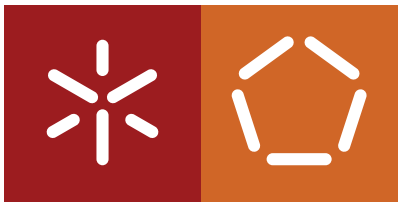
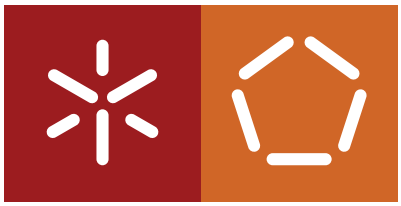


Universidade do Minho
Escola de Engenharia
Departamento de Informática

Leonardo de Jesus Silva

Detetor de conteúdo multimédia falso gerado através de algoritmos Deep Fake

September 2022



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Leonardo de Jesus Silva

Detetor de conteúdo multimédia falso gerado através de algoritmos Deep Fake

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho efetuado sob a orientação do(a)
Paulo Jorge Freitas de Oliveira Novais

September 2022

Direitos de Autor e Condições de Utilização do Trabalho por Terceiros

Agradecimentos

Os meus agradecimentos ao professor, Paulo Jorge Novais, por aceitar e proporcionar essa rica oportunidade de trabalhar num tópico do meu interesse. De igual forma, agradeço pelo suporte e apoio durante a execução do mesmo.

A minha família em especial a minha mãe, Silvania Costa de Jesus, mesmo sendo uma pessoa iletrada sempre esteve a incentivar os seus filhos a obtiverem o maior grau de instrução académica possível. Como também ensinando os valores do trabalho, honestidade e ser fraterno com os demais independente da raça, cor ou credo.

Não possuo palavras para expressar meus agradecimentos ao professor, Filipe Gonçalves. Ele foi uma pedra basilar durante minha formação profissional no decorrer deste trabalho, orientou-me tecnicamente. Se fez solidário nos momentos mais complicados, dos quais enfrentei na minha vida pessoal, durante o desenvolvimento dessa dissertação. Desta forma, foi fundamental sua orientação para o aprimoramento do meu pensamento crítico e analítico. O professor Filipe Gonçalves é uma referência para mim e a quem sou extremamente grato.

Ao Filipe Monteiro por auxiliar no processo de adaptação em Portugal. Por sempre oferecer uma escuta ativa aos meus problemas tantos profissionais e pessoais, consequentemente oferecendo-me uma nova perspetiva. Também agradeço a sua família, os Monteiros, pois esta me "adotou"cá em Portugal proporcionando que não estivesse sozinho nas festividades.

A Udeli Ribeiro e ao seu companheiro que ajudou-me desde arrendar um quarto, em seu lar, e auxilio para entender como Portugal funcionava, sendo de grande utilidade na minha jornada cá.

A equipa de investigação do projeto no INESCTEC agradeço a colaboração na minha formação, meu agradecimento faço na figura dos professores Victor Fonte, João Marco Silva e o António Luís Sousa e aos demais colegas mestrandos com quem tive a honra de trabalhar durante o projeto de investigação.

A Filipa Parente, a qual tive o privilégio de ser parceiro em N projetos. Agradeço a amizade e apoio, pois assim como o Filipe Monteiro foram indivíduos singulares na minha adaptação em Portugal.

Aos meus amigos que ficaram no Brasil, no entanto, seus suportes e amizades atravessaram o oceano atlântico ao meu encontro. Dentre estes faço meu agradecimento em especial a Hayanna Oliveria, Francyelle Mascarenhas e Andrey.

Leonardo Silva

Statement of Integrity

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

Abstract

A society generates a volume of data unprecedented in history. Despite this, misinformation is growing, causing concerns that are not the best of journalism and democracy. This problem becomes even more evident with technological advances, such as the ability to manipulate or semantic meaning of images, videos, or audio. The results of these manipulations, imply difficulty in distinguishing between fake and original content. In this way, manipulated multimedia content is called *deepfake*. This nomenclature is the result of the combination of two terms *Deep Learning (DL)* and *fake*.

To solve this problem, non state-of-the-art works present different algorithms for classifying *deepfake* videos, two of which use neural architectures based on *Convolutional Neural Network (CNN)*, *Long Short-Term Memory (LSTM)* or *Transformers* networks. Some studies show positive results for classifying *deepfake* data generated from a given technique for generating *deepfake*. However, these mostly show no positive results when the models classified data generated from another *deepfake* technique other than two data used rather than three models. Consequently, work is lacking for a more general classification regardless of the method used to create the manipulated content.

Thus, experiments developed used neural networks in different strategies. Consequently, propose a solution to a gap found in the state of the art, the ability of a model to classify a fake video independent of the creation technique. After the experiments, the algorithm chosen was multi-classification used in the detector. For greater accuracy when analyzing the videos, the classifier has confidence levels (Not Confident, Unreliable, Very Confident) for each classification performed. Thus, the detector validated with videos from the four text techniques and original videos. It was possible to know when the classifier labeled a video as deepfake with a "Very Confident"confidence level. This classification has an average 91% chance of being correct, the unmanipulated videos it has a 60% chance.

Keywords: multimedia content, deepfake, synthetic content, Deep Learning, detector

Resumo

A sociedade encontra-se a gerar um volume de dados sem precedentes na história. Apesar disso, a desinformação tem vindo a crescer, provocando preocupações no meio jornalístico e democrático. Este problema se tornou ainda mais evidente com os avanços tecnológicos, como a capacidade de manipular o significado semântico de imagens, vídeos ou áudios. Os resultados destas manipulações impõem uma dificuldade em distinguir entre conteúdo falso e original. Desta forma, os conteúdos multimédia manipulados são chamados de *deepfake*. Esta nomenclatura é resultado da combinação dos termos *Deep Learning (DL)* e *fake*.

Para solucionar tal problemática trabalhos no estado da arte apresentam diferentes algoritmos para classificação de vídeos *deepfake*, dos quais utilizam-se de arquiteturas neuronais baseadas em *Convolutional Neural Network (CNN)*, *Long Short-Term Memory (LSTM)* ou redes *Transformers*. Há trabalhos dos quais apresentam resultados positivos ao classificar dados *deepfake* criados a partir de uma determinada técnica para geração *deepfake*. No entanto, estes em sua maioria não apresentam resultados quando seus modelos são confrontados com dados gerados a partir de outra técnica *deepfake* distinta dos dados utilizados no treino do modelo. Consequentemente, há um lacuna nos trabalhos para uma classificação mais generalista independente do método utilizado para criação do conteúdo manipulado.

Desta forma, as experiências desenvolvidas nesse trabalho utilizaram redes neuronais em diferentes estratégias. Resultando na proposta de uma solução para tal lacuna encontrada no estado da arte, a capacidade de um modelo ao classificar um vídeo falso independente da técnica de criação. Após as experiências escolheu-se o algoritmo de multi-classificação para a utilização no detetor. Para uma maior precisão na análise dos vídeos, o detetor possui níveis de confiança (Não Confiável, Nada Confiável, Muito Confiável) para cada classificação realizada. Ao validar o detetor, com vídeos das quatro técnicas *deepfake* e vídeos originais, foi possível saber que o nível de confiança "Muito Confiável"a precisão média é de 91% quando a *label* era *deepfake*, independente da técnica *(DeepFakes, Face2Face, FaceSwap e NeuralTextures)* utilizada para criá-lo, enquanto que ao rotular dados como reais com o mesmo nível de confiança obtém um precisão de 60% .

Palavras-chave conteúdo multimédia, *deepfake*, conteúdo sintético, Deep Learning, detetor.

Conteúdo

Lista de Acrónimos

ACM Association for Computing Machinery. 27

ANN Artificial Neural Networks. 5, 6

BTT Backpropagation Through Time. 13

CAN Convolutional Attention Network. 28

CE Critérios de Exclusão. 27, 28

CI Critérios de Inclusão. 27, 28

CNN Convolutional Neural Network. iv, v, 10, 13, 28, 30, 34, 35, 39, 41, 50, 60, 73

CSV Comma-Separated Values. 64

DL Deep Learning. iv, v, 1, 2, 5, 6, 9, 23, 24, 42–45, 73, 74

FF++ FaceForensics++. xiii, 30, 39, 41, 43–47, 51

FN Falso Negativo. 48

FP Falso Positivo. 48

FPS Frames Por Segundos. xiii, 5, 39, 46, 47, 51, 63, 64

GAN Generative Adversarial Networks. 1, 2, 16, 17, 43

GPU Graphics Processing Unit. 19, 42

IA Inteligência Artificial. 5

IEEE Institute of Electrical and Electronics Engineers. 27

LSTM Long Short-Term Memory. iv, v, 14, 15, 17, 31, 35

ML Machine Learning. 5, 13, 42

MLP Multi-Layer Perceptron. 9

NLP Natural Language Processing. 20

NMT Neural Machine Translator. 17, 20, 21

RNN Recurrent Neural Network. 12–14, 17–19, 31

RSL Revisão Sistemática da Literatura. 25, 26, 40

VAE Variational Autoencoder. 16

VN Verdadeiro Negativo. 48, 56

VP Verdadeiro Positivo. 48, 56

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A sociedade gera um imenso volume de dados sem precedentes na história, um dos principais motivos é a utilização da Internet, em especial uso das redes sociais [1]. Contudo, há um aumento significativo no nível de desinformação propagada nestes meios. A origem desta desinformação é derivada do compartilhar de informações falsas tais como: teorias de conspiração, fraudes eleitorais, duvida sobre a segurança da vacinação contra a COVID-19 entre outras [2]. Por estes motivos, há uma preocupação na comunidade jornalística e democrática com os avanços tecnológicos referente a manipulação de conteúdos multimédia: vídeos, imagens e áudios. Pois a manipulação destes estão sendo feitas a partir de técnicas de computação gráfica e algoritmos de Deep Learning (DL), sendo conhecida como *deepfake* [3]. Este tipo de conteúdo ao ser compartilhado na Internet, aumenta o grau de desinformação nos meios de comunicação digital, tornando-se um dos tópicos mais críticos para a sociedade dentro do ambiente digital. Pois a capacidade de manipular o significado semântico de uma conteúdo multimédia pode causar consequências no mundo físico de dimensões monetários e reputações em organizações, seja estas pública ou privada, como também pode afetar diretamente uma pessoa física.

O *deepfake* tem como base modelos generativos. Em 2014, estes modelos popularizaram-se quando o até então estudante de *PhD* da Universidade de Montreal, Ian Goodfellow, propôs uma arquitetura neuronal intitulada Generative Adversarial Networks (GAN) [4], sendo um modelo de rede neuronal em DL cujo propósito é a geração de conteúdo sintético. Posteriormente, este método tornou-se numa das principais técnicas para geração de conteúdo *deepfake*. O fenómeno do *deepfake* veio a popularizar na plataforma de média social, *"Reddit"*. Quando um utilizador anónimo compartilhou um vídeo pornográfico manipulado, onde o rosto de uma atriz do segmento pornográfico foi substituído por o rosto de uma determinada celebridade de *Hollywood* fazendo com que as pessoas acreditassem que se tratava de um vídeo pornográfico da celebridade [5]. Após tal episódio o utilizador foi banido do *Reddit*, mas suas ações desencadearam um enorme interesse sobre o tema e novos conteúdos começaram a propagar-se em outras plataformas de média social como "*Twitter*" e "4chan" [6].

As consequências diretas deste tipo de conteúdo podem ser catastróficas [7], a exemplo dos casos de vídeos pornográficos falsos criados com celebridades ou criação de conteúdo pornográfico como forma de vingança. Outra utilização perigosa é a criação de conteúdo *deepfake* tendo como alvo indivíduos em

cargos importantes, por exemplo o ex-presidente dos EUA, Barack Obama, a transmitir uma mensagem falsa. Um exemplo clássico do estado da arte sobre *deepfake* é o famoso vídeo criado pelo comediante Jordan Peele, que retratou o ex-presidente dos Estados Unidos da América (EUA), Barack Obama, a fazer um discurso sobre ameaça do *deepfake* para sociedade.

O conteúdo multimédia *deepfake* representa uma progressiva ameaça para as empresas, como também a política e o sistema judicial em todo o mundo [8]. O uso malicioso do *deepfake* no mundo corporativo é representado pelo caso no qual golpistas defraudaram uma empresa com sede no Reino Unido, fazendo-se passar pelo Chefe Executivo. Os autores deste golpe convenceram os funcionários do departamento financeiro a transferir um montante de $ 220.000 para uma conta controlada pelos próprios golpistas [9], para isto utilizaram-se de conteúdo multimédia *deepfake*, no formato de áudio. A *Symantec*, empresa especializada em segurança cibernética, revelou que o uso de *deepfake* e engenharia social foram usadas para defraudarem três diretores financeiros de fundos substanciais não divulgados [8]. Além desses casos, a *Forrester Research* [8] previu uma perda monetária de $ 250 milhões no final de 2020 decorrente de fraudes, pois com os avanços contínuos da geração de conteúdo falso as empresas, sistemas políticos e judicias provavelmente continuarão a sofrer perdas, a múltiplos níveis seja de forma financeira, legal e ou de credibilidade.

1.1 Deepfake

Para uma maior compreensão da definição de *deepfake*, precisa-se entender o que são *fake news*, ou seja, noticias falsas. Como o próprio termo apresenta, são informações falsas que se espalham pelos meios de comunicação a se passar como factos. Nos últimos anos, as *fake news* trouxeram uma discussão nos ambientes virtuais, no contexto jornalístico e da própria sociedade democrática. O estudo [10] analisou que as noticias falsas na rede social *Facebook* durante as eleições dos Estados Unidos da América em 2016 recebiam uma maior atenção generalizada quando comparada com as noticias verdadeiras. O trabalho [11] validou a hipótese que uma noticia falsa possui uma maior probabilidade a se tornar uma informação viral em relação a notícias reais numa rede social tal como *Facebook* ou *Twitter*, pois há um maior engajamento para compartilhar quando a notícia é falsa.

No entanto, as noticias falsas ou designadas *fake news* são produzidas textualmente nas redes sociais, utilizando softwares capazes de disseminar rapidamente estas notícias. Posteriormente, surgiu a disseminação de conteúdo de multimédia manipulado. Este tipo de conteúdo vai além da manipulação textual, pois este tem como propósito a alteração do significado semântico de conteúdos multimédia (imagem, vídeo e áudio). Os trabalhos [12],[13] [14] apresentam uma definição de *deepfake* sendo um produto da aplicação de algoritmos do sub-campo da Inteligência Artificial, nomeadamente o *DL*. Em destaque para os algoritmos generativos a exemplo das redes *GANs*. Os trabalhos descreveram o crescente impacto prejudicial na fiabilidade de conteúdos multimédia digitais. Apesar disso, o uso destas tecnologias apresentam espectros positivos também. A exemplo, o uso do *deepfake* tem sido utilizado para clonagem de voz a partir de gravações de áudio de pessoas que por alguma razão já não as possuem. Vídeos *deepfake* podem animar galerias e museus. Para a indústria do entretenimento, a tecnologia do

deepfake pode ser usada para melhorar a dobragem de filmes na língua estrangeira e, de forma mais polémica, a "ressuscitar"atores para as cenas. A exemplo, o falecido James Dean atuou no filme *Finding Jack* em 2020, um filme sobre a guerra do Vietnã.

No entanto, os conteúdos *deepfake* e suas respetivas ferramentas de criação tem demonstrado seu impacto negativo a provocar preocupação, por prejuízos morais a imagem de diversos cidadãos, em especial o publico feminino, e.g., a criação de vídeos *deepfake* pornográfico. Empresas do segmento de investigações do impacto de tecnologias, a exemplo *Sensity AI* estima que entre 90% a 95% de todos os vídeos *deepfake* presentes na *Internet* são vídeos pornográficos não consensuais, e cerca de 90% destes tem como alvo mulheres [15].

Para além do impacto individual, há um risco para as instituições e governos. Durante a guerra da Ucrânia e Rússia, tem ocorrido reuniões via vídeo-chamada com os autarcas das principais capitais da Europa para discutir questões humanísticas e de guerra numa destas reuniões houve a participação de um intruso, o qual utilizou do *deepfake* para se passar por Vitali Klitschko, autarca de Kiev [16]. Tempos depois de iniciada a reunião foi detetado que o participante em questão não era o autarca de Kiev e sim um *deepfake*. Desta forma, percebe-se o nível de ameaça que a utilização do *deepfake* pode causar.

1.2 Multimédia

Segundo o dicionário de *Cambrige* [1] a palavra multimédia é definida da seguinte forma: combinação de imagens em movimento ou estáticas, som, música e palavras, em computadores ou para o entretenimento. Desta combinação deriva diversos elementos multimédia como: *Graphics Interchange Format*(GIFs), vídeos, musicas entre outros. Há diversos formatos de conteúdo multimédia para o escopo desse projeto. No entanto, o objeto de estudo dessa dissertação é o conteúdo multimédia no formato de vídeo. Para maior compreensão do que seria um vídeo, é necessário descrever outro elemento multimédia, a imagem. Um vídeo é uma combinação de várias imagens acrescido de áudio durante a transição das imagens no decorrer temporal do vídeo.

Uma imagem como é visualizada por humanos é caracterizada como uma grade bidimensional, onde cada célula da grade é formalmente chamada de elemento da imagem, também designado de *pixel*. Para um computador, a imagem em questão é uma matriz bidimensional de valores numéricos com cada célula da matriz a armazenar valores de seus respetivos *pixeis*.

O *pixel* é considerado a "cor"ou a"intensidade"da luz que aparece numa determinada posição na imagem. E.g. , numa imagem a preto e branco, cada pixel apresenta um valor entre 0 e 255, onde 0 corresponde ao "preto"e 255 ao "branco". Os valores entre 0 e 255 são tons variados de cinza, onde os valores mais próximos a 0 corresponde a tonalidades adjacentes ao "preto"e os valores mais próximos a 255 são tonalidades próximas ao "branco", a exemplo, tal como apresentado na Figura 1 aonde ao lado esquerdo está sua representação visual, enquanto que ao lado direito é apresentado a sua respetiva matriz da qual o computador interpreta a informação daquela imagem.

[1] Dicionário de *Cambridge* https://dictionary.cambridge.org/dictionary/english/multimedia

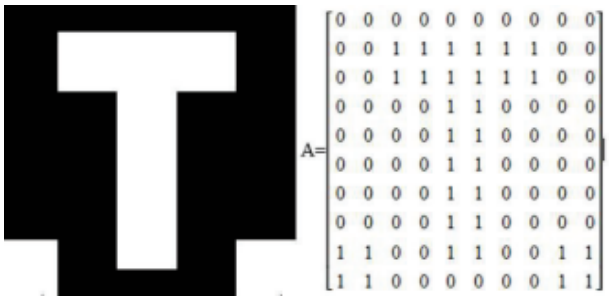


Figura 1: Matriz representativa da imagem [17].

Desta forma, imagens a preto e branco são compostas de apenas uma matriz de duas dimensões. No entanto, as imagens atualmente são maioritariamente coloridas das quais apresentam três matrizes de duas dimensões, cada uma representa uma das cores do formato *RGB(Red,Green,Blue)*. Cada *pixel* é formado de uma tupla de 3 inteiros de 8 *bits*, sendo que a tupla (0,0,0) representa o preto e a (255,255,255) o branco. Tal como demonstrado na Figura 2, uma imagem colorida no formato *RGB*, extraída do meio de comunicação *euronews*[2].

Figura 2: Imagem extraída de uma matéria do *euronews*

A Figura 2 é composta por três matrizes, das quais a literatura intitula de canais. Cada canal apresenta a imagem nos distintos espectros das cores do *RGB*. Desta forma, pode-se ter a imagem colorida sendo composta por 3 tal como apresentado na Figura 3.

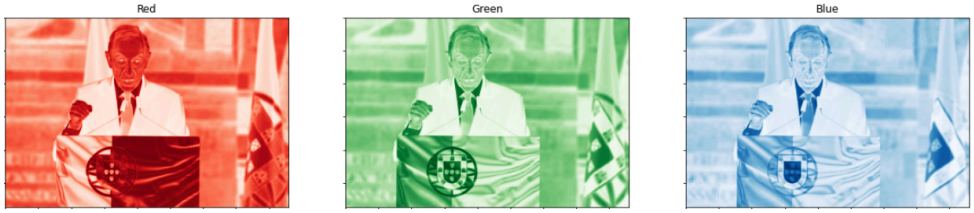


Figura 3: Imagem *RGB* modificada.

[2]Link do artigo do qual se extraiu a imagem: https://www.euronews.com/2021/02/18/portuguese-president-orders-constitutional-review-of-euthanasia-law

Consequentemente, um vídeo é composto por um conjunto de imagens. Estas são designadas na literatura de *frames*. A sequência de imagens é responsável no vídeo pela sensação de movimento. Para o número de imagens em relação ao tempo em segundos, existe uma métrica chamada Frames Por Segundos (FPS). Afirmar que um vídeo apresenta 30 FPS, este se trata de um vídeo que possui uma transição de 30 imagens por segundo. Desta forma, se um vídeo tem duração de 16.67 segundos com uma taxa de 30 FPS, isto significa que para a construção deste vídeo foram necessários aproximadamente 500 imagens.

1.3 Machine Learning

Machine Learning (ML) é um sub-campo da Inteligência Artificial (IA), sendo esta um ramo da Ciência da Computação, que estuda o desenvolvimento de algoritmos capazes de aprenderem a executar determinadas tarefas sem a necessidade de interferência humana. Este sub-campo do conhecimento tem se destacado, pela sua diversidade aplicacional, em investigações cientificas e em ambientes empresariais, desta forma, resultando em programas inteligentes para automação de rotinas como compreender a fala, reconhecer objetos e pessoas ou classifica-los, auxiliar em diagnósticos clínicos e suportar as pesquisas cientificas básicas [18].

