predições ao longo do vídeo. Eles reportam ter aumentado significativamente a taxa de detecção por meio dessa técnica.

Não obstante as observações supracitadas e tendo em vista as diversas fontes de variação aleatória, testes estatísticos podem ser adaptados e usados para medir a significância das diferenças entre métricas de desempenho de diferentes classificadores, como ferramenta de comparação objetiva entre os mesmos (SALZBERG, 1997). A escolha do teste deve ser criteriosa, levando em conta as diversas fontes de variação aleatória, a saber: (i) seleção dos dados de teste, (ii) seleção dos dados de treinamento, (iii) aleatoriedade intrínseca em algoritmos de otimização e (iv) erro aleatório de classificação (DIETTERICH, 1998). Para o problema em questão, o estudo meticuloso de Dietterich (1998) recomenda o *Teste de McNemar*, quando os dados puderem ser processados uma única vez ou o Teste t Pareado por Correlação-Cruzada 5×2, caso contrário. Este último produz resultados melhores e deveria ser priorizado sempre que possível.

Dada a quantidade de dados, optamos pelo *Teste de McNemar* (DIETTERICH, 1998) para validação estatística dos resultados obtidos, cujos resultados são mostrados na Tabela 10. Considerando a hipótese nula $H0 : \mu_d = 0$ e a hipótese alternativa $H1 : \mu_d \neq 0$, a partir do *p-value* $= 0.00294$, somos levados a rejeitar a hipótese $H0$, a um nível de significância $\alpha = 0.05$, donde concluímos que há evidências estatísticas de que o desempenho de `MesonetSobelConcat` é superior ao desempenho de `Meso-4`.

Tabela 10 – Teste de *McNemar* - Validação Estatística dos Resultados.

χ^2	`p-value`
$8,8430$	$0,00294$

Fonte: Elaborada pelo autor.

4.2 Desempenho no Dispositivo

Conforme resultados apresentados na Seção 3.8.2, Tabela 8, o experimento com a ferramenta de *Benchmark* `TFLite` resultou em um tempo médio por inferência de $108,47$ milissegundos. A detecção em nível de *frame* seria factível nesse tempo. Para a detecção em nível de vídeo, uma subsequência de *frames* deveria ser considerada. Nesse caso, Xia *et al.* (2022) sugerem amostrar aleatoriamente um em cada 20 *frames* seguidos, visto o grande número de *frames* similares numa seqüência. Portanto, considerando uma típica taxa média de 30 *frames* por segundo (*FPS*), para o tempo médio por inferência estimado, seria viável e suficiente o processamento de um a cada quatro *frames* para classificar um vídeo.

Conforme os resultados exibidos na Tabela 9 para a arquitetura modificada com camadas *Depthwise Separable Convolution* `MesonetSobelConcatDSC`, com tempo médio por inferência de $70,93$ milissegundos, um a cada três *frames* poderia ser processado.

No que diz respeito ao consumo de memória, os acréscimos de 32.97 Mb ou de $11,89$ Mb podem ser considerados isoladamente ínfimos em ambos os casos, se comparados à capacidade de 6 Gb daquele aparelho. No entanto, a versão que ocupe menor espaço preservando acurácia é preferível.

Algumas técnicas para elevar o nível de desempenho do modelo no dispositivo são discutidas na Seção 5.2.

5 CONCLUSÕES

A evolução das técnicas de manipulação de imagens digitais tem propiciado a geração de resultados cada vez mais convincentes, com destaque para aquelas baseadas em Redes Neurais e Aprendizagem Profunda, que tornam possíveis falsificações do tipo *Deepfake*. A qualidade surpreendente de seus resultados, associada ao livre acesso a essa tecnologia, tem despertado a atenção e o interesse de diversas comunidades.

Entretanto, os propósitos na utilização dessa tecnologia podem variar desde o interesse profissional genuíno para serviços na indústria de entretenimento, como comédia e cinema, até interesses excusos objetivando o ataque à reputação de indivíduos, por meio de vídeos forjados de pornografia de vingança ou vídeos forjados com notícias falsas, envolvendo autoridades ou figuras públicas.

Além disso, a capilaridade da rede mundial de computadores, aliada à profusão e ubiquidade dos dispositivos móveis catalizam o processo de difusão de notícias falsas, trazendo consequências preocupantes em todas as escalas da sociedade, desde o indivíduo até a nação.

Neste trabalho foi proposta uma inovação incremental sobre a arquitetura `Meso-4`, que reúne elementos que a favorecem como uma candidata possível para implantação em produto. A inovação proposta inspirou-se em um trabalho anterior, que adaptava a rede original, acoplando uma camada para extração de características baseadas em um operador diferencial. Conjecturamos que operadores diferenciais usados para extração de contorno, como o Filtro de `Sobel`, também melhorariam as taxas de acerto da arquitetura `Meso-4`. Três configurações diferentes foram propostas e testadas em um Estudo de Ablação, cujos resultados apontam para a superioridade da arquitetura `MesoNetSobelConcat` em relação ao modelo de base, conforme todas as métricas de desempenho utilizadas.