A IA pode ser definida como a capacidade de proporcionar ao computador a habilidade de realizar determinadas tarefas que os humanas até então as executam, tais como a tomada de decisões ou aprendizagem. O *ML* é uma das abordagens para concretização da IA, tal como demonstrado na Figura 4, pois os algoritmos de *ML*, utilizam-se de aprendizagem estatística para construção de uma inteligência. Para Russell [20] as aplicações são consideradas "inteligentes" se estas possuem a capacidade de realizar uma determinada tarefa, na qual possam otimizar sua execução, após realizadas algumas observações através dos dados.

Os algoritmos de *ML* estão agrupados maioritariamente em dois tipos de aprendizagem: aprendizagem não supervisionada e supervisionada. A aprendizagem supervisionada ocorre quando os algoritmos se utilizam de dados rotulados para realizar uma determinada tarefa, seja de classificação ou previsão. Os algoritmos de aprendizagem não supervisionada trabalham com dados não rotulados para tentar identificar semelhança entre os dados e encontrar padrões ocultos no conjunto de dados.

Essa dissertação esta focado em *DL*, que corresponde a um subconjunto de algoritmos de *ML* inspirados na estrutura e função do cérebro, os quais aprende características através de um processo de aprendizagem hierárquica. Esses algoritmos são normalmente chamados *Artificial Neural Networks (ANN)*. O *DL* baseia-se na representação da aprendizagem na qual o algoritmo aprende com as melhores características para realizar uma determinada tarefa por si só, navegando pelos dados fornecidos. *DL* é um dos campos em ascensão na ciência dos dados, com resultados positivos na robótica, reconhecimento de imagem, entre outros.

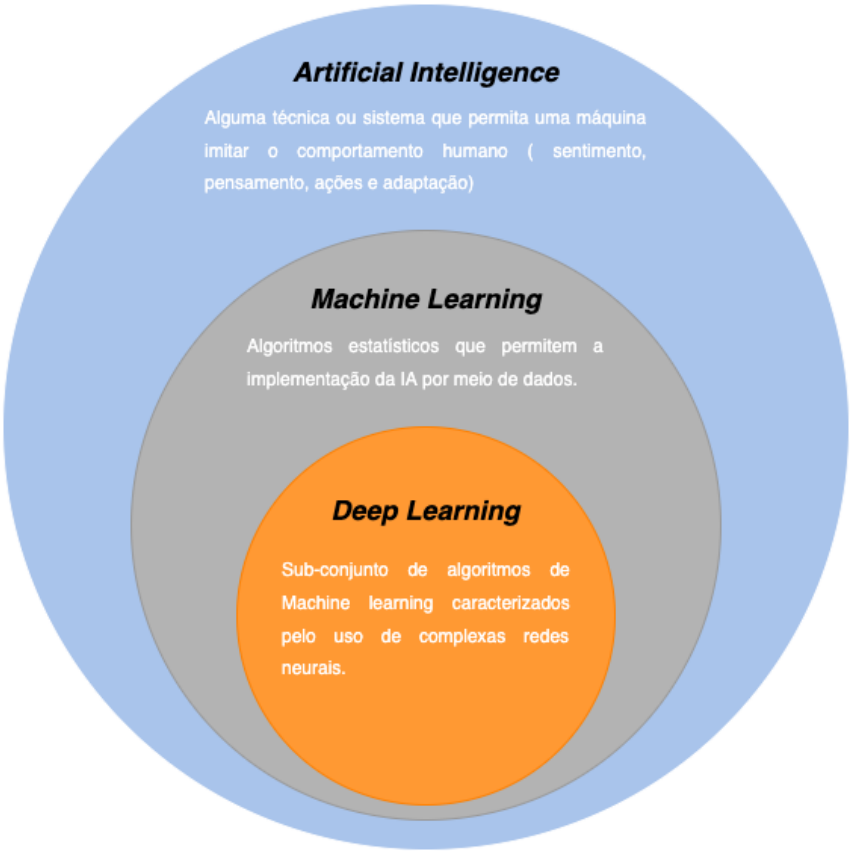


Figura 4: *Artificial Intelligence versus Machine Learning versus Deep Learning*

O *DL* tem sido amplamente revisto nos últimos anos, sendo um campo de investigação a desenvolver modelos para tarefas computacionais complexas, como modelação preditiva, classificação, entre outras. A base do *DL* é inspirada no sistema neuronal biológico. Consequentemente, proporcionando as *ANNs* a alta capacidade do cérebro humano em realizar cálculos complexos [21]. As *ANNs* proporcionam aprendizagem de máquinas e são constituídas por neurónios, estes com seus respetivos pesos e funções de ativação.

A capacidade preditiva das redes neuronais provém da estrutura hierárquica ou multi-camadas das redes. A estrutura de dados possibilita aprender ou representar características em diferentes escalas ou resoluções e combiná-las para sintetizar em informação. O bloco de construção das redes neuronais é constituído por neurónios artificiais. Muito semelhante aos neurónios biológicos, que têm dendritos e axónios, um único neurónio artificial é uma estrutura de uma árvore, que tem nós de entrada e um único nó de saída, este está ligado a cada nó de entrada [21]. A Figura 5 apresenta uma comparação visual dos dois neurónios, o artificial e o biológico.

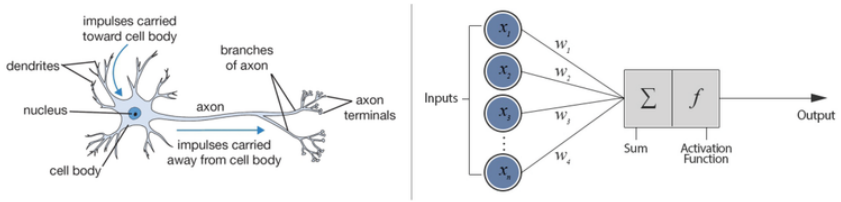


Figura 5: Neurónio Biológico e Neurónio Artificial (adaptada de [21]).

O neurónio artificial, é uma unidade computacional, a qual recebe um ou mais sinais de entrada resultando num único sinal de saída, que pode ser distribuído como sinal de saída da rede ou como sinal de entrada para um ou vários outros neurónios da camada posterior, formando-se assim uma rede neuronal. Os dendritos e axonios são representados matematicamente pelas sinapses, e a intensidade da ligação é determinada pela grandeza do peso sináptico, simbolizada pelo w. Quando as entradas, x são apresentadas ao neurónio, elas são multiplicadas pelos pesos sinápticos (w) correspondentes, gerando as entradas ponderadas, ou seja, x1 que multiplica w1, etc. Isto descreve uma das bases matemáticas do funcionamento de uma rede neuronal artificial, a multiplicação de matrizes, Xw = y:

$$\begin{bmatrix} x_{1,1} & \cdots & x_{1,d} \\ x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \end{bmatrix} \times \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_d \end{bmatrix}$$

O neurónio totaliza todos os produtos num único resultado. Este valor é apresentado a uma função de ativação, que tem, dentre outras, a finalidade de evitar o acréscimo progressivo dos valores de saída ao longo das camadas da rede, visto que tais funções possuem valores máximos e mínimos contidos em intervalos determinados. Um neurónio quando soma os sinais recebidos ativa ou não os neurónios posteriores de acordo com o *threshold* estabelecido pela função. Esta ativação do neurónio é obtida através da aplicação de uma "função de ativação", que ativa ou não os neurónios, dependendo do valor da soma ponderada das suas entradas. Há diferentes tipos de funções de ativação. No entanto, na tabela 1 está descrito algumas das funções de ativação mais conhecidas, são elas : Unidades Lineares Softmax, Sigmoid, e Rectificador (ReLU), *Leaky ReLU, Tahn*.

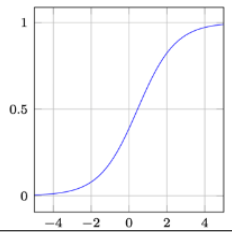
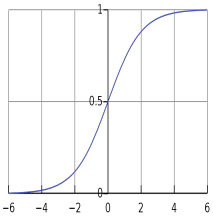
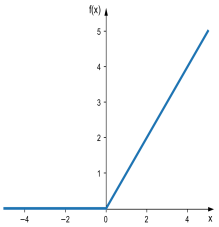
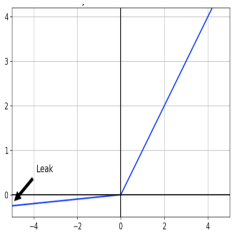
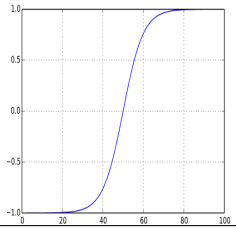
Função de Ativação	Equação	Gráfico representativo	Descritivo da aplicação:
Softmax	$f(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$		Utilizada em tarefas de multi-classificação,normalizando a saída das camadas anteriores para levar o total de uma, especificando o probabilidade para cada classe.
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$		Normalmente utilizada para classificação binária, retornando entre 0 e 1.
ReLU	$f(x) = max(0, x)$		Recentemente, tem sido utilizada como uma unidade oculta. ReLU Funciona bem com grandes e consistentes gradientes
Leaky ReLU	$f(x) = max(0.1x, x)$		É uma versão melhorada da função ReLU, pois esta o gradiente é 0 para x<0, o que desativaria os neurónios daquela região. Leaky ReLU é definido para resolver este problema. Em vez de definir a função Relu como 0 para valores negativos de x, pois possui uma pequena inclinação positiva na área negativa.
Tahn	$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$		A função tanh é muito semelhante à função sigmoid. A única diferença é que ele é simétrico em torno da origem. A faixa de valores neste caso é de -1 a 1. Assim, as entradas para as próximas camadas nem sempre serão do mesmo sinal

Tabela 1: Funções de Ativação.

Além das funções de ativação, há uma função que tem uma elevada relevância no treino das redes neuronais,a *Loss Function*. Esta avalia o quão bem o seu algoritmo modela o seu conjunto de dados. Se as suas previsões estiverem totalmente erradas, a *Loss Function* produzirá um número mais elevado. Se forem boas, produzirá um número mais baixo. Esta função auxilia no processo de *Backpropagation*, onde se ajusta os pesos da rede do fim para o início, de forma a minimizar o resultado da *Loss Function*. Um pormenor importante é que os pesos são habitualmente inicializados aleatoriamente para aumentar a capacidade de otimização dos pesos. Desta forma, busca-se o mínimo da *Loss Function* para otimizar os pesos para da rede durante o processo de *Backpropagation*.

Para além da *Loss Function* existem também algoritmos de otimização responsável por avaliar os erros de predição e corrigir os pesos da rede. Entre estes destaca-se os mais utilizados, são eles: os estocásticos *Gradient Descent (SGD)*, *Adaptive Moment Estimation (ADAM)*, e *RMSprop*.

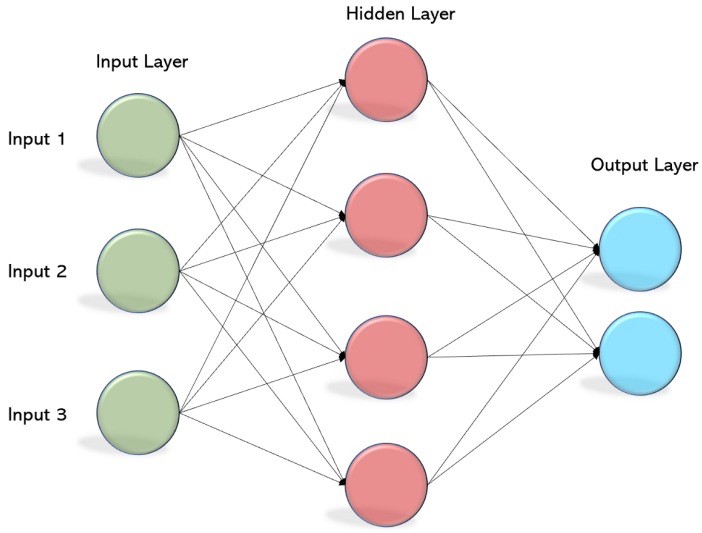


Figura 6: Modelo de uma Rede Neuronal Artificial Simples [22].

A Figura 6 apresenta um modelo simplificado de uma rede neuronal, da esquerda para direita, a primeira camada (*Input Layer*) recebe a o conjunto de dados. Esta camada é chamada de camada visível, porque é a parte exposta da rede. A ultima camada da rede (*Output Layer*) possui dois neurónios na camada, pois produz diretamente os valores que se deseja prever ou classificar. As camadas entre *Input Layer* e *Output Layer*, são chamadas de camadas ocultas (*Hidden Layer*). Mediante o aumento do poder computacional e bibliotecas eficientes, possibilitaram a construção de redes neuronais com um maior número de camadas ocultas e neurónios. Desta forma, o DL está diretamente correlacionado com a rede neuronal possuir muitas camadas ocultas.

1.3.1 Multi-Layer Perceptron (MLP)

A *Multi-Layer Perceptron (MLP)* é um tipo de rede neuronal artificial e proporciona muita flexibilidade e provou ser útil e fiável ao longo de décadas numa vasta gama de problemas [23]. As *MLPs* são adequadas para problemas de previsão e classificação, onde os *inputs* são atribuídos a uma classe [23]. As *MLPs* são o tipo clássico de rede neuronal e são utilizados para regressão, classificação binária e multi-classificação.

Há uma grande quantidade de tipos de arquiteturas de redes neuronais. Embora o design das camadas de entrada e saída de uma rede neuronal seja frequentemente direto, pode haver bastante variação para o design das camadas ocultas. As redes *MLPs* são redes neuronais, onde a saída de uma camada é usada como entrada para a próxima camada. Estas redes são chamadas de redes neuronais *feed forward*. Isto significa que não há *loops* na rede – as informações são sempre processadas para a frente, nunca

são enviadas de volta para camadas anteriores. No entanto, existem outros modelos de redes neuronais em que os *loops* são possíveis.

1.3.2 Convolutional Neural Network (CNN)

Em 1998, LeCun et al. apresentaram um reconhecedor, com resultados positivos, para dígitos manuscritos chamado *LeNet* [24], do qual utilizava *backpropagation* numa rede *feed forward* com muitas camadas ocultas. Esta arquitetura neuronal foi formalizada sob o nome de redes neuronais convolucionais. As Redes Neuronais Convolucionais (ConvNets ou CNNs) são redes neuronais utilizadas para classificar imagens, agrupá-las por similaridade e realizar reconhecimento de objetos dentro de uma cena. As *CNNs* também são aplicadas a problemas que envolve dados no formato de áudio e análise de texto além de dados visuais. No entanto, a eficácia das arquiteturas neuronais convolucionais no reconhecimento de dados gráficos (imagens) proporcionou à mesma o *status* de arquitetura padrão quando o tema é reconhecimento de imagens. Este tipo de rede impulsionou grandes avanços na Visão Computacional, que tem aplicações em carros autónomos, robótica, drones, segurança, diagnósticos médicos, entre outros.

As redes neuronais convolucionais distinguem-se de outras redes neuronais pelo seu desempenho superior em tarefas de classificação de imagem. As *CNNs* possuem três tipos principais de camadas: camadas convolucional *(Convolutional layer)*, camada de agrupamento *(Pooling layer)* e a camada totalmente ligada *(Fully-connected layer)* sendo precedida por uma camada de achatamento (Flatten layer), como demonstrado na Figura 7.

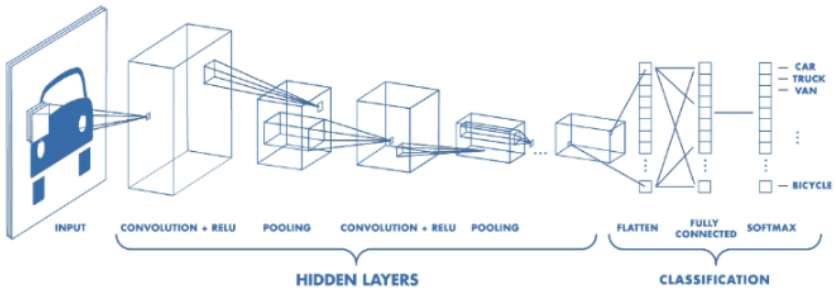


Figura 7: Rede Neuronal Convolucional (Google Imagens).

As camadas convolucionais são o núcleo da construção das redes neuronais *CNN*. Requer alguns componentes como dados de entrada, um filtro e um mapa de caraterísticas. Sendo uma imagem a cores a entrada, esta é composta por uma matriz de *pixels* em 3D. Isto significa que a entrada terá três dimensões - altura, largura e profundidade - esta última correspondendo ao *RGB* de uma imagem. Temos também um detetor de características, também conhecido como *kernel* ou filtro, que se moverá através da imagem, verificando se uma determinada característica está presente. Este processo é conhecido como convolução.

O detetor de características é um conjunto bidimensional (2-D) de pesos, que representa parte da imagem. O filtro é então aplicado a uma área da imagem, e é calculado entre os *pixels* de entrada e o

filtro. Este produto é então introduzido numa matriz de saída. Depois, o filtro desloca-se para outra área, repetindo o processo até que o filtro tenha analisado toda a imagem. A saída final da série de produtos da entrada e do filtro é conhecida como um mapa de características, tal como descrito na Figura 8.

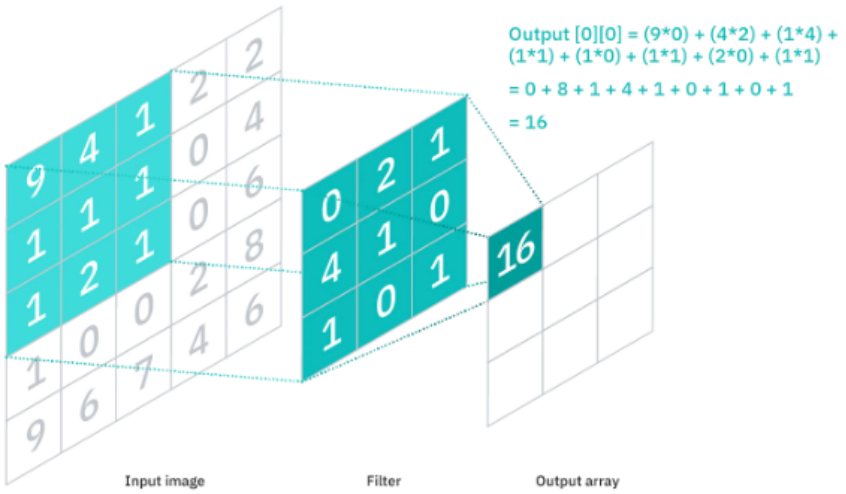


Figura 8: Processo da camada de convolução (Google Imagens).

Os pesos no detetor de características permanecem fixos à medida que esse se move pela imagem, o que também é conhecido como partilha de parâmetros. Alguns parâmetros, como os valores de peso, ajustam-se durante o treino através do processo de *backpropagation* e descida de gradiente. Contudo, existem três hiper-parâmetros que afetam o tamanho do volume de saída dos quais precisam ser definidos antes do início do treino da rede neuronal. O número de filtros afetam a profundidade da saída. Por exemplo, três filtros distintos produziriam três mapas de características diferentes, criando uma profundidade de três; *Stride* é a distância, ou número de *pixels*, que o núcleo se movimentará sobre a matriz de entrada (imagem); O preenchimento zero é normalmente utilizado quando os filtros não se ajustam à imagem de entrada. Isto coloca a zero todos os elementos que ficam fora da matriz de entrada, produzindo uma saída maior ou de igual tamanho.

A camada de agrupamento (*Pool Layer*) conduz à redução da dimensão, reduzindo o número de parâmetros da entrada. Semelhante à camada convolucional, a operação de *pooling* varre com um filtro toda a entrada, mas a diferença é que este filtro não tem qualquer peso. Em vez disso, o *kernel* aplica uma função de agregação aos valores dentro do campo, povoando a matriz de saída. Existem diferentes tipos de agregação, ao destacar dois destes na Figura 9: *Max pooling* à medida que o filtro percorre a matriz de entrada, seleciona o pixel com o valor máximo e enviar para a matriz de saída; *Average pooling* é outra abordagem de agregação, onde o filtro ao percorrer a matriz de entrada, calcula o valor médio aritmética dentro do campo e envia para a matriz de saída. Estes dois tipos de agrupamento são frequentemente utilizados pelas suas características. O *Max pooling* auxilia a extrair recursos de baixo nível, como arestas, pontos, etc. Enquanto que a *Average pooling* é utilizada para recursos suaves encontrados nos dados.

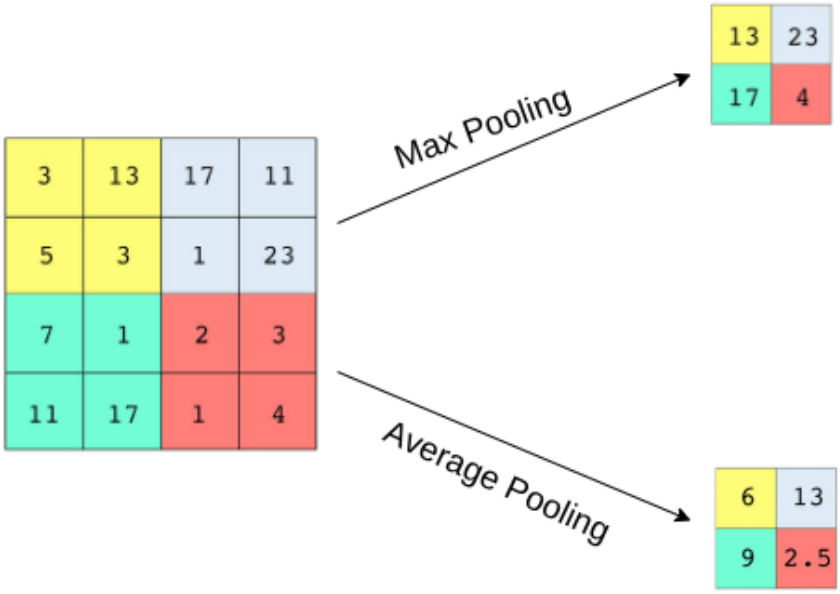


Figura 9: Dois tipos principais de agregação da camada *pooling* (adapta de [25]).

Uma camada de achatamento *(Flatten)* remodela o tensor para ter o mesmo número de neurónios contidos na camada totalmente conectada. Desta forma, transforma numa matriz 1D de elementos. Consequentemente, a camada totalmente conectada (*Fully-connected layer*) combina as características nos neurónios das camadas conectadas e ocultas.

1.3.3 Recurrent Neural Network (RNN)

As Redes Neuronais Recorrentes *(Recurrent Neural Network (RNN))* é um conjunto de algoritmos de redes neuronais artificiais para o processamento de dados sequenciais, como som, dados de séries temporais ou linguagem natural. As redes neuronais *feed-forward* comuns destinam-se apenas a pontos de dados, que são independentes uns dos outros. No entanto, se tivermos dados numa sequência tal que um determinado ponto de dado N dependa do ponto de dado n-1, é necessário a modificação da rede neuronal para incorporar as dependências entre esses pontos de dados resultando nas *RNN*.

Na Figura 10 é representada uma *RNN*, a camada de entrada X processa a entrada inicial e passa para a camada intermediária A. Posteriormente a camada intermediária consiste em várias camadas ocultas, cada uma com suas respetivas funções de ativação, pesos e *bias*. Estes parâmetros são padronizados em toda as camadas ocultas, pois as camadas ocultas são criadas a partir de um *loop*. Por exemplo se tiver que prever valores de ações usando dados simples como [20,45,60,50, ...], cada entrada de X0 a Xt conteria um respetivo valor, X0 conteria 20, X1 conteria 45 e estes valores seriam usados para prever o próximo número da sequencia, tal como demonstrado na Figura 10.

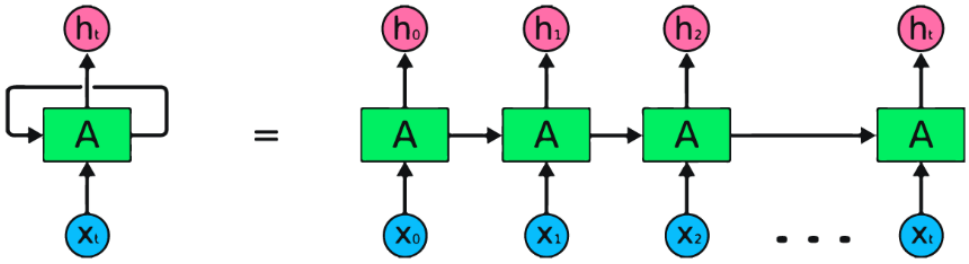


Figura 10: Recurrent Neural Network (RNN).

As *RNNs* não utilizam o algoritmo *backpropagation* tradicional utilizado na rede *CNN*. Ao invés disso, a *RNN* utiliza o algoritmo *Backpropagation Through Time (BTT)* para determinar o gradiente. Com o processo de *backpropagation*, a rede ajusta os parâmetros a partir do calculo dos erros. O *BTT* soma o erro em cada etapa de tempo, pois a *RNN* compartilha os parâmetros em cada camada oculta. As redes *RNNs* possuem também uma flexibilidade em relação ao comprimento das entradas e saídas que possuem, podendo estes serem alterá-los. Esta flexibilidade permite que as redes sejam aplicadas para gerar música, classificação de sentimentos e tradução automática, entre outras tarefas.

Há quatro tipos de *RNN*, como demonstrado na Figura 11, das quais apresentam diferentes tamanhos para entrada e saída da rede. Um para um é utilizada normalmente em *ML*, pois tem uma entrada e uma saída; Um para muitos, possui uma entrada e múltiplas saídas, a geração de música é um exemplo da aplicação para este caso; Muitos para um, processa uma sequencia de múltiplas entradas de diferentes passos de tempo produzindo uma única saída, sendo aplicadas por exemplo na análise de sentimentos, onde o rótulo da classe depende de uma sequência de palavras; Muitos para muitos, este tipo de rede processa múltiplas entradas e saídas, são aplicadas em tradução automática, por exemplo, sistemas de tradução de inglês para português ou vice-versa.

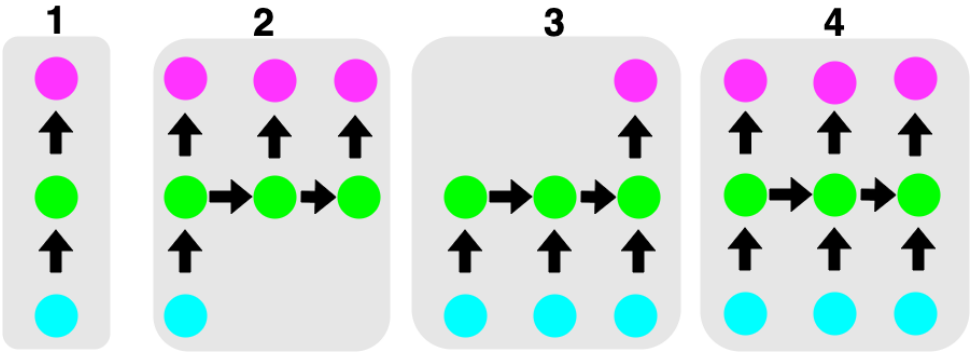


Figura 11: Tipos de *Recurrent Neural Network (RNN)*.

No entanto, as redes *RNNs* simples possuem duas limitações, que estão diretamente relacionadas com o gradiente, que é o declive da função de perda juntamente com a função de erro. Estas limitações são o *Vanishing Gradient* e *Exploding Gradient*. A primeira, *Vanishing Gradient*, é a limitação de quando o gradiente é pequeno, pois ele continua a diminuir, atualizando os parâmetros do peso até este tornar-se

insignificante - i.e. 0. Quando isto ocorre, o algoritmo já não esta a aprender mais. Já o *Exploding Gradient* é um problema que ocorre quando o gradiente se torna grande, criando modelos instáveis. Neste caso, os pesos do modelo crescem demasiado, e acabam por ser representados como *NaN*. Uma possível solução para estes problemas é reduzir o numero de camadas ocultas da rede, eliminando assim a complexidade das *RNNs*.

1.3.4 Long Short-Term Memory (LSTM)

Nos anos 90, os investigadores Hochreiter and Schmidhuber apresentaram uma variação da rede neuronal recorrente, a qual tem como propósito evitar os problemas de *Vanishing Gradient* e *Exploding Gradient*, chamada de *Long Short-Term Memory (LSTM)* [26]. Este tipo de rede neuronal é capaz de aprender dependências de longo prazo. As *LSTMs* ajudam a preservar o erro que pode ser transmitido por tempo e camadas. Ao manter um erro mais constante, elas permitem que as redes recorrentes continuem aprendendo durante vários passos de tempo, resolvendo o problema das dependências de longo prazo.

Assim como a *RNN*, a *LSTM* possui módulos de repetição, mas a estrutura é diferente, com mais de uma camada *tanh*. A *LSTM* possui quatro camadas que interagem entre elas, tal como demonstrado na Figura 12.

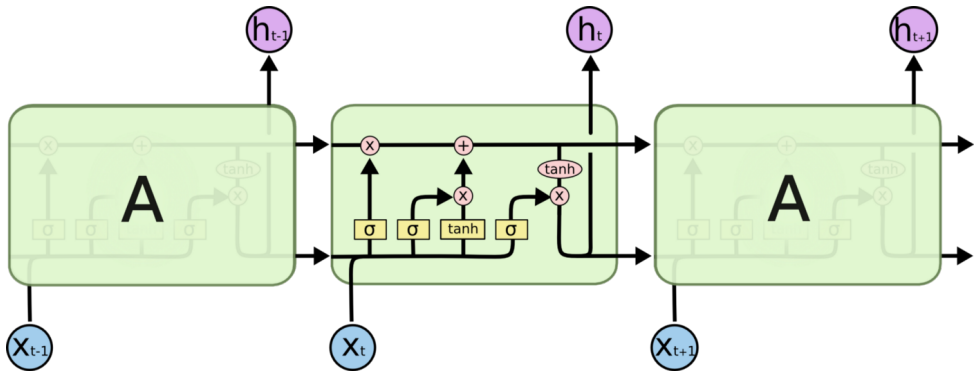


Figura 12: O módulo de repetição numa *Long Short-Term Memory (LSTM)*.

A *LSTM* tem a capacidade de remover ou adicionar informações ao estado da célula, esta sendo a linha horizontal que atravessa a parte superior do diagrama representado na Figura 13, cuidadosamente regulada por estruturas chamadas de portões. O estado da célula é como uma correia transportadora. Esta percorre toda a cadeia, com apenas algumas interações lineares menores. É muito fácil que as informações fluam ao longo dela sem alterações.

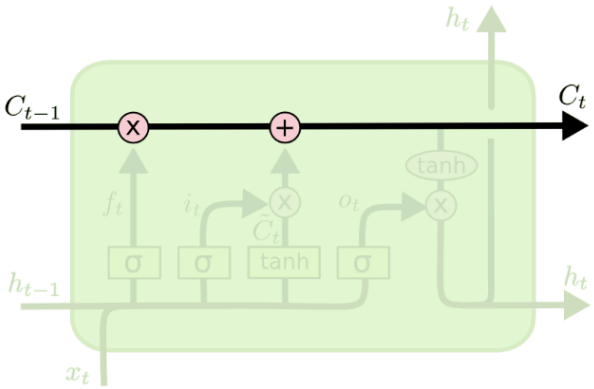


Figura 13: Estado da célula.

A primeira etapa da *LSTM* é decidir quais informações será descartada do estado da célula. Esta decisão é feita por uma função sigmóide chamada de portão de esquecimento. Este utiliza novas informações das quais será armazenadas no estado da célula, primeiro o estado atual X(t) e depois o estado oculto h(t-1) passam pela função sigmóide gerando valores entre 0 e 1. Sendo 1 a representar "manter isto completamente"enquanto o 0 representa "livre-se completamente disso". A seguir, a mesma informação do estado oculto e do estado atual será passada através da função tanh. Para regular a rede, o operador tanh criará um vetor (C (t)) com todos os valores possíveis entre -1 e 1. Estes valores de saída gerados a partir das funções de ativação são multiplicados pela saída da porta sigmóide. Por fim, o portão de saída determina o próximo valor do estado oculto. Este estado contém informação sobre as entradas anteriores. Os valores do estado atual X(t) e o estado oculto h(t-1) são enviados para porta da terceira função sigmóide. Depois, o novo estado da célula gerado é enviado através da função tanh e multiplica-se pela saída da porta sigmóide, para que apenas produza as partes importantes.

1.3.5 AutoEncoder (AE)

O *Autoencoder (AE)* é um tipo de rede neuronal de aprendizagem não supervisionada. Como o próprio nome da rede sugere, *AutoEncoder* auxilia na codificação dos dados. Uma rede *AE* primeiro codifica a informação numa representação latente de menor dimensão, posteriormente descodifica a representação latente de volta para uma informação numa outra dimensão. Um *AE* aprende a compactar os dados, enquanto minimiza o erro de reconstrução, tal como demonstrado na Figura 14.

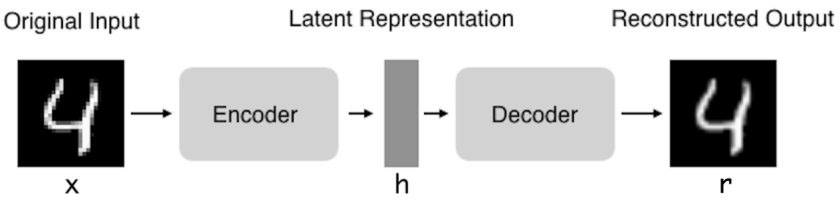


Figura 14: Rede neuronal *Autoencoder*.

Este tipo de rede neuronal é composta por dois elementos: Codificador(*Encoder*), o elemento da rede que compacta o dado da entrada numa representação de espaço latente (codificando a entrada), podendo ser representado por uma função de codificação h = f(x); e o Descodificador (*Decoder*) que tem como objetivo reconstruir a representação do espaço latente, representado pela seguinte função de descodificação r = g(h).

A remoção de ruídos e a redução de dimensão para visualização de dados são duas das principais aplicações do uso desta rede neuronal. Com restrições de dimensão e latências apropriadas aos *AE*s podem aprender projeções de dados mais relevantes que o *PCA* ou outras técnicas básicas. Uma variação do *AE*, *Variational Autoencoder (VAE)*, juntamente com a rede neuronal *GAN* são as duas redes neuronais de forte investigação para construção dos chamados modelos generativos em *DL* [27].

1.3.6 Generative Adversarial Networks (GAN)

Os autores Goodfellow et al. apresentaram uma arquitetura neuronal, intitulada *GAN* num trabalho publicado em 2014, da qual é composta por duas redes neuronais uma denominada Gerador(*Generator*) e outra de Discriminador(*Discriminator*) [4]. O Gerador é responsável pela criação de dados sintéticos, o Discriminador possui a finalidade de classificar se um determinado dado é real ou falsos (criado pela rede Gerador), tal como apresentado na Figura 15. Desta forma, as redes ajudam-se uma à outra. Os *AEs* bem como outras rede neuronais são treinadas apenas com uma função de perda, contudo, as *GANs* possuem funções de perdas para cada uma das suas redes (Gerador e Discriminador).

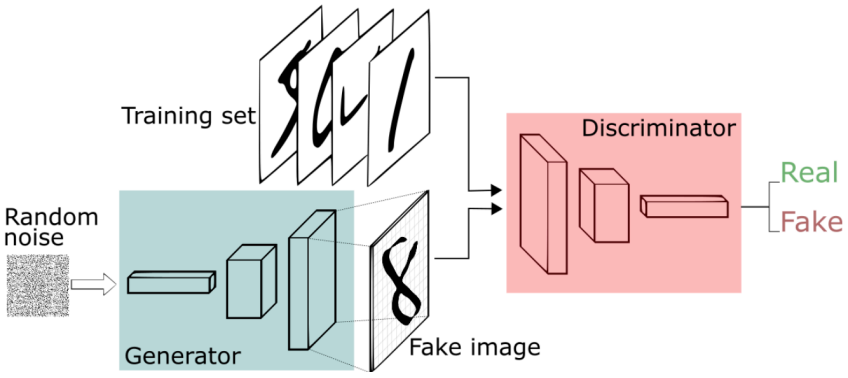


Figura 15: Arquitetura neuronal da *Generative Adversarial Networks (GAN)*

As duas redes Gerador (G) e o Discriminador (D), são treinadas com *backpropagation* utilizando a *loss* do Discriminador, tal qual apresentado na Figura 16. Para isso, o Discriminador tem como objetivo minimizar a *loss* tanto para os dados reais quanto para os falsos, enquanto a rede Gerador tenta maximizar a *loss* do Discriminador para os dados falsos que ela produziu [28]. Consequentemente afetando a função de custo, a função de custo do Gerador (G) é representada por $J^{(G)}$ e a do Discriminador representada por $J^{(D)}$. Os parâmetros treináveis (peso e *biases*) são representados pela letra grega theta: sendo $\theta^{(G)}$ para o Gerador e o Discriminador representado por $\theta^{(D)}$.

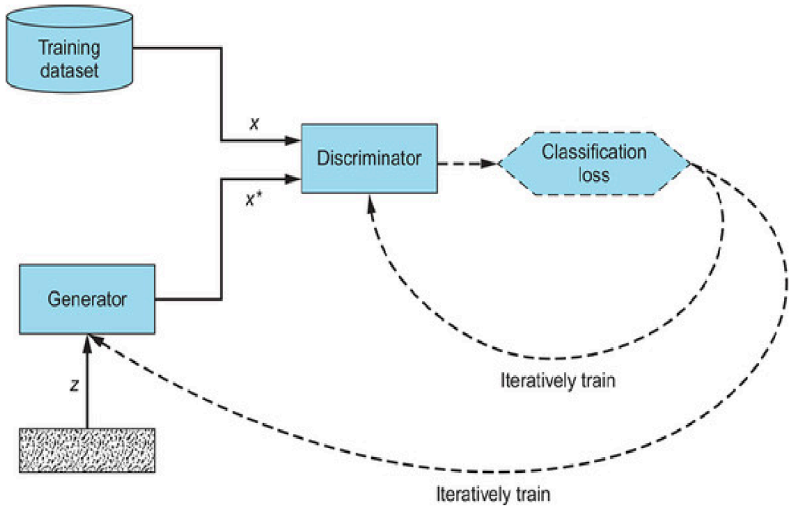


Figura 16: Processo de treino de uma *Generative Adversarial Networks (GAN)* [28]

Desta forma, a *GAN* é diferente das demais redes neuronais tradicionais, porque as redes tradicionais possuem suas funções de custo, J, definidas pelos seus próprios parâmetros de treino, θ, sendo expresso matematicamente $J(\theta)$. Ao contrário das *GANs* que possuem duas redes neuronais cujo a função de custo depende de ambos os parâmetros das redes. Desta forma a função de custo do Gerador é $J^{(G)}(\theta^{(G)}, \theta^{(D)})$ para o discriminador a função de custo trata-se da seguinte formula $J^{(D)}(\theta^{(G)}, \theta^{(D)})$ [29].

Outra diferença é que as redes neuronais tradicionais são capazes de otimizar todos os seus parâmetros, θ, durante o treino da rede. Diferentemente das *GANs*, cada rede só pode otimizar seus respetivos pesos e enviesamento. Consequentemente, o Gerador e o Discriminador só podem otimizar durante o treino os respetivos parâmetros da sua rede, $\theta^{(G)}$ e $\theta^{(D)}$.

Durante o treino de uma *GAN* temos o Gerador e Discriminador otimizando somente seus próprios parâmetros, podendo-se assimilar como um jogo onde os jogadores são as duas redes. Desta forma, o treino com o termino ideal seria quando se obtém o Equilíbrio de Nash, o ponto num jogo do qual nenhuma das redes pode melhorar a sua situação alterando as suas estratégias. Isto ocorre quando o Gerador tem a função de custo $J^{(G)}(\theta^{(G)}, \theta^{(D)})$ é minimizada em relação aos seus respetivos parâmetros $\theta^{(G)}$ e concomitantemente ocorra o mesmo com o a função de custo do Discriminador $J^{(D)}(\theta^{(G)}, \theta^{(D)})$ em relação aos seus parâmetros $\theta^{(D)}$.

1.3.7 Mecanismo de Atenção

As redes neuronais *RNNs*, em especial as *LSTMs*, são redes neuronais utilizadas para criação de arquiteturas neuronais *sequence-to-sequence (seq2seq)*, formadas por duas redes *RNNs* sendo um *encoder - decoder*. Esta arquitetura neuronal é utilizada em especial para criar um *Neural Machine Translator (NMT)*, o qual realiza uma tradução de forma automática, *Machine Translation*. Um modelo *seq2seq* recebe uma sequência de itens (palavras, letras, características de uma imagem entre outras) e gera outra

sequência de itens como saída.

Um exemplo de utilização no mundo real é o *Google Translate*, o qual utilizou esse modelo em produção no final de 2016 [30]. O *seq2seq*, tem em seu *Encoder* a função de captar o contexto da sequência de entrada sob a forma de um vetor de estado oculto e envia-o para o *Decoder*, que produz a sequência de saída. Um exemplo do caso de traduzir uma frase do francês (*Je suis brésillen*) para o inglês (*I am brazilian*), como demonstra a Figura 17. O *Encoder* processa cada item da sequência de entrada e compila as informações capturadas num vetor. Após processar toda a sequência de entrada, o *Encoder* envia o vetor para o *Decoder*, que começa a produzir a sequência de saída item por item.

Por design, uma *RNN* recebe duas entradas em cada etapa de tempo: uma entrada (no caso do *Encoder*, uma palavra da sentença de entrada) e um estado oculto. Sendo que o ultimo estado oculto (HS3) do *Encoder* é dado como entrada no *Decoder* como demonstrado na Figura 17. A desvantagem é que a sequência aonde obtém-se a saída do *Decoder* depende fortemente do contexto definido pelo último estado oculto do *Encoder* tornando difícil para o modelo lidar com sequências longas, porque há uma alta probabilidade de que o contexto inicial tenha sido perdido ao final da sequência.

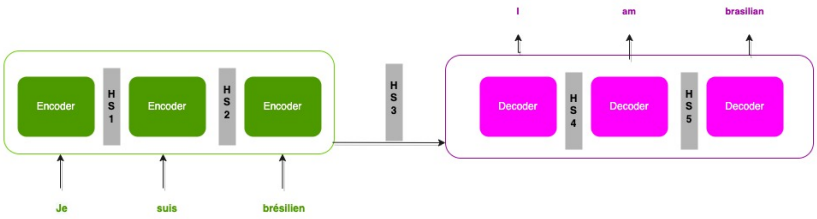


Figura 17: Modelo *Encoder-Decoder* para Modelagem *Seq2Seq* [31].

Desta forma os trabalhos [32, 33] propuseram uma técnica intitulada de *Attention*, que possibilita que o *Encoder*, no caso do exemplo anterior, se concentre em diferentes partes da sequência de entrada e consequentemente tendo como saída não somente um estado oculto com as informações condensadas dessas partes, mas também uma quantidade de vetores de estado oculto equivalente ao número de instâncias na sequência de entrada, tal como representado na Figura 18. O mecanismo proporciona ao *Decoder* receber todos os estados ocultos, cada um destes contendo o contexto acumulado até aquela palavra, com um *score* associado aquele estado oculto. Ao receber os estados, o *Decoder* multiplica cada estado oculto pela sua pontuação *softmaxed*, amplificando assim os estados ocultos com pontuações altas, e ignorando os estados ocultos com pontuações baixas.