Diante do exposto, a `MesoNetSobelConcat` pode ser considerada apta para a execução da tarefa de detecção de *Deepfake* em dispositivos embarcados e/ou móveis.

5.1 Impactos

Os resultados apresentados na Seção 3.8 e discutidos na Seção 4 oferecem suporte à Conjectura 3.1.1 e acenam para a viabilidade de desenvolvimento de ferramentas para a rápida detecção de *Deepfake* no dispositivo, evitando os inconvenientes da falta de conexão, latência e preocupações com violação de privacidade e segurança de dados. Tais ferramentas contribuiriam para a atenuação da difusão de conteúdo prejudicial, mediante recompartilhamentos, seja pelo desinteresse gerado após a descoberta da falsificação ou mesmo por medidas judiciais que inibissem sua propagação.

5.2 Trabalhos Futuros

Neste trabalho, a aproximação de um operador diferencial, conhecido como Filtro de `Sobel`, foi utilizada como um bloco não-treinável em diferentes configurações nas arquiteturas propostas, buscando enfatizar o conteúdo de altas frequências e favorecer a distinção de superfícies texturizadas reais e forjadas.

5.2.1 Análise Multiescala

A melhoria introduzida pelo operador Filtro de `Sobel` sobre o desempenho do modelo de base sugere que o uso de outros operadores, conhecidos em processamento espacial e análise multiescala, poderiam beneficiar o desempenho da `Meso-4`.

Como perspectiva futura, a investigação de aperfeiçoamento da `Meso-4` pode incluir abordagens exploradas em trabalhos recentes, que reportam bons resultados para tarefas de classificação semelhantes, empregando camadas com parâmetros treináveis que implementam *Wavelets*. Luan *et al.* (2018) propuseram o modelo *GCN - Gabor Convolutional Networks*, que incorporam a *wavelet* de Gabor para tornar o aprendizado de características mais robusto às mudanças de orientação e escala, melhorando o desempenho na tarefa de reconhecimento de objetos, com menos parâmetros treináveis. Alekseev and Bobe (2019) criaram a `GaborNet`, usando a *wavelet* de Gabor como a primeira camada com parâmetros treináveis em uma arquitetura baseada em *CNN*, mostrando que qualquer rede convolucional poderia ser assim facilmente adaptada. Yuan *et al.* (2022) propõem as redes neurais convolucionais adaptativas de Gabor, *AGCN*, em que os filtros convolucionais são adaptativamente modulados pelos filtros de Gabor, construindo assim os filtros convolucionais de Gabor, com parâmetros treináveis.

5.2.2 Outras Bases de Dados

A despeito do presente estudo ter limitado-se à base de dados *Deepfake Dataset*, pelas razões discutidas na Seção 3.4, futuros trabalhos podem incluir outras bases como as constantes na Tabela 4, a fim de melhorar o poder de generalização do `MesonetSobelConcat` para distribuições diferentes de *Deepfake* e até mesmo para outras técnicas de manipulação de imagens de face.

5.2.3 Compressão de Modelo

No que tange à otimização do desempenho do modelo em produto, trabalhos futuros podem (*i*) explorar operadores customizados para aceleradores do tipo *GPU*, assim como (*ii*) técnicas de compressão de modelo, como Quantização e Poda.

5.2.4 Aplicativos

Aplicativos de detecção de *Deepfake* em nível de *frame* para imagens estáticas ou *vídeo*, contendo o modelo aqui apresentado para inferência no dispositivo, podem ser desenvolvidos para sistemas `Android` e `iOS`.

REFERÊNCIAS

AFCHAR, D. *et al.* Mesonet: a compact facial video forgery detection network. **CoRR**, abs/1809.00888, 2018. Available at: http://arxiv.org/abs/1809.00888.

AGARWAL, M. **MesoNet - A Deepfake Detector Built Using Python and Deep Learning**. 2021. https://github.com/MalayAgr/MesoNet-DeepFakeDetection. Capstone work for Bachelor Degree in Computer Science.

ALEKSEEV, A.; BOBE, A. Gabornet: Gabor filters with learnable parameters in deep convolutional neural network. *In*: **2019 International Conference on Engineering and Telecommunication (EnT)**. [*S.l.: s.n.*], 2019. p. 1–4.

BBC-NEWS. **Shamook: Star Wars effects company ILM hires Mandalorian deepfaker**. 2021. https://www.bbc.com/news/entertainment-arts-57996094. Accessed: 2021-10-12 - http://bbc.co.uk – © [2021] BBC.