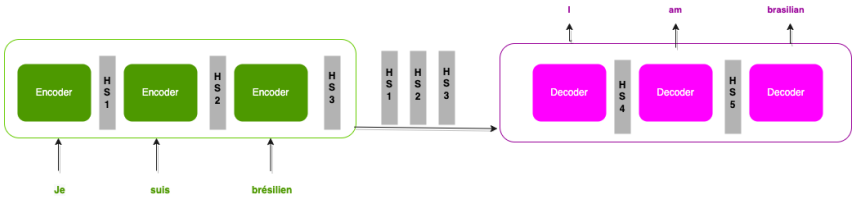


Figura 18: Modelo *Encoder-Decoder* para Modelagem *Seq2Seq* com mecanismo de *Attetion* [31].

1.3.8 Transformer

Em 2017 num trabalho intitulado *Attention Is All You Need*, foi apresentado uma nova arquitetura de rede neuronal *Transformer*[31]. Até então as *RNNs* eram consideradas as redes *standard* para o tratamento de dados sequenciais, e.g., tarefa de tradução. No entanto, o trabalho [34] apresentou uma arquitetura da qual utilizava o mecanismo de atenção, sem qualquer RNN, onde se obteve resultados superiores em relação ao estado da arte para tarefas de processamento de dados sequenciais. Este fato corrobora a informação que as *Transformers* são atualmente as novas redes *standard*, fazendo uso do mecanismo de atenção.

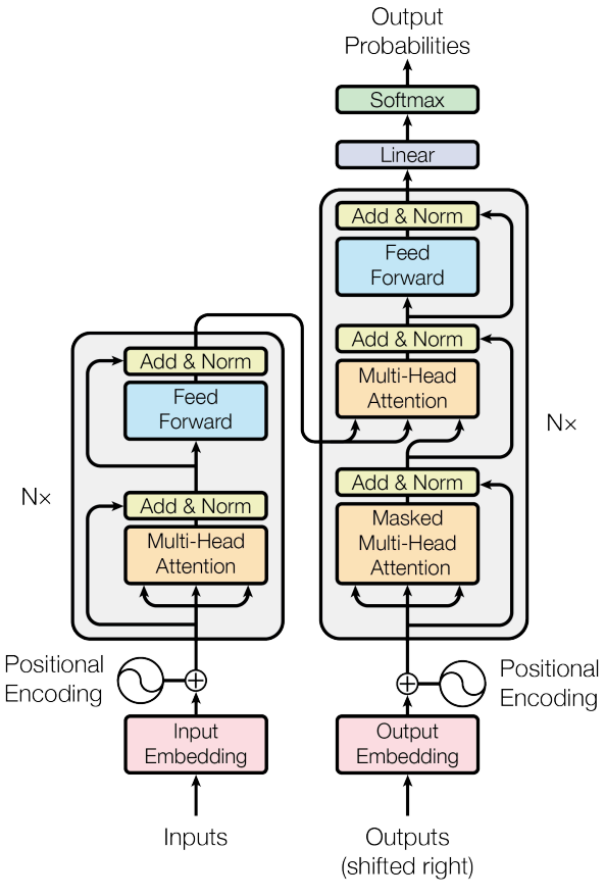


Figura 19: Arquitetura de uma *Transformer* [31].

Uma rede neuronal *Transformer* é composta por uma arquitetura *Encoder-Decoder*, como apresentado na Figura 19, o *Encoder* é descrito do lado esquerdo enquanto que o *Decoder* apresentado do lado direito da Figura 19. Tanto o *Encoder-Decoder* são compostos por módulos, dos quais podem ser formados por um empilhamento de módulos N vezes. Uma das principais diferença desta rede neuronal é que a sequência de entrada pode ser transmitida paralelamente, o que possibilita a utilização da *Graphics Processing Unit (GPU)*, consequentemente impactando na velocidade do treino da rede. A *Transformer* faz uso de múltiplas camadas de atenção. Desta forma, resolve-se o problema de *Vanishing Gradient* o qual é presente em outras redes como as *RNN*s.

No trabalho [31] a rede neuronal *Transformer* foi aplicada a um problema relacionado ao campo *NMT*, do qual obteve-se resultados relevantes. O trabalho [31] utilizou *Transformers* para demonstrar a superioridade desta em relação a outras redes neuronais em tarefas relacionadas a *Natural Language Processing (NLP)*. Para o entendimento da rede neuronal *Transformer* é necessário conhecimento dos dois blocos que a compõe, *Encoder* e *Decoder* e para isto será considerado o caso de aplicação do trabalho de origem da rede[31].

1.3.8.1 Bloco Encoder

O *Encoder*, representado na Figura 20, primeiramente recebe os dados de entrada sequencial, no caso descrito pelo trabalho [31], sendo sentenças composta por palavras das quais as máquinas não são capazes de processar. Desta forma, estas são convertidas para números, vetores e matrizes. Sendo que cada palavra é convertida num vetor o qual é incorporado num espaço n-dimensional, onde palavras de significados semelhantes são agrupadas ou estão próximas umas das outras naquele espaço. Outra questão a ser resolvida é que, em frases diferentes, cada palavra pode assumir significados diferentes. Então, para resolver este problema, usa-se codificadores posicionais (*Positional Enconding*), vetores dos quais contém o contexto de acordo com a posição da palavra numa frase.

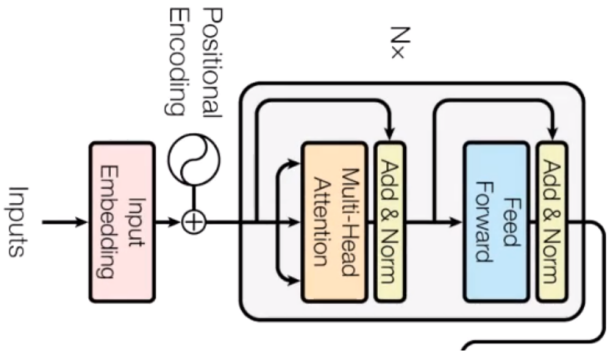


Figura 20: Bloco Encoder da Rede Neuronal Transformer (adapta de [31]).

O mecanismo de atenção é a essência das redes neuronais *Transformers*, o que consequentemente torna o componente *Multi-Head Atttention* o principal elemento, pois ele concentra no quanto uma determinada palavra é relevante em relação as demais palavras numa frase. Para cada palavra é gerado um vetor de atenção o qual captura a relação contextual entre as palavras naquela frase. No entanto, o valor da correlação da palavra X, sobre si mesma na frase é muito alto em relação a interação desta com as outras palavras da sentença. Desta forma, uma vez tendo vários vetores de atenção pelas respetivas palavras no final é utilizado a média ponderada para calcular o vetor de atenção final de cada palavra. Esta é a razão pela qual este processo é intitulado de atenção de várias cabeças *(Multi-Head Atttention)*.

Próxima etapa, os vetores de atenção são enviados a uma rede *feed-forward* simples onde seu objetivo principal é transformar os vetores de atenção numa forma que seja aceitável pelo próximo bloco de

Enconder ou *Decoder*. Como os vetores de atenção são independentes entre si, pode-se aplicar a paralelizarão, passando assim todas as palavras ao mesmo tempo para o bloco *Encoder* e obtendo o conjunto de vetores codificados para cada palavra simultaneamente.

1.3.8.2 Bloco Decoder

O *Decoder*, descrito visualmente na Figura 21, possui uma estrutura similar ao *Encoder*. No caso do trabalho [31] o qual propôs um *NMT*, suponha-se que seja um tradutor de inglês para francês utilizando uma rede neuronal *Transformer*. Desta forma, as frases em inglês passam pelo bloco *Encoder* enquanto que as frases em francês passam pelo bloco *Decoder*.

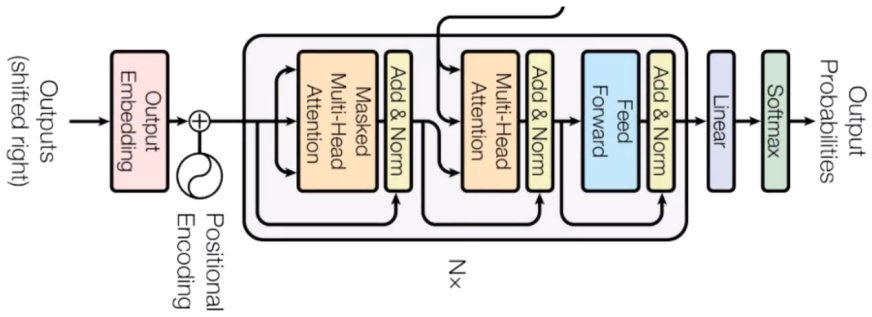


Figura 21: Bloco Decoder da Rede Neuronal Transformer (adapta de [31]).

A princípio, temos a camada *Embedding* e os codificadores posicionais (Positional Enconding) que transformam as palavras nos respetivos vetores, semelhante ao que foi descrito no bloco *Encoder*.

Os vetores, seguem para o componente com mecanismo de auto-atenção, onde são gerados vetores de atenção para cada palavra nas frases no idioma francês para representar o quanto cada palavra está relacionada as demais palavras da frase, assim como descrito no bloco *encoder*. No entanto, este componente é chamado de *Masked Multi-Head Attention Block*, funcionando como mecanismo de aprendizagem. Primeiro, é dado uma palavra em inglês, a qual será traduzida em sua própria versão em francês usando resultados anteriores, depois combinará e comparará com a tradução real em francês, da qual foi alimentada o bloco *Decoder*. Depois de comparar ambos, ele atualizará seu valor na matriz. Desta forma, este aprenderá após várias iterações. Se faz necessário ocultar a próxima palavra em francês, para que a princípio seja previsto a próxima palavra usando resultados anteriores, sem conhecer a palavra real traduzida a partir do inglês, pois a rede pode retirar qualquer palavra da frase em inglês, no entanto, só é permitido a retirada da palavra anterior da frase em francês para aprender. Assim, ao realizar a operação paralela na matriz, é atribuída na matriz uma mascara as palavras que aparecem posteriormente, transformando-as em 0's para que a rede de atenção não possa usá-las.

Desta forma, os vetores de atenção resultantes da camada *Masked Multi-Head Attention Block* e os vetores do bloco *Encoder* são passados para outro componente *Multi-Head Attention* do bloco *Decoder*. Sendo assim terá vetor de cada palavra das respetivas frases nos dois idiomas (inglês e francês). Assim

sendo a camada *Masked Multi-Head Attention Block* faz o mapeamento das palavras em inglês e francês e descobre a relação entre elas. Por fim, as saídas desta camada são vetores de atenção para cada palavra dos dois idiomas. Cada vetor representa a relação com outras palavras em ambas as línguas.

Da mesma forma, que os vetores de atenção são enviados para um rede *feed-forward* no bloco *Encoder*, isto acontece no bloco *Decoder* da rede *Transformer*. Esta rede *feed-forward* fará com que os vetores de saída se formem em algo que seja facilmente aceitável por outro bloco *Decoder* ou uma camada linear.

A camada linear é outra camada de alimentação. Ela é usado para expandir as dimensões em números de palavras no idioma francês após a tradução. Posteriormente é passado por uma camada *Softmax*, que transforma a entrada numa distribuição de probabilidade, que é interpretável por humanos. E a palavra resultante é produzida a partir da maior probabilidade após a tradução.

1.4 Motivação

A motivação desse trabalho surgiu a partir das consequências geradas a partir da criação e disseminação do conteúdo *deepfake* no meio digital. Como forma de mitigar esse problema, esse trabalho de investigação foca-se nas suas consequências, resultantes das aplicações dos modelos generativos para geração de conteúdo *deepfake*. Na atual conjetura imposta pela pandemia do COVID-19, há um grande aumento de conteúdo digital, mais especificamente de imagens e vídeos de cidadãos comuns. O aumento do uso das médias sociais e ferramentas de videoconferência, possibilita o fornecimento de mais recursos para a geração de conteúdo *deepfake*. Desta forma, esse trabalho tem como objetivo propor uma solução para detetar se um determinado conteúdo multimédia é *deepfake* ou não.

1.5 Objetivos e Resultados Esperados

Objetivo geral:

- Desenvolvimento de um algoritmo de IA capaz de classificar se um conteúdo multimédia é *deepfake*;
- Identificar e avaliar os casos mal classificados pelo algoritmo;
- Otimização do algoritmo de classificação.

Objetivos específicos:

- Levantamento bibliográfico do estado da arte, referente a geração de conteúdo *deepfake* ;
- Analise de vantagens e limitações do uso de algoritmos e modelos levantados no tópico anterior;
- Levantamento bibliográfica do estado da arte, referente a identificação de conteúdo multimédia gerado por via de algoritmos de Deep Fake;

- Desenvolvimento e implementação de métodos e algoritmos de classificação levantados, no tópico anterior, para obter uma classificação;
- Analisar e validar os resultados do classificador desenvolvido, em específico os casos mal classificados, e otimizar o algoritmo para melhorar os resultados de classificação.

Os resultados esperados desse projeto são: análise e comparação de resultados de múltiplos algoritmos para geração de conteúdo multimédia *deepfake*; proposta de um algoritmo de classificação capaz de identificar se um determinado conteúdo multimédia é real ou manipulado sintaticamente, bem como a sua otimização.

1.6 Contribuições

As contribuições desse trabalho são resultados a partir de experimentos empíricos, dos quais um algoritmo, ao ser treinado com dados *deepfake* gerados com uma técnica X, possuir uma capacidade generalista de classificar dados *deepfake* criados a partir de outra técnica Y. Para isto é aplicado um algoritmo multi-classificação para mitigar tal problema, bem como a criação de um nível de confiança associado a classificação do algoritmo ao estimar uma determinada classe ao analisar um dado, desta forma dando mais confiança a predição do modelo.

1.7 Metodologia

Atividades	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**
Levantamento do estado da arte	X	X	X						
Experiência com classificação multimédia do estado da arte	X	X	X						
Investigação na geração sintético	X	X	X	X					
Aquisição e preparação de *datasets*			X	X					
Implementação de algoritmos Deep Leanring				X	X	X			
Validar resultados do tópico anterior					X	X			
Optimizar o modelo					X	X	X		
Escrita da dissertação	X	X	X	X	X	X	X	X	X

Tabela 2: Calendarização do trabalho.

Esse trabalho foi realizado um levantamento do estado da arte sobre deteção de conteúdo multimédia *deepfake*, paralelamente foi investigado sobre classificação de conteúdo multimédia em especifico classificação de vídeo. Houve também uma investigação sobre a geração de dados sintéticos utilizando-se de modelos generativos.

Consequentemente esse projeto analisou os *datasets* presentes no estado da arte, posteriormente realizou-se a aquisição e preparação dos dados, para validar a implementação de um detetor de conteúdo *deepfake* o qual utiliza-se de arquiteturas dos algoritmos de DL. Uma vez executados experimentos, foram

realizados validações e otimização do modelo, das quais demonstram qual algoritmo deveria ser utilizado no detetor. Concomitantemente foi realizado o desenvolvimento da escrita da dissertação.

1.8 Estrutura do Documento

Esta dissertação está estruturada em 4 capítulos, com os seguintes capítulos:

- **Capitulo 1 - Introdução :** este capítulo contextualiza a problemática e introduz a teórica sobre os conceitos abordados nesse trabalho, bem como apresenta os principais objetivos e metas desse trabalho e contribuições para o estado da arte;
- **Capitulo 2 - Estado da Arte:** descreve o levantamento de informações sobre o contexto de deteção de conteúdo multimédia *deepfake*, onde é explicitado o estado da arte. Posteriormente neste capítulo, será feita uma visão geral dos trabalhos relacionados a respeito da deteção de conteúdo *deepfake*;
- **Capitulo 3 - Desenvolvimento:** são descritas as experiências realizadas para classificação de vídeos *deepfake*, sendo descrito os datasets existentes, bem como as abordagens e pipeline adotados na aplicação de algoritmos de DL utilizados na implementação de um detetor de conteúdo *deepfake*;
- **Capitulo 4 - Conclusão:** por fim, este capítulo demonstra a síntese das discussões e conclusões do projeto, bem como descreve as limitações e algumas possibilidades de investigações a realizarem-se posteriormente.

Capítulo 2

Estado da Arte

Para esse projeto foi realizado uma análise exploratória em estudos relacionados com o tema em questão, detetor de conteúdo *deepfake*. Consequentemente, foi executada uma investigação por meio de uma analise exploratório sobre os principais artigos científicos encontrados no estado da arte. Desta forma, foi realizado uma Revisão Sistemática da Literatura (RSL) para selecionar os trabalhos relacionados deste trabalho.

A RSL é uma metodologia de carácter exploratório para investigações cientificas. Para [35], a RSL é um estudo secundário, produto da análise de estudos primários sobre uma determinada temática dentro de um tópico estabelecido. Este estudo tem como objetivo proporcionar os investigadores conhecimento a respeito de um determinada área de conhecimento.

A RSL nasceu nas ciências sociais na década de 80. Consequentemente, alargou-se para as demais áreas do conhecimento, a exemplo a medicina [36] pela necessidade de captar, reconhecer e sintetizar as evidências científicas de N resultados num número de produções científicas sobre a mesma temática [37].

Posto isto, o levantamento de informações a respeito do estado da arte desse trabalho foi realizado utilizando-se de uma RSL. Devido a capacidade desta ferramenta em auxiliar na síntese da ampla gama de configurações e métodos empíricos aplicados a uma determinada temática [38], esta sendo a deteção de conteúdo multimédia *deepfake*.

O conteúdo multimédia *deepfake* têm se tornado popular devido à crescente qualidade destes. De igual forma à facilidade que os aplicativos proporcionam utilizadores com distintos graus de conhecimento computacional, de profissionais a novatos, a criarem conteúdo multimédia *deepfake*. Estes aplicativos são desenvolvidos utilizando-se de algoritmos de *Deep Learning* dos quais apresentam uma capacidade de representar dados complexos e de alta dimensão. E.g., os *Deep Auto-Encoders* aplicados para redução de dimensionalidade e compressão de imagem [39]. A primeira tentativa de criação de *deepfake* foi o *FakeApp* [40], da qual utiliza uma estrutura de emparelhamento *autoencoder-decoder*. Nesse método, o *autoencoder* extrai características latentes das imagens faciais e o *decoder* é usado para reconstruir as imagens faciais.

Os procedimentos para realizar uma RSL têm como principal objetivo a reprodutibilidade da execução

da RSL. Os procedimentos proposto pelo trabalho [38] para RSL são sub-dividido em três etapas:

1. Planeamento - Planning the Review;
2. Execução - Conducting the Review;
3. Análise e Divulgação dos Resultados - Reporting the Review.

2.1 Planeamento

É elaborado um protocolo, este composto por um conjunto de etapas e critérios básicos para a execução da RSL [41]. No planeamento são definidas as questões principais de investigação, as palavras-chaves para delinear o tópico de investigação, os critérios de inclusão e exclusão dos trabalhos relacionados encontrados. As questões de pesquisas, na Tabela 3.

ID da questão	Questão de Pesquisa
Q1	Qual o ano de publicação do trabalho ?
Q2	Qual o tipo do conteúdo multimédia classificado?
Q3	Qual o algoritmo e respetiva arquitetura aplicado para classificação ?
Q4	Qual o *dataset* analisado ?

Tabela 3: Questões de Pesquisa

Para responder à essas questões, foi necessário selecionar os trabalhos presentes nas bases de dados científicas. Desta forma, estabeleceu-se um critério temporal. Os trabalhos analisados foram com o intervalo temporal de publicação dos últimos 6 anos (2014 a 2020), este intervalo foi escolhido dado a natureza da temática, como anteriormente explicado, pois em 2014 teve o surgimento das *GAN*s e o tema *deepfake* obteve destaque a partir de então. Os estudos deverão ser escritos em inglês ou português. Para seleção para dos trabalhos foi utilizado um conjunto de palavras-chaves, as quais caracterizem a temática desse trabalho. Estas palavras-chaves estão discriminados na Tabela 4.

Palavras Chaves
Deep Learning;
Multimedia
*deepfake*s
Detection; Classifier

Tabela 4: Palavras chaves

O critério para seleccionar as bases de dados científicas foram a sua acessibilidade de forma *online*. Como também detenham trabalhos relacionados apresentados em eventos e conferências correlacionados ao tema de deteção de conteúdo *deepfake*. É imprescindível que os trabalhos presentes nessas bases de dados cientificas sejam acessíveis integralmente para o autor desse trabalho, enquanto estudante pesquisador da Universidade do Minho.

Consequentemente, as bases de dados seleccionadas foram 3. A base de dados da Institute of Electrical and Electronics Engineers (IEEE)[1], pois contém publicações de conteúdo técnico e científica do IEEE e parceiros; a base da Association for Computing Machinery (ACM)[2], esta base de dados é especializada em computação; A **arXiv**[3] também foi seleccionada como uma base de dados, apesar de ser uma base de dados aberta e sem propriamente uma revisão de pares. Esta base apresenta trabalhos recentes nesse tema e de grande relevância, por vezes trabalhos relacionados indisponível em bases como IEEE e ACM, encontram-se disponíveis na base do arXiv. Para além dos trabalhos lidos das bases seleccionadas, foram analisados trabalhos de outras fontes de acordo com sua relevância para esse trabalho.

Uma vez seleccionadas as questões de pesquisa, palavras-chaves e bases de dados científicas. Foi definido os Critérios de Inclusão (CI) e Critérios de Exclusão (CE) na Tabela 5.

Critérios de Inclusão	Critérios de Exclusão
CI1- Trabalhos dos quais apresentem métodos e/ou técnicas para deteção de conteúdo multimédia *deepfake*;	CE1 - Trabalhos duplicados;
CI2 - Trabalhos dos quais tenham como objecto de estudo, conteúdo multimédia;	CE2 - Trabalhos dos quais não estejam disponíveis integralmente em bases de dados científicas ou em versões impressas;
CI3- Trabalhos dos quais não abordem a técnica utilizada para classificação.	CE3 - Trabalhos dos quais abordem assunto divergente a deteção de *deepfake*;
	CE4 - Trabalhos dos quais sejam livros;
	CE5 - Trabalhos dos quais não forem escritos em Inglês ou em Português;
	CE6 - Trabalhos publicados dos quais não se enquadrem no intervalo temporal estipulado 2014 a 2020;
	CE7 - Trabalhos dos quais não menciona sequer uma técnica e/ou algoritmo de *Deep Learning* para classificação de *deepfake*.
	CE8 - Trabalhos dos quais não apontem os *datasets* utilizados.

Tabela 5: Critérios de inclusão (CI) e de exclusão (CE) para na seleção de trabalhos relevantes.

2.2 Execução

Após o planeamento da RSL, o qual resultou 3000 trabalhos presentes nas bases de dados selecionadas. Destes foram selecionados 10% dos quais foi realizada uma primeira leitura dos seus respetivos títulos, resumos e conclusões. Nesta primeira leitura foram aplicados os critérios de inclusão e exclusão.

[1]Site da base IEEE: https://ieeexplore.ieee.org
[2]Site da base ACM:https://dl.acm.org/
[3]Site da base arXiv:https://arxiv.org/search/cs

Posteriormente, foram selecionados 70 trabalhos submetidos a uma nova leitura completa do trabalho, e novamente aplicando os mesmos CI e CE, filtrando os estudos. Consequentemente, realizou-se uma síntese dos mesmos e a exclusão dos trabalhos considerados incompletos, indisponíveis ou que não correspondam ao tópico estabelecido.

2.3 Trabalhos Relacionados

Os trabalhos sintetizados abaixo foram subtraídos dos trabalhos resultantes da execução da RSL. Esta sintetize descreve o estado da arte destacando técnicas, algoritmos, conjunto de dados utilizados para para o treino dos modelos de deteção de conteúdo multimédia *deepfake*.

2.3.1 DeepFakesON-Phys: Deteção de deepFakes baseado em estimativa de frequência cardíaca

No trabalho de Hernandez-Ortega et al.[42] é proposto um classificador para deteção de conteúdo multimédia *deepfake*, no formato de vídeo. Os *datasets* abordados neste trabalho foram os *Celeb-DF v2* e *DFDC*. O trabalho combina uma arquitetura CNN , especificamente uma rede *Convolutional Attention Network (CAN)*. Esta arquitetura é descrita juntamente com o pré-processamento necessário realizado nos vídeos na Figura 22.

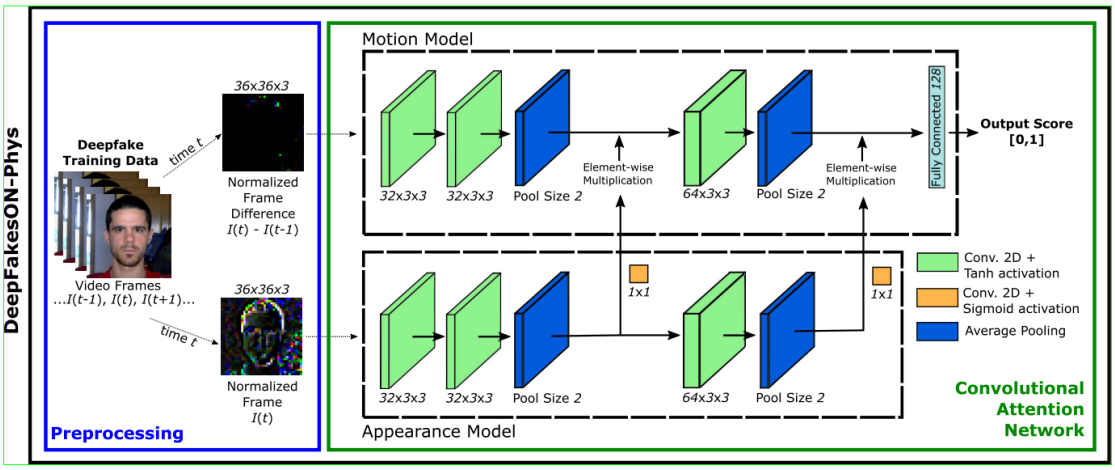


Figura 22: Arquitetura da *Convolutional Attention Network* [42].

Este trabalho utilizou para além da arquitetura neuronal, propôs a utilização do mecanismo de atenção para analisar as questões fisiológicas do indivíduo ao detetar se o conteúdo é um *deepfake* ou não. Para isso, utilizou-se de uma técnica intitulada *photoplethysmography(rPPG)* da qual é usada para detetar mudanças no volume sanguíneo no leito micro-vascular do tecido. Desta forma, a hipótese deste trabalho é que os vídeos *deepfake* não reproduzem essa mudança sanguínea natural durante uma fala, como normalmente ocorreria em um vídeo real.

Ao aplicar tal arquitetura obteve-se resultados significativos, AUC de 99,9% e 98,2% nos *datasets* *Celeb-DF* v2 [43] e *DFDC* [44], respetivamente. A métrica AUC refere-se a capacidade da qual modelo detém em classificar corretamente os dados nunca visto, sendo que quanto mais próximo do valor 1 maior é o êxito do modelo obter um resultado na classificação de 100% e quanto mais distante desse valor se traduzem numa classificação errada do modelo. Esta métrica utiliza a taxa de verdadeiro positivo contra a taxa de falsos positivos.

Os resultados deste trabalho superaram outros detetores de *deepfake* de última geração com base em *features* de distorção de rosto e de *Deep Learning*, entre outros. Por fim, os resultados experimentais deste estudo demonstrou que as técnicas atuais de manipulação facial não dão atenção às informações fisiológicas relacionadas à frequência cardíaca ou ao sangue. Uma comparação de resultados em relação a outros trabalhos do estado da arte é apresentado na Figura 23. Na primeira análise o detetor de *deepfake* proposto por este trabalho destacou-se em relação aos demais. No entanto, a comparação não é totalmente justa uma vez que os *datasets* utilizados diferem dos demais trabalhos ao qual é realizado a comparação dos resultados.

Study	Method	Classifiers	Best Performance	Databases
(Matern, Riess, and Stamminger 2019)	Visual Features	Logistic Regression MLP	AUC = 85.1% *AUC = 78.0%* *AUC = 66.2%* *AUC = 55.1%*	Own *FF++ / DFD* *DFDC Preview* *Celeb-DF*
(Li and Lyu 2019; Li et al. 2020)	Face Warping Features	CNN	AUC = 97.7% *AUC = 93.0%* *AUC = 75.5%* *AUC = 64.6%*	UADFV *FF++ / DFD* *DFDC Preview* *Celeb-DF*
(Rössler et al. 2019)	Mesoscopic Features Steganalysis Features Deep Learning Features	CNN	Acc. ≃ 94.0% Acc. ≃ 98.0% Acc. ≃ 100.0% Acc. ≃ 93.0% Acc. ≃ 97.0% Acc. ≃ 99.0%	FF++ (DeepFake, LQ) FF++ (DeepFake, HQ) FF++ (DeepFake, RAW) FF++ (FaceSwap, LQ) FF++ (FaceSwap, HQ) FF++ (FaceSwap, RAW)
(Nguyen, Yamagishi, and Echizen 2019)	Deep Learning Features	Capsule Networks	*AUC = 61.3%* *AUC = 96.6%* *AUC = 53.3%* *AUC = 57.5%*	*UADFV* *FF++ / DFD* *DFDC Preview* *Celeb-DF*
(Dang et al. 2020)	Deep Learning Features	CNN + Attention Mechanism	AUC = 99.4% EER = 3.1%	DFFD
(Dolhansky et al. 2019)	Deep Learning Features	CNN	Precision = 93.0% Recall = 8.4%	DFDC Preview
(Sabir et al. 2019)	Image + Temporal Features	CNN + RNN	AUC = 96.9% AUC = 96.3%	FF++ (DeepFake, LQ) FF++ (FaceSwap, LQ)
(Tolosana et al. 2020a)	Facial Regions Features	CNN	AUC = 100.0% AUC = 99.5% AUC = 91.1% AUC = 83.6%	UADFV FF++ (FaceSwap, HQ) DFDC Preview Celeb-DF
(Conotter et al. 2014)	Physiological Features	-	Acc. = 100%	Own
(Li, Chang, and Lyu 2018)	Physiological Features	LRCN	AUC = 99.0%	UADFV
(Agarwal and Farid 2019)	Physiological Features	SVM	AUC = 96.3%	Own (FaceSwap, HQ)
(Ciftci, Demir, and Yin 2020)	Physiological Features	SVM/CNN	Acc. = 94.9% Acc. = 91.5%	FF++ (DeepFakes) Celeb-DF
(Jung, Kim, and Kim 2020)	Physiological Features	Distance	Acc. = 87.5%	Own
(Qi et al. 2020)	Physiological Features	CNN + Attention Mechanism	Acc. = 100.0% Acc. = 100.0% *Acc. = 64.1%*	FF++ (FaceSwap) FF++ (DeepFake) *DFDC Preview*
DeepFakesON-Phys [Ours]	**Physiological Features**	**CAN**	**AUC = 99.9%** **AUC = 98.2%**	**Celeb-DF v2** **DFDC Preview**

Figura 23: Tabela de comparação de resultados entre os modelos de deteção de conteúdo *deepfake* [42].

2.3.2 Uma rede residual baseada em LSTM convolucional para deteção de vídeo deepfake

No trabalho Tariq et al. [45] é descrito o surgimento de N métodos de deteção de *deepfake* com alta precisão de teste *zero-shot* em *datasets* de *deepfake* cujo o treino é realizado com dados específico de uma determinada técnica de geração *deepfake*. Consequentemente, estes não possuem um capacidade de generalização ao classificar dados produzidos a partir de outra técnica, proporcionando aos modelos a capacidade limitada de obterem somente um alto desempenho na classificação dos dados *deepfake*, criados por uma determinada técnica igual aos dados utilizados no treinado do modelo.

Estudos sobre o desenvolvimento de um detetor genérico de *deepfake* contém limitadas investigações [46], pois a maioria das abordagens de classificação utilizam camadas CNN das quais apresentam excelentes desempenhos na deteção das manipulações nas imagens quando são submetidas, durante o treino, a uma especifica técnica de criação *deepfake*.

Portanto, este trabalho propôs uma solução utilizando o *dataset FaceForensics++ (FF++)* [47] o qual contem oito *sub-datasets*, dos quais dois são referentes a vídeos reais e seis de vídeos *deepfake*. Destes os autores selecionaram quatro *sub-datasets deepfake (Deepfakes (DF), Face2Face (F2F), FaceShifter(FS), FaceSwap(FS), NeuralTextures(NT))* e o *sub-dataset* (*youtube*) que foi utilizado para geração dos vídeos dos *sub-datasets deepfake*.

Consequentemente, a abordagem adotada por este estudo foi destacar os pequenos artefactos entre as imagens consecutivas dentro de vídeo *deepfake*, como demonstrado na Figura 24. Sendo o (a) e (b) a enésima e (n + 1)ésima imagens, respetivamente. Para vídeos originais, estas imagens são fortemente semelhantes. Contudo, para vídeos *deepfake*, existem inconsistências. E.g., a diferença entre a enésima e a (n + 1) enésima imagem presentes nas imagens (c).

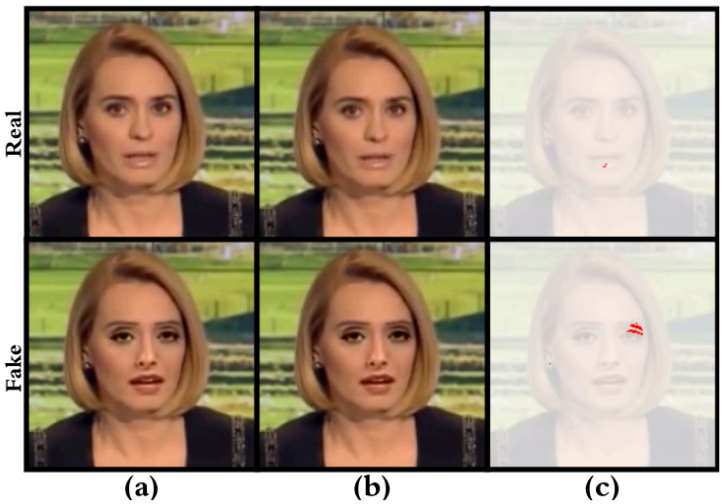


Figura 24: Diferença entre dois quadros consecutivos de um vídeo *deepfake* vs vídeo real [45].

Posto isto, este trabalho propôs uma arquitetura neuronal da qual as informações temporais, entre as imagens do vídeo, sejam avaliados na classificação do vídeo. Para incorporar o aspecto temporal, utilizou-

se de camadas do tipo Recurrent Neural Network (RNN), especificamente uma *LSTM* Convolucional, uma vez que esta é útil para tais tarefas. os autores propuseram então a *CLRNet*, descrita na Figura 25, uma Rede Residual baseada em *LSTM* Convolucional para deteção dos vídeos *deepfake* usando aprendizado por transferência.

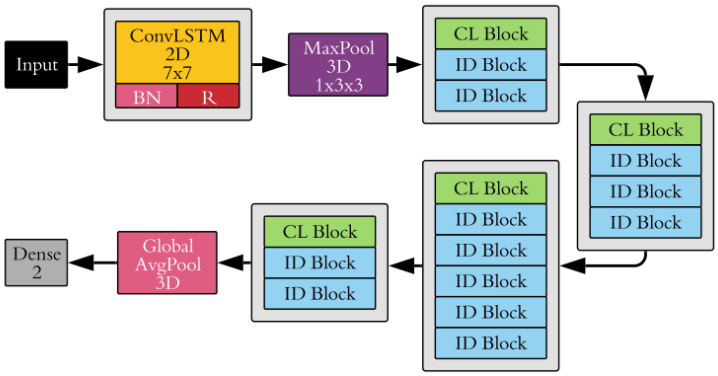


Figura 25: Arquitetura da *CLRNet* [45].

Esta arquitetura é descrita pelos blocos *CL Block*, representado pela Figura 26, e o *ID Block* representado pela Figura 27. Estes são elementos da arquitetura neuronal proposta pelos autores.

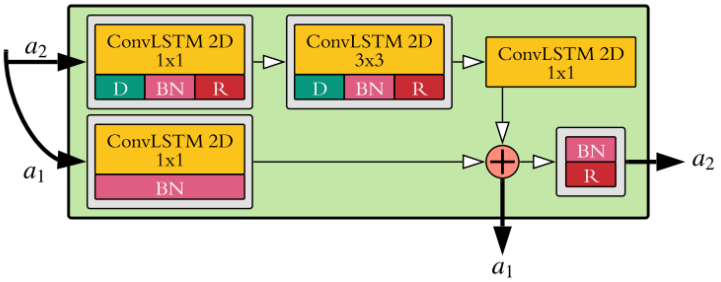


Figura 26: Representação do CL Block [45].

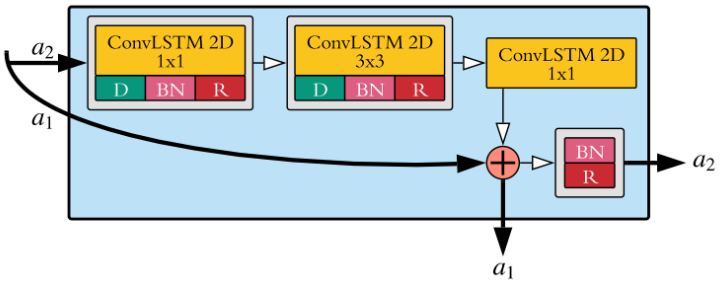


Figura 27: Representação do ID Block [45].

A configuração utilizada foi: Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz com 256.0GB RAM e NVIDIA

GeForce Titan RTX. As *frameworks* utilizadas foram o *TensorFlow v1.13.0*, *Keras Library* e a linguagem de programação *Python* v3.7.5 para implementação da *CLRNet*.

Este trabalho demonstrou resultados positivos na generalização da deteção de conteúdo *deepfake* independente do método utilizado para criação do vídeo *deepfake*. A Figura 28 descreve a comparação de desempenho da *CLRNet*.

O *CLRNet* utilizou no treino dados de um determinado *sub-dataset*. Posteriormente, avaliou com dados de outro *sub-dataset*. Nesta comparação obteve-se desempenho mediano para quase todos os *sub-datasets*, com exceção do *sub-dataset FS*. Estes resultados quando a rede tinha seu treino do zero.

Method	Base Dataset	DF (%)	FS (%)	F2F (%)	NT (%)	DFD (%)
ShallowNet‡	DF	56.65	**50.32**	54.15	35.13	41.23
Xception		86.00	49.78	57.26	61.49	**68.10**
FF++ [35]		96.36	-	-	-	-
Sabir *et al.* [36]		96.90	-	-	-	-
FT‡		**99.35**	50.00	**74.12**	**83.54**	48.45
CLRNet (Ours)		99.02	50.00	53.73	69.75	60.38
ShallowNet‡	FS	47.09	52.75	47.50	46.96	**49.58**
Xception		**49.49**	85.37	**56.61**	52.05	48.08
FF++ [35]		-	90.29	-	-	-
Sabir *et al.* [36]		-	94.35	-	-	-
FT‡		48.86	96.17	54.39	47.45	36.66
CLRNet (Ours)		48.53	**98.05**	50.15	**53.33**	49.13
FF++ [35]	NT	-	-	-	80.67	-
CLRNet (Ours)		**50.12**	**49.93**	**49.80**	**99.50**	**50.00**
CLRNet (Ours)	DFD	**65.08**	**51.92**	**60.32**	**63.10**	**96.00**

Figura 28: comparação de desempenho usando a aprendizagem zero-shot nos *datasets* [45].

Este trabalho propôs uma aprendizado por transferência com três estratégias 1) Origem única para destino único: o treino do modelo com dados de um *sub-dataset deepfake* X e consequentemente realizado o aprendizado por transferência a um treino de outro modelo utilizando um *sub-dataset* Y (e.g, realizado o treino com o *sub-dataset DF* e aplicado aprendizado por transferência para *sub-dataset FS*); 2) Múltipla origem para destino único: O treino é realizado com múltiplos *sub-datasets* para aplicar o aprendizado por transferência num *sub-dataset* específico (e.g, realizado o treino com dados de diferentes *sub-datasets deepfake* e aplicado aprendizado por transferência no modelo usando os dados do *sub-dataset F2F*; 3) Origem única para destino múltiplo: utiliza-se no treino os dados de um *sub-dataset* e utiliza-se uma quantidade de dados menor com múltiplos dados de origem de diferentes *sub-dataset* e aplica a transferência de aprendizado no treino do modelo. Na Figura 29 é descrito uma comparação do desempenho das diferentes estratégia de aprendizado por transferência aplicado aos modelos criados com arquitetura *CLRNet*.

Method	Source	Target	DF (%)	FS (%)	F2F (%)	NT (%)	DFD (%)
Single–source to Single–target							
FT‡	DF	FS	62.59	47.72	**61.34**	**68.51**	51.38
CLRNet			**90.95**	**83.08**	48.68	65.00	**53.87**
FT‡		F2F	83.70	**54.92**	79.31	**82.24**	54.92
CLRNet			**97.18**	49.28	**88.35**	78.05	**68.13**
FT‡	FS	DF	76.94	44.93	**66.42**	70.53	49.78
CLRNet			**92.47**	**96.33**	65.58	**75.80**	**59.13**
FT‡		F2F	56.06	55.74	53.73	55.38	47.63
CLRNet			**79.87**	**93.93**	**87.48**	**78.00**	**52.50**
FT‡		NT	-	-	-	-	-
CLRNet			51.82	98.12	50.90	95.50	49.13
FT‡		DFD	-	-	-	-	-
CLRNet			70.97	96.15	51.85	77.02	88.88
Multi–source to Single–target							
FT‡	FS+DF	F2F	64.08	45.17	55.98	62.09	51.38
CLRNet			**92.15**	**94.37**	**86.22**	**79.75**	**63.12**
Single–source to Multi–target							
CLRNet (best)	FS	DF+F2F	**91.77**	**94.55**	**87.75**	85.12	69.13
		DF+F2F +NT	90.67	94.35	85.78	**91.47**	69.50
		DF+F2F +NT+DFD	91.23	93.70	87.50	91.30	**87.13**

Figura 29: comparação de desempenho usando o aprendizado *zero-shot* sob os *datasets* [45].

2.3.3 Detecção de vídeo deepfake usando redes neuronais recorrentes

No trabalho de Güera and Delp [48] descreve o crescente uso de técnicas de geração de conteúdo *deepfake* por utilizadores sem um alto grau de conhecimento de manipulação de conteúdo multimédia. Este trabalho destacou o uso de aplicações moveis e *desktop* como *FaceApp* [49] e *FakeApp* [50]. O *FaceApp* é uma aplicação móvel, a qual possibilita gerar automaticamente manipulações altamente realistas nas faces utilizando-se de fotografias. Esta permite mudar o estilo do cabelo, sexo, idade e outros atributos. O *FakeApp* é uma aplicação *desktop*, a qual auxilia na construção dos vídeos *deepfakes*. Esta utiliza uma técnica para geração de conteúdo multimédia *deepfake*, da qual utiliza de *autoenconders*, descrito na Figura 30. Destacando o que torna o *deepfake* possível é a capacidade de forçar os duas faces serem codificadas nos mesmos vector de características para um conteúdo alvo. Isso é resolvido com duas redes compartilhando o mesmo *encoder*, mas usando dois *decoders* diferentes, apresentado na Figura 30 na etapa de *"Training"*.