BIANCO, S. *et al.* Benchmark analysis of representative deep neural network architectures. **IEEE Access**, v. 6, p. 64270–64277, 2018.

BISHOP, C. **Pattern Recognition and Machine Learning**. Springer, 2006. Available at: https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

BROWNLEE, J. **Generative Adversarial Networks with Python: Deep Learning Generative Models for Image Synthesis and Image Translation**. Machine Learning Mastery, 2019. Available at: https://books.google.com.br/books?id=YBimDwAAQBAJ.

CHOLLET, F. Xception: Deep learning with depthwise separable convolutions. *In*: **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*S.l.: s.n.*], 2017. p. 1800–1807.

CHOLLET, F. **Deep Learning with Python, Second Edition**. Manning, 2021. ISBN 9781638350095. Available at: https://books.google.com.br/books?id=mjVKEAAAQBAJ.

CHU, D. *et al.* White paper : DEEP FAKERY — an action plan. *In*: . [*S.l.: s.n.*], 2020. Available at: http://cailinoconnor.com/wp-content/uploads/2020/07/O3-White-Paper-Deepfakery-an-Action-Plan.pdf.

COCCOMINI, D. *et al.* **Combining EfficientNet and Vision Transformers for Video Deepfake Detection**. arXiv, 2022. Available at: https://arxiv.org/abs/2107.02612.

DAS, S. *et al.* **Towards Solving the DeepFake Problem : An Analysis on Improving DeepFake Detection using Dynamic Face Augmentation**. arXiv, 2021. Available at: https://arxiv.org/abs/2102.09603.

DENG, Y. Deep learning on mobile devices - A review. **CoRR**, abs/1904.09274, 2019. Available at: http://arxiv.org/abs/1904.09274.

DIETMAR, J. **GANs And Deepfakes Could Revolutionize The Fashion Industry**. 2019. Available at: https://www.forbes.com/sites/forbestechcouncil/2019/05/21/gans-and-deepfakes-could-revolutionize-the-fashion-industry/?sh=64df49803d17.

DIETTERICH, T. G. Approximate statistical tests for comparing supervised classification learning algorithms. **Neural computation**, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA, v. 10, n. 7, p. 1895–1923, 1998.

DOLHANSKY, B. *et al.* **The Deepfake Detection Challenge (DFDC) Preview Dataset**. 2019.

GEIRHOS, R. *et al.* Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. **CoRR**, abs/1811.12231, 2018. Available at: http://arxiv.org/abs/1811.12231.

GONZALES, R. C.; WINTZ, P. **Digital Image Processing (2nd Ed.)**. USA: Addison-Wesley Longman Publishing Co., Inc., 1987. ISBN 0201110261.

GOODFELLOW, I. J.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge, MA, USA: MIT Press, 2016. http://www.deeplearningbook.org.

JÄHNE, B. **Digital Image Processing. 5th Revised and Extended Edition**. 5th. ed. Springer Berlin Heidelberg, 2002. ISBN 9783540677543. Available at: https://books.google.com.br/books?id=CqrwCAAAQBAJ.

JIANG, L. *et al.* Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection. **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, p. 2886–2895, 2020.

KONG, L. *et al.* A competition of shape and texture bias by multi-view image representation. *In*: **Pattern Recognition and Computer Vision: 4th Chinese Conference, PRCV 2021, Beijing, China, October 29 – November 1, 2021, Proceedings, Part IV**. Berlin, Heidelberg: Springer-Verlag, 2021. p. 140–151. ISBN 978-3-030-88012-5. Available at: https://doi.org/10.1007/978-3-030-88013-2_12.

LI, Y.; CHANG, M.-C.; LYU, S. In ictu oculi: Exposing AI created fake videos by detecting eye blinking. *In*: **2018 IEEE International Workshop on Information Forensics and Security (WIFS)**. [*S.l.: s.n.*], 2018. p. 1–7.

LI, Y. *et al.* Celeb-df: A large-scale challenging dataset for deepfake forensics. *In*: **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [*S.l.: s.n.*], 2020. p. 3204–3213.

LIU, D. *et al.* Bringing AI to edge: From deep learning's perspective. **CoRR**, abs/2011.14808, 2020. Available at: https://arxiv.org/abs/2011.14808.

LOUSKY, S. **Launching DAGsHub integration with MLflow**. 2021. https://dagshub.com/blog/launching-dagshub-integration-with-databricks-mlflow.

LUAN, S. *et al.* Gabor convolutional networks. **IEEE Transactions on Image Processing**, v. 27, n. 9, p. 4357–4366, 2018.

MA, N. *et al.* Shufflenet V2: practical guidelines for efficient CNN architecture design. **CoRR**, abs/1807.11164, 2018. Available at: http://arxiv.org/abs/1807.11164.