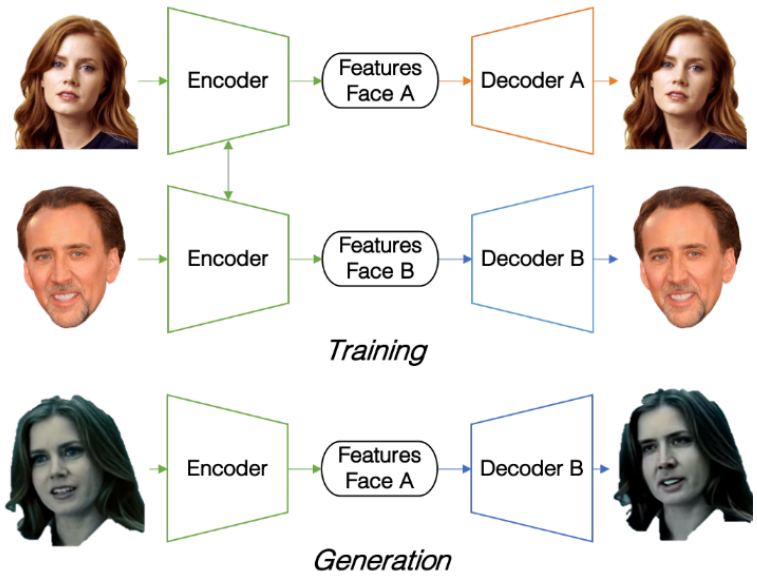


Figura 30: Processo de geração de uma *deepfake* [48].

Durante este processo o trabalho fez apontamentos acerca das fraquezas, durante a geração de conteúdo *deepfake*, como as inconsistências na saturação da cor da pele da face da pessoa do vídeo original e a pessoa alvo que se quer gerar no *deepfake*. Também há uma questão da iluminação presente de uma imagem a outra, quando a manipulação é feita em vídeo. Um terceira fraqueza apontada pelo estudo é relacionada ao processo de geração do vídeo final. Como o *encoder* automático utiliza-se de imagem a imagem, ele não tem conhecimento de nenhuma face da imagem anterior criado. Esta falta de consciência temporal é a fonte de múltiplas anomalias o que resulta em inconsistências numa imagem para outra. Contudo, isso por vezes não é perceptível ao olho humano.

Posto isto, os autores deste trabalho propuseram uma arquitetura para a qual pudessem extrair essas características dos imagens para analisar possíveis anomalias considerando o espaço temporal entre as imagens. A arquitetura deste trabalho está descrita na Figura 31.

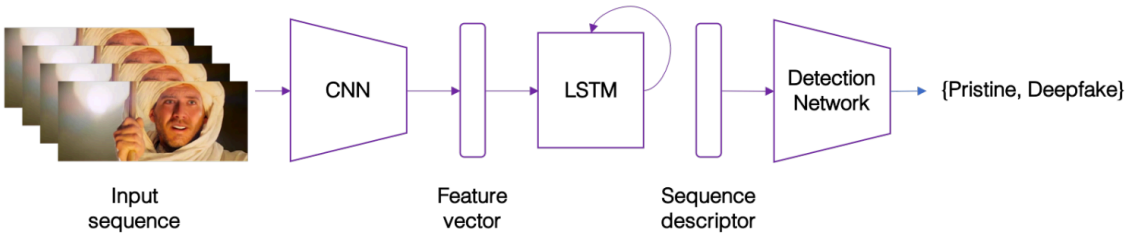


Figura 31: Arquitetura neuronal do trabalho [48].

Segundo os autores, para a camada CNN responsável pela extração das características, foi escolhida a

rede neuronal *InceptionV3*. Desta rede foi utilizado as camadas *CNN* pré-treinado com o *dataset ImageNet* [51] com uma camada totalmente conectada na parte superior da rede [51]. Não houve ajuste a rede, o vetor com dimensão de 2048 características, após as últimas camadas de agrupamento, foram utilizadas como entrada para a camada *LSTM*.

Uma vez recebendo o vector de características, tem como desafio processar recursivamente uma sequência de imagens de maneira significativa. Para este problema, recorreu-se ao uso da camada *LSTM* de 2.048 neurónios seguida de uma camada *Dropout* de 0.5. Consequentemente, passa por uma camada de 512 neurónios totalmente conectada com outra camada *Dropout* de 0.5 . Por fim, utiliza-se de uma camada conectada com a função de activação *softmax* para calcular as probabilidades de a sequência de imagens pertencer a um vídeo real ou *deepfake*.

O *dataset* utilizado possui 600 vídeos, sendo que 50% destes foram colectados, pelos próprios autores, de vários sites de hospedagem de vídeos *deepfake*. Os outros 50% foi retirado do *dataset HOHA* [52]. Foram utilizados 70%, 15%, 15% para treino, validação e teste respetivamente. Cada percentagem foi assegurada que contivesse 50% de vídeos reais e vídeos *deepfake*.

No que se refere ao pré-processamento dos dados foram subtraídos a média de cada canal de cada imagem; redimensionadas para 299 × 299; Para cada vídeo foi escolhido um amostragem de imagens de sub-sequência de quantidade N, esta variável N possuindo três valores distintos de 20, 40, 80 imagens. Isto permitiu que os autores observassem quantas imagens se faz necessário para classificação de um vídeo com resultados positivos e se diferencia os resultados entre os valores que N assumia em cada modelo; O optimizar utilizado foi o Adam para o treino do modelo com uma taxa de aprendizagem de 1e-5 e e decadência de 1e−6.

Após realizar o pré-processamento de dados, realizou-se o treino da arquitetura proposta com os diferentes valores relacionados a quantidade de imagens extraídas da sequência de cada vídeo. Os resultados obtidos demonstraram que não há uma diferença significante entre extrair uma sequência 20 ou 80 imagens do vídeo a ser classificado, como descrito na Tabela 6.

Modelo	Acurácia do Treino (%)	Acurácia do Validação (%)	Acurácia do Teste (%)
Conv-LSTM,20 imagens	99.5	96.9	96.7
Conv-LSTM,40 imagens	99.3	97.1	97.1
Conv-LSTM,80 imagens	99.7	97.2	97.1

Tabela 6: Resultados do trabalho [48].

2.3.4 Detecteçao de deepfake : humanos vs máquinas

No trabalho de Korshunov and Marcel [53] realizou um estudo comparativo da capacidade, de deteção dos seres humanos frente a duas redes neuronais, ao classificarem vídeos *deepfake*. As duas redes neuronais, a *Xception* [54] e *Efficient Net* (variante B4) [55] foram pré-treinadas com dois *datasets* públicos de vídeos *deepfake*: um subconjunto do *Google* da *FaceForensics++* [47] e o recente *dataset Celeb-DF* [43].

Ao realizar o treino das redes neuronais com os respetivos *datasets*, foram gerados 4 modelos de classificação. Estes foram avaliados com o conjunto de dados dos respetivos *datasets* separados para testa-los. O resultado descrito na Tabela 7 do qual é apresentado uma boa performance de deteção ao se um determinado vídeo é *deepfake*.

Modelo	**Treino com**	**Acurácia do Teste (%)**
Xception	*dataset* Google	100.00
Xception	*dataset Celeb-DF*	100.00
EfiientNet	*dataset* Google	99.99
EfiientNet	*dataset Celeb-DF*	100.00

Tabela 7: Valor da área sob a curva (AUC) nos conjuntos de dados para teste dos *datasets* da *Google* e *Celeb-DF* para os modelos criados com a *Xception* e EfficientNet.[53].

O total de voluntários os quais participaram da avaliação subjetiva do estudo foram o total de 60 indivíduos. Estes tinham idades de 20 aos 65 anos. O grupo era composto por estudantes de doutoramento, investigadores e pessoas do quadro da administração. Para realizar o comparativo da classificação subjetiva dos voluntários em relação aos modelos foram pré-selecionados 120 vídeos distintos (60 deepfakes e 60 originais) do *dataset DFDC2019* [56].

Para avaliação subjetiva utilizou-se a ferramenta *QualityCrowd2* [57]. Esta ferramenta permitiu os autores garantirem que os voluntários assistissem a cada vídeo e o classifiquem, não permitindo que saltem para o vídeo posterior.

Para garantir uma equidade na classificação subjetiva dos voluntários em relação a classificação automática realizada pelos algoritmos, os quais foram treinados com as faces presentes de cada imagem extraída dos vídeos. Desta forma, foi utilizada a ferramenta *QualityCrowd2* da qual apresentou aos voluntários os vídeos juntamente com as faces detetadas no vídeo a ser avaliado pelo voluntário, tal como descrito na Figura 32.

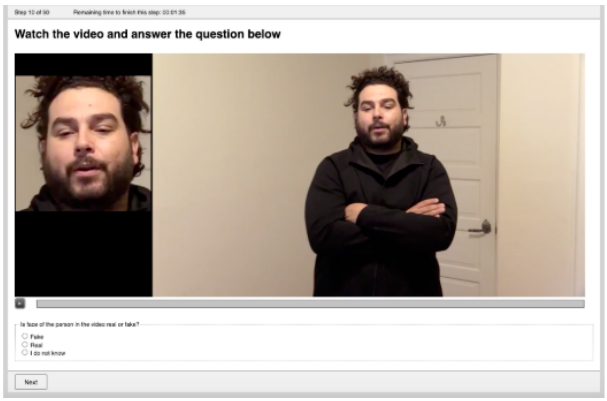


Figura 32: Imagem de uma etapa de avaliação subjetiva [53].

Os voluntários foram instruídos de como utilizarem a ferramenta *QualityCrowd2* para assistirem o vídeo enquanto observavam a face detetada do vídeo ao lado. Posteriormente, os voluntários eram questionados

com a seguinte questão: "A face da pessoa no vídeo em questão é real ou falsa?"possuindo as seguintes respostas: "falso", "real", e "não sei". Os 120 vídeos foram sub-divididos aleatoriamente em lotes, para cada cada lote foram designados 40 vídeos, resultando em media de 16 minutos para avaliação por cada voluntário.

Além da subdivisão dos vídeos, estes também foram categorizados com as seguintes categorias para os vídeos *deepfake*: muito fácil (*very_easy*), fácil (*easy*), moderado (*moderate*), difícil (*difficult*), muito difícil (*very_difficult*) e os vídeos reais a categoria é original (original). Na Figura 33 é apresentado o resultado da classificação subjectiva realizada pelos voluntários do trabalho.

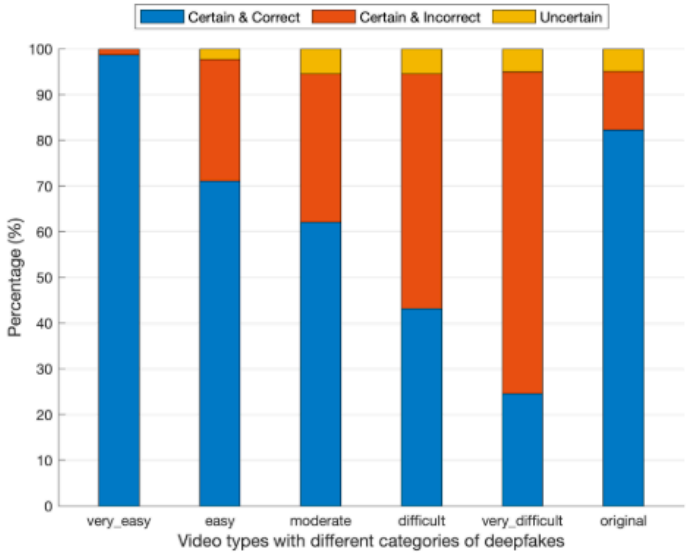
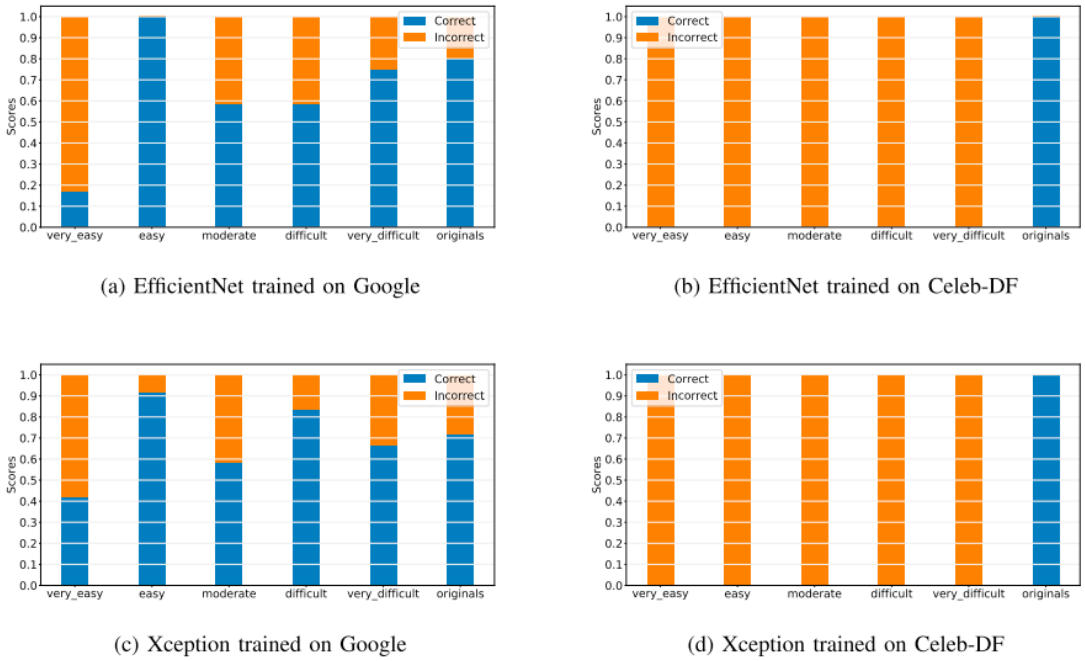


Figura 33: Respostas subjectivas para cada categoria de *deepfake* e reais [53].

De igual forma, os 120 vídeos foram submetidos aos quatros modelos criados a partir das duas arquiteturas neuronais. Os autores deste trabalho consideraram uma faixa de classificação errada de 10% em relação a classificação subjetiva como um valor aceitável para os modelos. Na Figura 34 é apresentado o resultado da classificação dos modelos.



(a) EfficientNet trained on Google

(b) EfficientNet trained on Celeb-DF

(c) Xception trained on Google

(d) Xception trained on Celeb-DF

Figura 34: Classificação dos vídeos pelas categorias dos vídeos *deepfake* e reais [53].

Desta forma, ao comparar a classificação subjetiva dos 60 voluntários em relação aos modelos criados a partir das arquiteturas neuronais, os resultados demonstram que a avaliação subjetiva apresenta uma consistência dos voluntários ao analisarem diferentes categorias de *deepfake*. No entanto, estes apresentaram uma dificuldade superior ao classificar os vídeos *deepfake* da categoria muito difícil em relação as demais, errando aproximadamente em 75,5% dos casos. O oposto disto, foram os modelos dos quais possuem uma perceção totalmente distinta ao classificar os vídeos das categorias *deepfake* (muito fácil, fácil) demonstraram uma dificuldade e uma maior precisão ao classificar as categorias que para os participantes eram mais complexos de classificar.

2.3.5 Deteção de vídeo deepfake através de fluxo óptico baseado em CNN

No trabalho Amerini et al. [58] foi proposto uma método para deteção de vídeo *deepfake*, do qual não analisa as imagens isoladamente para realizar tal predição de uma classe. O método proposto por este trabalho adota uma técnica chamada *Optical Flow*, numa tradução livre fluxo óptico, para explorar possíveis diferenças entre as imagens. O fluxo óptico é o termo para indicar a área de velocidade gerado pelo movimento relativo entre os objetos e a camera em um quadro [59].

A arquitetura neuronal utilizada é representada na Figura 35. A estrutura foi construída para compreender a eficácia real dos áreas de fluxo óptico para distinguir entre um vídeo *deepfake* e real. O fluxo óptico [60] é uma área vectorial da qual é computada em duas imagens consecutivos f(t) e f(t +1) para extrair o movimento aparente entre o observador e a própria cena. A hipótese dos autores é que o fluxo óptico seria capaz de detetar as diferenças no movimento através das imagens sinteticamente criados

pelas técnicas para criação de vídeo *deepfake* em relação aos vídeos reais.

Consequentemente, seria mais percetível nas matrizes de fluxo óptico os movimentos falsos nos lábios, olhos e em toda a face quando o vídeo analisado é *deepfake*. Por esta razão, para cada imagem f(t), num determinado momento t, extrai-se um fluxo *OF(f(t),f(t +1))* - utilizando uma arquitetura neuronal CNN para fluxo óptico intitulado *PWC-Net* [61]. Esta técnica baseia-se no processamento piramidal e na urdidura e na utilização de um volume de custo processado pela própria CNN para estimar o fluxo óptico. Sucessivamente (ver Figura 35), o fluxo é dado como entrada para a arquitetura da rede neuronal pré-treinada com camadas CNN, os autores intitularam a arquitetura proposta de *Flow-CNN*. As redes neuronais CNN utilizadas para este trabalho foram as VGG16[62] e ResNet50[63].

Figura 35: A arquitetura proposta [58].

O *dataset* FaceForensics++ (FF++) [47] contém oito *sub-datasets*, onde dois destes contém vídeos reais *(youtube, actors)* dos quais foram utilizados para criação dos *sub-datasets* de vídeos *deepfake*. Nos dados do *sub-dataset youtube* foram aplicados cinco métodos diferentes para geração *deepfake*, resultando para cada método um *sub-dataset (Deepfakes, Face2face, FaceShifter, FaceSwap, NeuralTextures)*. Enquanto que o *sub-dataset actors* deu origem ao *sub-dataset deepfake DeepFakeDetection*. Posto isto, este trabalho selecionou o *sub-dataset Face2Face* juntamente com o *sub-dataset youtube* para avaliar os resultados, apresentados na Tabela 8, das arquiteturas neuronais propostas. Dos 1.000 vídeos presentes nos *sub-datasets* foram utilizados 720 para o treino, 120 para validação e 120 para teste.

Modelo	**VGG16**	**ResNet50**
Face2Face	81.61%	75.46 %

Tabela 8: Resultado da classificação binária da *Flow-CNN* .

2.3.6 Deteção de deepfake utilizando-se de Deep Learning

No trabalho de Pan et al. [64] realizou uma comparação entre duas arquiteturas neuronais, *Xception* [54] e *Mobile Net* [65], aplicadas ao problema de classificação de vídeos *deepfake*. A *Xception* é uma das arquiteturas mais utilizadas no estado da arte para deteção de *deepfake*, enquanto que a *Mobile Net* é uma arquitetura que exige menos poder computacional em relação a *Xception*. Os dados utilizados para este trabalho foram os 4 *sub-datasets deepfake* (*Deepfake, Face2Face, FaceSwap e NeuralTextures*) contidos no *dataset FaceForensics++* [47].

Foram selecionados os vídeos de 30 FPS, sendo que em cada vídeo foi realizado uma extração de uma imagem a cada quatro imagens. Diferente dos trabalhos encontrados no estado da arte foi utilizado para

detetar as faces presentes nas imagens, o classificador em cascata fornecido pelo *OpenCV*[4] o classificador *haarcascade_frontalface_alt*. Desta forma, foi realizado o treino dos algoritmos neuronais para cada método de manipulação facial. Consequentemente, foi realizado uma comparação da acurácia entre os modelos criados, tal como apresentado na Tabela 9. Para cada treino os *sub-datasets* foram divididos em duas partes 80% para o treino dos modelos e 20% para validação.

Modelo	Xception	Mobile Net
Deepfakes	97.90%	96.50 %
Face2Face	98.27%	96.77 %
FaceSwap	98.35%	98.48 %
NeuralTextures	91.20%	88.49 %

Tabela 9: Comparação da Precisão Total entre *Xception*e MobileNet .

Os autores avaliaram a taxa de verdadeiros positivos em relação a taxa de verdadeiros negativos de cada modelo utilizado no treino e validados dos respetivos *sub-datasets*. Na Tabela 10 são apresentados os resultados obtidos na rede *Xception*, enquanto que, na Tabela 11 são apresentados os resultados obtidos com a *Mobile Net*.

Xception	Verdadeiro Positivo	Verdadeiro Negativo
Deepfakes	97.56%	98.24 %
Face2Face	97.65%	98.90 %
FaceSwap	97.28%	99.18 %
NeuralTextures	91.14%	91.25 %

Tabela 10: Taxa de verdadeiros positivos e negativos [64].

Mobile Net	Verdadeiro Positivo	Verdadeiro Negativo
Deepfakes	94.48%	98.50 %
Face2Face	95.20%	98.36 %
FaceSwap	98.31%	98.62 %
NeuralTextures	91.57%	85.97 %

Tabela 11: Taxa de verdadeiros positivos e negativos [64].

2.4 Análise dos Resultados

Os trabalhos analisados do estado da arte proporcionaram respostas às questões de investigações elaboradas no planeamento da RSL. A questão de investigação sobre ao ano de publicação dos trabalhos, foram verificados que há uma maior número de publicações nos últimos 4 anos (2018 -2021) sobre deteção de *deepfake*. Ao verificar no portal *Dimensions.IA* [5], corroborou com tal conclusão, descrito na Figura 36. A investigação foi realizada no *Dimensions.IA* no dia 01 de Janeiro de 2022.

[4]Link do *OpenCV*: https://opencv.org/

[5]Portal Dimensions: https://app.dimensions.ai

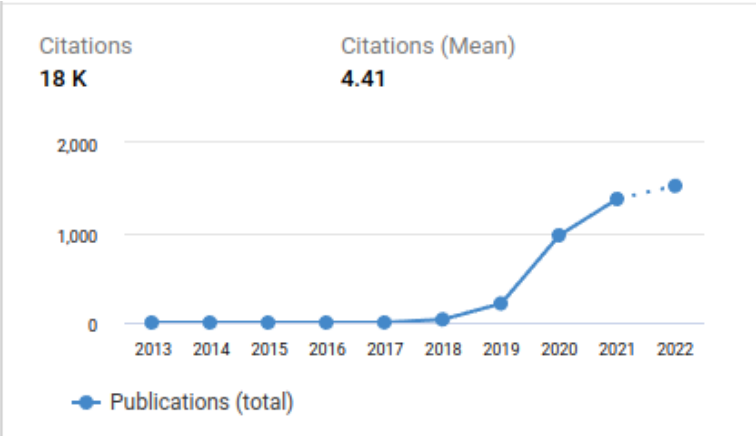


Figura 36: Número de artigos relacionados a deepfakes últimos anos. [66].

Consequentemente, em relação ao tipo de de conteúdo multimédia analisado pelos detetores de conteúdo de *deepfake*, mais de 50% dos trabalhos analisados tem como objeto de estudo o conteúdo multimédia no formato de vídeo. Sobre qual algoritmo e ou técnica utilizados, no estado da arte, para deteção de conteúdo multimédia *deepfake*? As redes neuronais, que utilizem camadas do tipo CNN, são as mais utilizadas a exemplo da *Xception*. No entanto, os trabalhos que obtiveram melhores resultados são aqueles que se concentram e aplicaram métodos e técnicas no pré-processamento dos dados. Por fim, quais os *datasets* utilizados para classificação de *deepfake*. Existem os que são criados pelos próprios autores dos trabalhos e a utilização de *datasets* públicos, sendo seis *datasets* públicos de maior popularidade. Destes o *standard* para o estado da arte é o *dataset* FaceForensics++ (FF++).

Capítulo 3

Desenvolvimento

Para execução deste trabalho utilizou de um ambiente de desenvolvimento, bem como a utilização de dados oriundos de *dataset open source*, para desta forma, conseguir criar um modelo de classificação, o qual proporcionem ao detetor *deepfake* a capacidade de detetar se um determinado vídeo é produto de uma técnica de criação *deepfake*.

3.1 Ambiente de Desenvolvimento

Foi utilizado a *stack* de ferramentas disponibilizadas na *Cloud* pela *Google*, nomeadamente o *Google Colaboratory* e o *Google Drive*. O *Google Colaboratory* ou sua abreviatura *Colab*, é um produto do *Google Research*, uma plataforma a qual proporciona poder computacional para executar algoritmos de Machine Learning (ML) ou DLnum grande conjunto de dados. O *Colab* proporciona um ambiente com o *Jupyter notebook, online,* de forma gratuita. Este ambiente é baseado em nuvem o que possibilita treinar os modelos utilizando CPUs, GPUs e TPUs fornecidas pela própria plataforma.

A *Graphics Processing Unit (GPU)* disponível na versão gratuita do *Colab* é uma Tesla K80 com 12GB de memoria GDRR5 com um processador Intel Xeon com 2 cores de 2.2Ghz e 13 GB de memória RAM. Existe a limitação da *GPU* ficar disponível durante 12 horas para execução do seu algoritmo, no entanto, há possibilidade desse recurso desligado a depender da demanda de utilização à plataforma*Colab*. Devido a ausência de poder computacional para execução desse trabalho foi utilizado esta a plataforma, pois havia necessidade de executar algoritmos de DL utilizando-se de grande conjunto de dados.

No entanto, os recursos oferecido pela versão gratuita não foram suficiente no decorrer da investigação desse trabalho. Como consequência disso foi necessário a aquisição de uma assinatura dessa plataforma, o *Colab Pro* que oferece uma Tesla T4 com 27GB de memoria GDRR5 com um Intel (R) Xeon (R) a 2,30 GHz e 13 GB de memória RAM. Com este serviço seus algoritmos podem permanecer executando até 24 horas, e os tempos de inatividade são relativamente brandos. Contudo, as durações não são garantidas e os tempos limite de inatividade podem variar.

O Google Drive é um serviço de armazenamento em nuvem, o qual pode ser utilizado como forma de backup de ficheiros, assim liberando espaço de memoria da maquina local. Ao criar uma conta do *Gmail*,

o provedor de email da Google, o utilizador consequentemente é contemplado com um Google Drive com espaço total de 15 GB a sua disposição. Devido ao alto volume de dados resultantes do pré-processamento dos vídeos, serem superiores a 20 GB, foi necessário utilização do *Google Drive* na modalidade estudantil o qual possui uma capacidade de armazenamento ilimitada.

3.2 Datasets

Os *datasets* de vídeos *deepfake* começaram a surgir a partir do ano 2018, baseando-se na utilização de *GANs* aplicadas em diversas direções de investigação. Desta forma, surgiram vários *datasets* de vídeos *deepfake* dos quais se utilizam ferramentas baseados em algoritmos de *DL* e ferramentas de visão computacional para criação de conteúdo multimédia *deepfake* [67]. Há seis *datasets* de maior destaques na literatura:

- **UADFV :** criado em 2018, composto por 49 vídeos reais adquiridos do *YouTube* e 49 vídeos *deepfake* . Os vídeos *deepfake* foram gerados utilizando o modelo *DNN* com *software* FakeAPP;
- **Deepfake-TIMIT:** foi o primeiro *dataset* a sintetizar vídeos usando *faceswap-GAN* para gerar 640 vídeos *deepfake* [68]. Os vídeos foram divididos em baixa e alta qualidade, dependendo da resolução das imagens;
- **FaceForensics++ (FF++):** primeiro *dataset* de grande proporção em quantidade de vídeos *deepfake*, criado em 2019, contendo 1.000 vídeos reais extraídos do conjunto de dados do *Youtube-8M*[1]. Estes foram manipuladas com quatro métodos automatizados de manipulação facial: *Deepfakes* utilizado uma adaptação do trabalho [69], *Face2Face* [70], *FaceSwap* [47] e *NeuralTextures* [71, 72], sendo que dois dos métodos são baseados em computação gráfica e os outros dois se utilizam de algoritmos de *DL*. Consequentemente, gerou-se 1.000 vídeos *deepfake* para cada método, totalizando 5.000 vídeos [47] ao total para o *dataset FF++*;
- **DFD:** este *dataset* foi criado a partir da cooperação entre a Google e Jigsaw, criado em 2019. A Jigsaw é uma unidade do grupo Alphabet, a qual investiga os tipos de ameaças as sociedades abertas, e constroem tecnologias que almejam soluções escaláveis para tais. Dessa colaboração resultou um *dataset* de vídeos *deepfake* [73], o qual é composto por 3.068 vídeos *deepfake* gerados a partir de 363 vídeos originais produzidos por 28 atores de vários géneros, idades e grupos étnicos. Não é informado especificamente a técnica ou algoritmo utilizado para gerar os vídeos *deepfake*, no entanto, é informado que foram produzidos por modelos *deep generative* que podem manipular vídeo com áudio. Posteriormente, este *dataset* foi agregado ao *dataset FF++*;
- **DFDC:** o *dataset* é referente do desafio de deteção de vídeos *deepfake* publicado, pelo *Facebook* em 2019 [56], no formato de competição na plataforma *Kaggle*[2]. O *dataset* foi construído com o

[1] Site do Youtube-8M:https://research.google.com/youtube8m/

[2] Link para competição no kaggle: https://www.kaggle.com/c/deepfake-detection-challenge

auxílio de 66 atores, com uma diversidade em vários características (sexo, tom de pele, idade, etc..). Os atores gravaram 1.131 vídeos com plano de fundo arbitrário, resultando numa variabilidade visual. Dos vídeos originais, foram produzidos 4.113 vídeos *deepfake*. Posteriormente, em 2020, lançaram uma nova versão com mais de 100.000 vídeos [74].

- **Celeb-DF:** este *dataset* foi criado em 2019 com 408 vídeos originais recolhidos do *YouTube* com uma diversidade atendendo ao plano de fundo, idade, grupo étnico e género dos atores. Dos vídeos originais, foram criados 795 vídeos *deepfake*. Contudo, em 2020, lançou-se uma nova versão com 590 vídeos originais extraídos do *YouTube* com a mesma diversidade da primeira versão. Dos vídeos originais, criou-se 5.639 vídeos *deepfake* [75].

Para este trabalho foi selecionado o *dataset FF++*, pelas seguintes razões: (1) encontra-se disponível publicamente e possui uma documentação detalhada sobre os dados; (2) a maioria dos vídeos contem gravação de apenas um individuo no campo de visão frontal, isso possibilita uma extração da face sem obstrução, da qual pode ser facilmente rastreada; (3) o *dataset* possui uma variedade de métodos populares de geração de vídeo *deepfake*, informando quais dos vídeos foram gerados a partir de uma determinada técnica; (4) o *dataset FF++* oferece diferentes opções no que se refere à qualidade do vídeo dos dados para realização do *download* dos mesmos. Este critério foi de extrema relevância considerando os recursos computacionais disponíveis para este trabalho serem inferiores quando comparado com os recursos computacionais apresentados no estado da arte.

3.2.1 Versões do dataset FF++

O *FF++* possui *sub-datasets*, os quais correspondem a diferentes métodos de geração de vídeos *deepfake* e os *sub-datasets* com os vídeos originais, sendo que até 2021 totaliza-se seis métodos de geração *deepfake*. Os dados *deepfake* deste *dataset* foram gerados a partir de dois *sub-datasets* de vídeos originais, totalizando em 8 *sub-datasets*. Para fins didáticos, este trabalho considerou que cada inserção de *sub-dataset* é o equivalente a uma nova versão do *dataset FF++*.

- **1º Versão do FF++:** composta por um *sub-dataset* de vídeos originais (*YouTube*), o qual foi extraído 1.000 vídeos do *dataset Youtube-8M*. Consequentemente, foram aplicados 4 técnicas *deepfake* para criação de 4 *sub-datasets* de vídeos *deepfake*, sendo dois oriundos de abordagens baseadas em computação gráfica (*Face2Face* [70] e *FaceSwap*[47]) e outros aplicados com *DL* (*DeepFakes* [69] e *NeuralTextures*[71, 72]). Todos os quatro métodos requerem pares de vídeo de atores de origem e alvo como entrada. Os dados extraídos de cada método é um vídeo composto de imagens geradas. Esta versão do *FF++* totaliza 5.000 vídeos, sendo que 80% desses são *deepfake*;

- **2º Versão do FF++:** esta versão é referente à inserção dos vídeos do *dataset DFD* ao *FF++*. Esta nova versão do *FF++* adicionou um *sub-dataset* com 363 vídeos originais de atores bem como um *sub-dataset* com 3.000 vídeos *deepfake*, gerados a partir dos vídeos originais, totalizando 8.363 vídeos no *FF++* sendo aproximadamente 84% de vídeos *deepfake*;

- **3º Versão do FF++:** esta versão corresponde à inserção de mais um *sub-dataset* com 1.000 vídeos *deepfake*, criado com o algoritmo de troca facial, o *FaceShifter* [76]. Ele é capaz de gerar uma identidade de alta fidelidade, preservando os resultados da troca facial, em comparação com os métodos anteriores. Para criação dos vídeos do *sub-dataset FaceShifter* foram utilizados os 1.000 vídeos do *sub-dataset youtube*. Desta forma, o *dataset FF++* detém 9.363 vídeos, onde somente 15% dos vídeos são de originais, não manipulados.

Desta forma, o *dataset FaceForensics++ (FF++)* possui um total 8 *sub-datasets*. Dois destes são referentes à vídeos reais, presentes na Tabela 12, dos quais foram matéria prima para geração dos outros 6 *sub-datasets* de vídeos *deepfake*, tal como descritos na Tabela 13.

Nome do sub-dataset	Quantidade de vídeos	Origem
youtube	1.000	*dataset* Youtube-8M
actors	363	confeccionado pela Google

Tabela 12: *Sub-datasets* de vídeos reais do *dataset* FaceForensics++ (FF++)

Nome do sub-dataset	Quantidade de vídeos	Vídeos Originais	Método deepfake
Face2Face	1.000	*sub-dataset youtube*	Computação Gráfica
FaceSwap	1.000	*sub-dataset youtube*	Computação Gráfica
DeepFakes	1.000	*sub-dataset youtube*	Deep Learning (DL)
NeuralTextures	1.000	*sub-dataset youtube*	Deep Learning (DL)
FaceShifter	1.000	*sub-dataset youtube*	Deep Learning (DL)
DeepFakeDetection	3.000	*sub-dataset actors*	Deep Learning (DL)

Tabela 13: *Sub-datasets* de vídeos *deepfake* do *dataset* FaceForensics++ (FF++)

No desenvolvimento deste trabalho foram utilizados diferentes agrupamentos dos *sub-datasets* do *FF++* . Esse *dataset* tem os vídeos em compreensão H.264, com fatores de compressão de 0(bruto), 23(médio) e 40(alto). O H.264 ou MPEG-4 AVC *(Advanced Video Coding)* é um formato de codificação de vídeo para gravação e distribuição de áudio e vídeo *full HD*. Foi desenvolvido e mantido pelo *ITU-T Video Coding Experts Group (VCEG)* com o ISO / IEC JTC1 *Moving Picture Experts Group (MPEG)*.

3.2.2 Aquisição dos dados

Para realizar a aquisição do *dataset* com os seus respetivos *sub-datasets*, foi necessário o preenchimento de um formulário[3] no *Google Form*. Este formulário, para além de requerer as informações dos investigadores, é solicitado uma justificação do motivo para aceder ao *dataset FF++*. Após a submissão da solicitação e analise, são enviados instruções para o requerente realizar o *download* do *dataset* armazenados nos servidores da Universidade Técnica de Munique.

Essas instruções contém um *script* em *Python* para realizar o *download*. É possível personalizar qual *sub-dataset*, quantidade de vídeos e o tipo de compressão dos vídeos para realizar o *download*.

[3]Formulário para aceder ao *dataset* FF++: www.encurtador.com.br/ioOQ3

Neste trabalho foi realizado o *download* de todos *sub-datasets* do *FF++* com a taxa compressão de 23 (media). Foram realizados *download* da totalidade de todos os dados de cada *sub-dataset*, com exceção do *subdataset DeepFakeDetection*, do qual foi realizado *download* de 1.000 vídeos, 1/3 da sua totalidade, tal como demonstrado na Tabela 14.

Nome do sub-dataset	**N. de Vídeos**	**Download**	**Tipo do vídeo**
youtube	1.000	1.000	real
actors	363	1.000	real
Face2Face	1.000	1.000	*deepfake*
FaceSwap	1.000	1.000	*deepfake*
DeepFakes	1.000	1.000	*deepfake*
NeuralTextures	1.000	1.000	*deepfake*
FaceShifter	1.000	1.000	*deepfake*
DeepFakeDetection	3.000	1.000	*deepfake*

Tabela 14: *Download* dos *sub-datasets* e suas respectivas quantidade de vídeos

Os dados de cada *sub-dataset* foram armazenados na nuvem, especificamente no *Google Drive* do autor deste projeto [4], para dessa forma realizar as etapas seguintes.

3.2.3 Análise dos dados

Após o *download* do *dataset FF++*, realizou-se uma análise dos dados presentes em cada *sub-dataset*. Esta análise serviu para averiguar o tempo de duração e número de Frames Por Segundos (FPS) de cada vídeo. Para realizar a análise dos cenários de gravação dos vídeos por *sub-dataset*, utilizou-se dois *sub-datasets* com vídeos originais, respectivamente *youtube* e *actors*.

A partir da distribuição da duração de cada vídeo presente nos *sub-datasets*, é constada uma forte similaridade na distribuição temporal dos dados presentes no *sub-dataset youtube* com os 5 *sub-datasets deepfake (Deepfakes, Face2Face, FaceShifter, FaceSwap, NeuralTextures)*. Enquanto a distribuição temporal do *sub-dataset DeepFakeDetection* é extremamente oposta, este apresenta uma similaridade com o *sub-dataset actors*, como descrito na Figura 37.

[4] Link para acesso datasets: www.encurtador.com.br/oHJLY

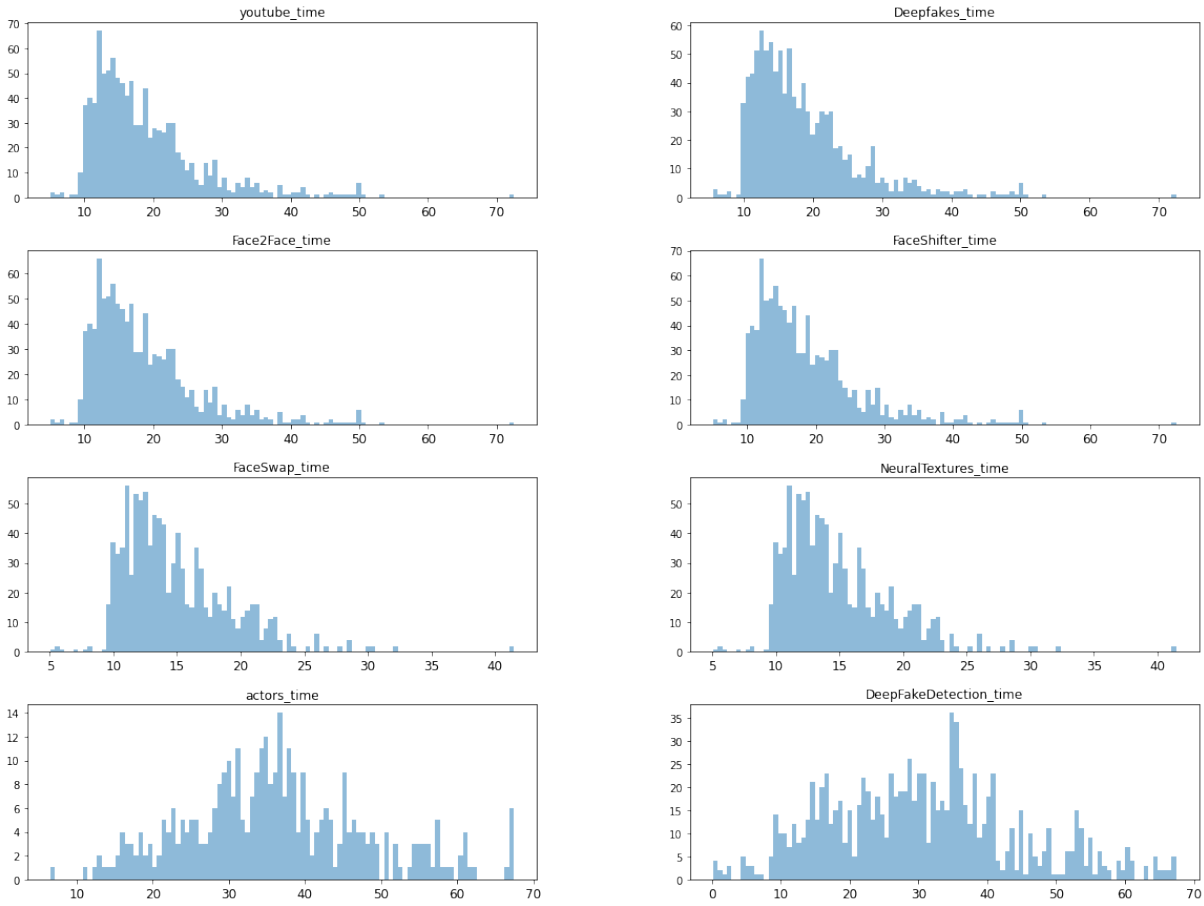


Figura 37: Histogramas da quantidade de videos por tempo de duração em segundos em cada *sub-dataset*.

O número de FPS presente nos vídeos do *dataset FF++*, apresenta 8 configurações FPS (15,18,24,25,29,30,50,60). O número de vídeos de cada *sub-dataset* que possui uma determinada configuração de FPS é descrita na Tabela 15. Esta demonstra que somente dois *sub-datasets* possuem dados com um único número de FPS.

Sub-dataset/FPS	**15 FPS**	**18 FPS**	**24 FPS**	**25 FPS**	**29 FPS**	**30 FPS**	**50 FPS**	**60 FPS**
youtube	4	1	25	478	2	480	2	7
Face2Face	4	1	25	478	2	480	2	7
FaceSwap	4	1	25	478	2	480	2	7
DeepFakes	4	1	25	478	2	480	2	7
NeuralTextures	4	1	25	478	2	480	2	7
FaceShifter	4	1	25	478	2	480	2	7
actors	-	-	363	-	-	-	-	-
DeepFakeDetection	-	-	1.000	-	-	-	-	-

Tabela 15: Quantidade de vídeos por Frames Por Segundos (FPS) de cada *sub-dataset*.

A caracterização dos dados contidos nos *sub-datasets* podem ser observados a partir dos dois *sub-dataset* com vídeos originais, uma vez que os dados falsos dos *sub-datasets deepfake* são derivados

destes. O *sub-dataset youtube* contém um plano de gravação frontal e maioritariamente os vídeos contém um único indivíduo, com ambiente estatístico como apresentado na Figura 38. Enquanto, que no *sub-dataset actors* contém vídeos com mais de um individuo em diferentes perspetivas em ambientes dinâmicos como demonstrado na Figura 39.

Figura 38: Vídeo do *sub-dataset youtube*

Figura 39: Vídeo do *sub-dataset actors*

3.3 Métricas de Avaliação

Para avaliação do modelo de cada experiência foi considerado a acurácia de validação. Esta métrica é a proporção dos resultados verdadeiros, positivo ou negativo, pelo total dos casos analisados pelo modelo. Foi atribuído o valor 0 para os casos de não manipulação, ou seja imagens de vídeos reais. Para as imagens manipulados foi atribuído o valor 1. Atendendo ao uso da matriz de confusão, foi possível analisar a precisão, *recall* e a métrica F1 que mensura a sensibilidade do modelo e verificar a ocorrência dos dois tipos de erros para classificação observáveis pela matriz de confusão.