MASOOD, M. *et al.* Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward. **CoRR**, abs/2103.00484, 2021. Available at: https://arxiv.org/abs/2103.00484.

MEYES, R. *et al.* Ablation studies in artificial neural networks. **CoRR**, abs/1901.08644, 2019. Available at: http://arxiv.org/abs/1901.08644.

MIRSKY, Y.; LEE, W. The creation and detection of deepfakes: A survey. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 54, n. 1, jan 2021. ISSN 0360-0300. Available at: https://doi.org/10.1145/3425780.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997. ISBN 978-0-07-042807-2.

MOLNAR, C. **Interpretable Machine Learning**: A guide for making black box models explainable. 2. ed. [*S.l.: s.n.*], 2022. Available at: https://christophm.github.io/interpretable-ml-book.

MURPHY, K. P. **Probabilistic Machine Learning: An introduction**. MIT Press, 2022. Available at: probml.ai.

PETROU, M.; PETROU, C. **Image Processing: The Fundamentals**. Wiley, 2010. ISBN 9780470745861. Available at: https://books.google.com.br/books?id=w3BpSIxN9ZYC.

QI, H. *et al.* Deeprhythm: Exposing deepfakes with attentional visual heartbeat rhythms. **CoRR**, abs/2006.07634, 2020. Available at: https://arxiv.org/abs/2006.07634.

ROETTGERS, J. **How AI Tech Is Changing Dubbing, Making Stars Like David Beckham Multilingual**. 2019. Available at: https://variety.com/2019/biz/news/ai-dubbing-david-beckham-multilingual-1203309213/.

ROSSLER, A. *et al.* Faceforensics++: Learning to detect manipulated facial images. *In*: **Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)**. [*S.l.: s.n.*], 2019.

SALZBERG, S. L. On comparing classifiers: Pitfalls to avoid and a recommended approach. **Data mining and knowledge discovery**, Springer, v. 1, n. 3, p. 317–328, 1997.

SZEGEDY, C. *et al.* Going deeper with convolutions. *In*: **2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*S.l.: s.n.*], 2015. p. 1–9.

TOLOSANA, R. *et al.* Deepfakes and beyond: A survey of face manipulation and fake detection. **CoRR**, abs/2001.00179, 2020. Available at: http://arxiv.org/abs/2001.00179.

TUNG, K. **TensorFlow 2 Pocket Reference: Building and Deploying Machine Learning Models**. O'Reilly Media, Incorporated, 2021. ISBN 9781492089186. Available at: https://books.google.com.br/books?id=7xEizgEACAAJ.

WALSH, J. *et al.* Deep learning vs. traditional computer vision. *In*: . [*S.l.: s.n.*], 2019. ISBN 978-981-13-6209-5.

WANG, J. *et al.* Deep learning towards mobile applications. **CoRR**, abs/1809.03559, 2018. Available at: http://arxiv.org/abs/1809.03559.

WEERAWARDANA, M.; FERNANDO, T. Deepfakes detection methods: A literature survey. *In*: **2021 10th International Conference on Information and Automation for Sustainability (ICIAfS)**. [*S.l.: s.n.*], 2021. p. 76–81.

Xia, Z. *et al.* Deepfake Video Detection Based on MesoNet with Preprocessing Module. **Symmetry**, v. 14, n. 5, p. 939, maio 2022.

YE, Z. *et al.* Dufenet: Improve the accuracy and increase shape bias of neural network models. **Signal, Image and Video Processing**, v. 16, p. 1153–1160, 2022.

YUAN, Y. *et al.* Adaptive gabor convolutional networks. **Pattern Recognition**, v. 124, p. 108495, 2022. ISSN 0031-3203. Available at: https://www.sciencedirect.com/science/article/pii/S0031320321006713.

ZAHARIA, M. A. *et al.* Accelerating the machine learning lifecycle with mlflow. **IEEE Data Eng. Bull.**, v. 41, p. 39–45, 2018.

ZHANG, Y. *et al.* A survey on face forgery detection of Deepfake. *In*: JIANG, X.; FUJITA, H. (ed.). **Thirteenth International Conference on Digital Image Processing (ICDIP 2021)**. SPIE, 2021. v. 11878, p. 153 – 159. Available at: https://doi.org/10.1117/12.2600889.

ZHENG, L.; ZHANG, Y.; THING, V. L. A survey on image tampering and its detection in real-world photos. **Journal of Visual Communication and Image Representation**, v. 58, p. 380–399, 2019. ISSN 1047-3203.

ZI, B. *et al.* WildDeepfake: A challenging real-world dataset for deepfake detection. *In*: _____. **Proceedings of the 28th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2020. p. 2382–2390. ISBN 9781450379885. Available at: https://doi.org/10.1145/3394171.3413769.