		Valores preditos	
		Negativo (0)	**Positivo(1)**
Valores reais	**Negativo (0)**	Verdadeiro Negativo (VN)	Falso Negativo (FN)
	Positivo (1)	Falso Positivo (FP)	Verdadeiro Positivo (VP)

Tabela 16: Matriz de Confusão

Erro do tipo 1 é quando uma imagem *fake* ao ser analisada pelo modelo foi classificada como real. Este erro representa o quadrante da matriz de confusão FP. O erro do tipo 2, no entanto, é o oposto quando uma imagem real é classificada pelo modelo como *fake*. Este erro é representado no quadrante da matriz de confusão FN. Esse trabalho buscou um equilíbrio entre estes dois valores, dando preferência a mitigar o erro do tipo 2.

Consequentemente, é considerado a *recall* de cada classe, pois esta avalia o desempenho do modelo ao classificar os elementos de uma determinada classe no conjunto total de dados dessa classe, dentre as imagens *fake*, quantas de fato foram classificadas como sendo de vídeos manipulados.

Outra métrica relevante é a precisão do modelo, capaz de responder à seguinte questão: qual a proporção de positivos previstos é verdadeiramente positiva? Por fim, a métrica F1 tem como objetivo

medir o média harmónica entre o *recall* e a precisão de uma determinada classe. Dito isso, os critérios de seleção do modelo para esse trabalho é avaliado com as métricas acurácia de validação e os modelos com melhores resultados nas métricas *recall* e F1.

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

$$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$

$$Recall = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

$$F_1 = 2 \times \frac{\text{Precisão} \times \textit{Recall}}{\textit{Precisão} + \textit{Recall}}$$

3.4 Classificação a partir de imagens

A primeira experiência foi realizada como um estudo preliminar, onde foram selecionados 168 vídeos, metade destes é composta por vídeos reais retirados do *sub-dataset youtube* e os restantes compostos por vídeos dos 6 *sub-datasets* de vídeos *deepfake* (*Face2Face, FaceSwap, DeepFakes, NeuralTextures, FaceShifter, DeepFakeDetection*). Posteriormente foi realizado um pré-processamento nos dados, do qual foi extraído as 20 primeiras imagens de cada vídeo.

No entanto, a extração das imagens foi realizada usando duas abordagens distintas. Na primeira abordagem foi extraído a totalidade da imagem, tal como demonstrado na Figura 40. A segunda abordagem de extração considerou a extração da face detetada em cada imagem, tal como apresentado na Figura 41. Para esta extração, foi utilizado uma biblioteca desenvolvida em *Python* designada *MTCNN* para reconhecimento de faces implementada a partir do trabalho [77].

Figura 40: Imagem 19 do vídeo.

Figura 41: Face contida na imagem 19.

Dos 168 vídeos, foram selecionados 75% para o treino, 14% para validação e 11% para teste. O total de imagens extraídas dos vídeos encontra-se descrita na Tabela 17.

Quantidade de imagens extraídos	Imagens reais	Imagens deepfake	Total de imagens
Para Treino	1200	1320	2520
Para Validação	260	220	480
Para Teste	220	120	360
Total dos 168 vídeos	**1680**	**1660**	**3340**

Tabela 17: Quantidade de imagens extraídos por tipo de vídeo.

Foi utilizado para criação do modelo aprendizagem por transferência. Desta forma, utilizou-se a rede neuronal *Xception* [54], na qual tem como base de extração de características 36 camadas CNN, que estão estruturadas em 14 módulos, com os pesos da *ImageNet* [51]. Foram utilizadas as camadas CNN em seguida de uma camada *GlobalMaxPool2D* e consequentemente 4 camadas densas com 1024,512,128,64 neurónios respetivamente com a função de ativação *Relu*. Entre essas camadas foram aplicados um *Dropout* de 0.5. A última camada densa apresenta um único neurónio, no qual é aplicado a função de ativação sigmoide, por se tratar de uma classificação binária.

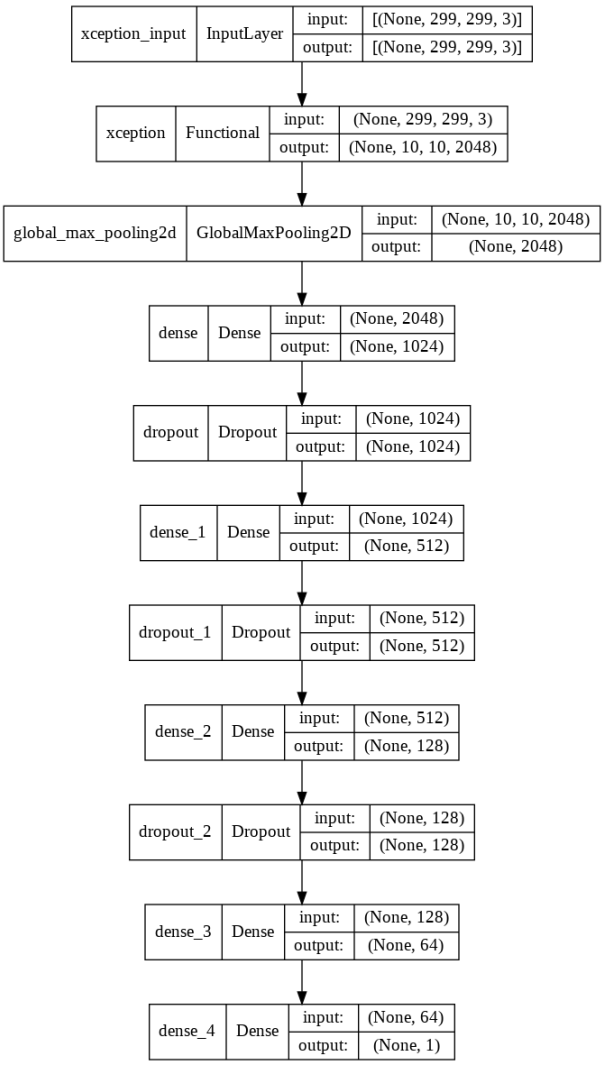


Figura 42: Arquitetura neuronal da primeira experiência.

Após realizar o treino da rede com os dois conjunto de imagens extraídas dos vídeos, isto resultou em dois modelos distintos. Quando o modelo foi treinado com imagens extraídas em sua totalidade dos vídeos, apresentou uma acurácia de validação de 41.47%, enquanto que com as faces detetadas obteve-se uma acurácia de 46.09%. Os resultaram apresentaram um *bias* para classificação de casos reais. Um fator a considerar é a hipótese que a rede neuronal teve acesso a um volume pequeno de dados durante o treino, este não sendo suficiente para proporcionar a rede a capacidade distinguir características dos dados entre as classes real ou *fake*. Desta forma, foi necessário um aumento considerável dos dados em experiências futuras.

Posteriormente, foi realizado outra experiência com 200 vídeos, sendo 50% retirados do *sub-dataset youtube* e a outra metade resultante da junção dos 5 *sub-datasets deepfake (Face2Face, FaceSwap, DeepFakes, NeuralTextures, FaceShifter)*. No entanto, foram extraídas as 80 primeiras imagens de cada vídeo, e de cada imagem extraída a face detetada, totalizando 16.000 imagens, tal como descrito na Tabela 18. Foi mantida a estrutura da rede neuronal elaborada na primeira experiência, apresentada na Figura 42.

Quantidade de imagens extraídos	**Imagens reais**	**Imagens deepfake**	**Total de imagens**
Para Treino	5600	6000	11600
Para Validação	1120	800	1920
Para Teste	1280	1200	2480
Total dos 200 vídeos	**8.000**	**8.000**	**16.000**

Tabela 18: Quantidade de imagens extraídas dos 200 vídeos.

Contudo, mesmo com aproximadamente cinco vezes mais dados do que a experiência anterior o modelo obteve uma acurácia de 49.19%. Este valor, mesmo sendo ligeiramente melhor do que o resultado dos modelos anteriores, não demonstra melhorias na capacidade de distinguir as classes real e *fake*, pois há um forte *bias* na classe real. Desta forma, ficou evidente que a rede neuronal não estava somente a precisar de um aumento no volume de dados de treino. Outra hipótese foi elaborado: a diversidade dos dados *fake* oriundos dos *sub-datasets deepfake*, os quais utilizam diferentes métodos para geração de vídeo *deepfake*, impossibilita o modelo de distinguir uma imagem *fake* de uma real. Posto isto, foi elaborado uma terceira experiência.

3.4.1 Classificação por técnica deepfake

Para essa experiência foi planeado utilizar somente a 1º versão do *dataset FF++*, tal como descrito na secção 3.2.1, onde o *dataset* é composto por um *sub-dataset* de vídeos originais (*sub-dataset youtube*) e quatro *sub-datasets deepfake (Face2Face, FaceSwap, DeepFakes, NeuralTextures)* criados a partir do *sub-dataset youtube*. Consequentemente, desenvolveu-se um modelo para cada método de geração de *deepfake*.

Foram escolhidos os vídeos com 30 FPS, o que, de acordo com a Tabela 15, eram 480 vídeos de cada *sub-dataset* de dados originais (*youtube*) ou de dados *deepfake (Face2Face, FaceSwap, DeepFakes,*

NeuralTextures). Para extração das imagens foi adotada uma nova abordagem, onde a cada 4 imagens num vídeo é selecionado uma e desta é extraída a face do ator.

Apesar das soluções do SOTA evidenciarem o uso maioritariamente da biblioteca *MTCNN* para deteção e extração de faces nas imagens, optou-se pelo uso do classificador em cascata fornecido pelo *OpenCV* [78], o classificador *haarcascade_frontalface_alt*. Esta biblioteca foi escolhida por exigir um poder computacional inferior à biblioteca *MTCNN*. O pipeline do pré-processamento de cada vídeo é descrito na Figura 43.

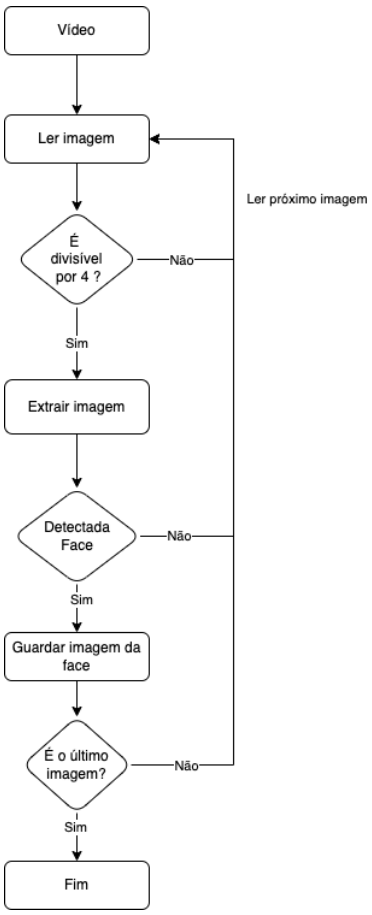


Figura 43: Pipeline do pré-processamento dos dados.

Ao pré-processar os vídeos, as imagens das faces extraídas foram guardadas no seu respetivo diretório a partir dos meta-dados ao fazer a junção de um *sub-dataset deepfake* com o *sub-dataset youtube*, resultando em 960 vídeos para cada método de geração *deepfake*. Foi realizado uma divisão dos vídeos em 80% destinado ao treino e 20% para avaliar o desempenho do modelo de classificação. Após o processamento dos dados para cada método de geração de *deepfake*, foi obtido a quantidade de imagens apresentada na Tabela 19.

Quantidade imagens	Face2Face	FaceSwap	DeepFakes	NeuralTextures
Treino - Real	46.905	46.905	46.905	46.905
Treino - Fake	46.947	37.267	46.709	37.621
Validação - Real	13.070	13.070	13.070	13.070
Validação - Fake	13.088	9.988	12.586	10.112

Tabela 19: Quantidade de imagens guardada por método de geração *deepfake*.

Foram utilizadas arquiteturas de rede neuronal *Xception* [54] e *Mobile Net* [65], esta última sendo uma arquitetura neuronal eficiente e adequada para aplicações móveis, com reduzido uso de poder computacional. Adicionalmente, foram gerados modelos de classificação para cada método de geração dos vídeos *deepfake*.

Para treinar o modelo, foi aplicado o optimizador Adam, pois é um optimizador eficiente e requer pouca memória, sendo adequado para problemas complexos em termos de dados e/ou parâmetros [79]. Adicionalmente foi utilizado o mecanismo de parada no treino, intitulado *EarlyStopping*. O pipeline do desenvolvimento de cada modelo de classificação é descrito na Figura 44.

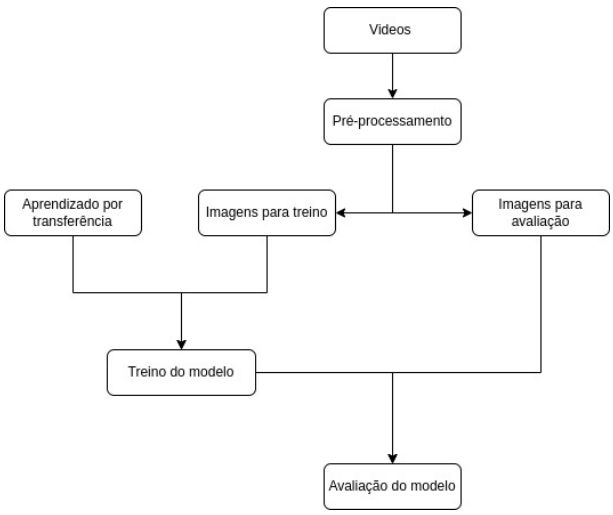


Figura 44: Fluxograma do processo geral de treino e avaliação do modelo de CNN desenvolvido.

Ao realizar o treino das duas arquiteturas de rede neuronal com os respetivos dados de cada método *deepfake*, resultou na criação de oito modelos, dos quais foram avaliados com os respetivos dados de validação para cada técnica de criação *deepfake*, obtendo os valores de acurácia de validação, tal como apresentado na Tabela 20. A *Xception* apresentou valores ligeiramente superiores em relação a *Mobile Net* em três dos quatros métodos *deepfake*.

	Arquitetura	Face2Face	FaceSwap	DeepFakes	NeuralTextures
Validação	*Xception*	97.37%	98.27%	98.17%	91.04%
	Mobile Net	97.42%	97.48%	94.56%	90.96%
Treino	*Xception*	99.43%	99.35%	99.61%	99.36%
	Mobile Net	99.39%	99.24%	99.55%	99.06%

Tabela 20: Acurácia de validação e treino para cada *sub-dataset* nas respetivas redes neuronais.

Contudo, somente a métrica de acurácia de validação não justifica à conclusão de que a rede neuronal *Xception*, é a arquitetura com resultados superiores absolutos para problema de classificação *deepfake*. Posto isto, foi necessário analisar outras métricas destacadas à priori na secção 3.3. Destas, foram avaliados os valores obtidos em cada rede neuronal nos respetivos *sub-datasets deepfake*, possibilitando apresentar uma conclusão mais precisa de qual rede neuronal é mais apropriada para classificação de cada técnica *deepfake*.

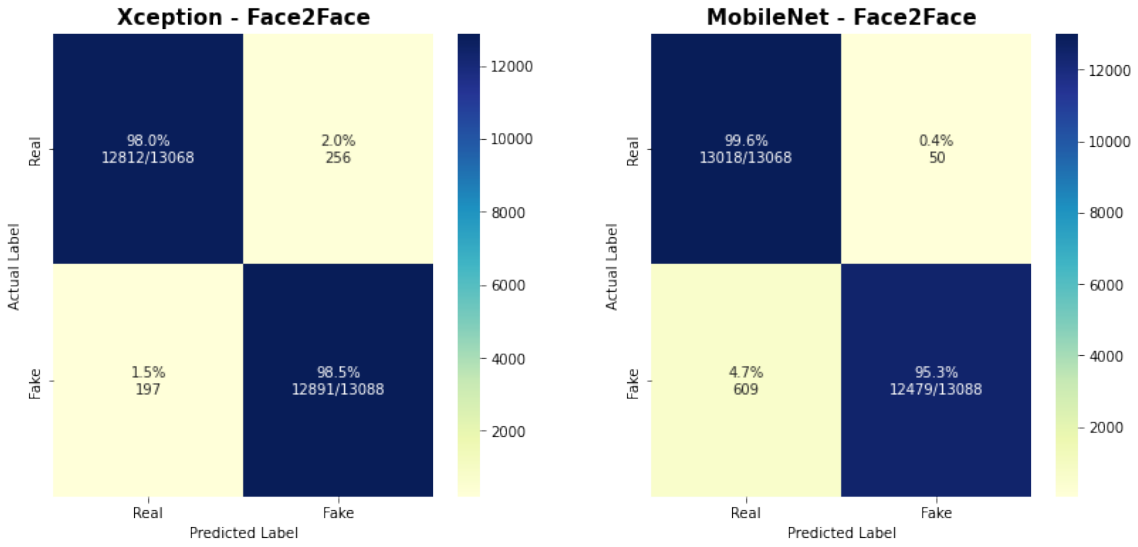


Figura 45: Matrizes de confusão das arquiteturas no sub-dataset *Face2Face*.

As matrizes de confusão das arquiteturas *Xception* e *Mobile Net* exibem os resultados obtidos pelos modelos ao classificarem os dados do *sub-dataset Fac2Face*, tal como apresentado na Figura 45. Há uma maior eficiência da *Mobile Net* ao identificar imagens da classe real. Na Tabela 21 apresenta os valores das métricas dos quais verifica-se que há um maior equilibro dos valores entre as classes (real e *fake*) obtidos pela arquitetura *Xception*.

Face2Face	Xception	Mobile Net
Acurácia	97.37%	97.42%
Erro do tipo 1	1.5%	4.7%
Erro do tipo 2	2.0%.	0.4%
Recall - classe real	0.98	1.00
Recall - classe *fake*	0.98	0.95
F1-Score - classe real	0.98	0.98
F1-Score - classe *fake*	0.98	0.97

Tabela 21: Valores das métricas de avaliação do modelos para o técnica *Face2Face.*

Ao analisar a matriz de confusão do algoritmo treinado com os dados do *sub-dataset FaceSwap* (Figura 46), esta demonstra que a *Xception* possui uma performance ligeiramente superior a rede *Mobile Net*. Na Tabela 22, aonde são apresentados os resultados de performance obtidos pelas redes neuronais, é verificado que a *Mobile Net* possui um *F1-Score* e *recall* da classe real com 0.02 e 0.22 a mais respetivamente quando comparado com a classe *fake* da mesma rede.

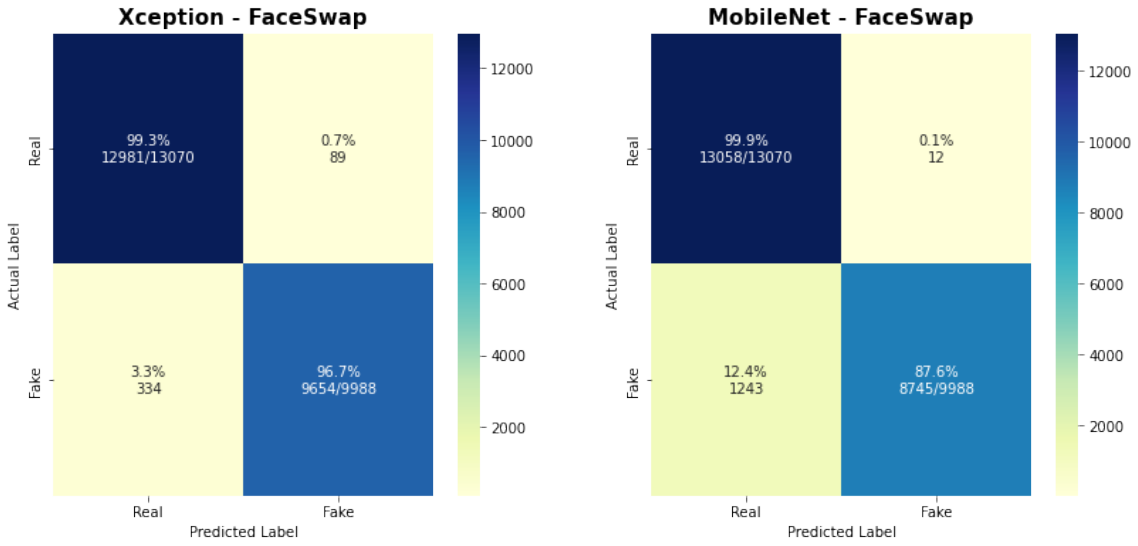


Figura 46: Matrizes de confusão das arquiteturas no *sub-dataset FaceSwap.*

FaceSwap	Xception	Mobile Net
Acurácia	98.27%	97.48%
Erro do tipo 1	3.3%	12.4%
Erro do tipo 2	0.7%.	0.1%
Recall - classe real	0.99	1.00
Recall - classe *fake*	0.97	0.88
F1-Score - classe real	0.98	0.95
F1-Score - classe *fake*	0.98	0.93

Tabela 22: Valores das métricas de avaliação do modelos para o técnica *FaceSwap.*

Ao analisar o modelo com treino dos dados do *sub-dataset Deefakes* pelas redes neuronais obteve-se as matrizes de confusão apresentadas na Figura 47. Estas demonstram os valores de VP e VN ligeiramente semelhantes, no entanto, em quadrantes inverso a *Xception* possui no VP com 98.4% e no VN com 96.3%, enquanto que a *Mobile Net* possui VP com 96.5% e VN com 98.3%. O mesmo acontece com os tipos de erros. Contudo, a métrica *recall* da classe *fake* é superior a da classe real e o erro do tipo 2 é maior que o erro do tipo 1 na rede *Xception* bem como a acurácia, tal como demonstrado na Tabela 23.

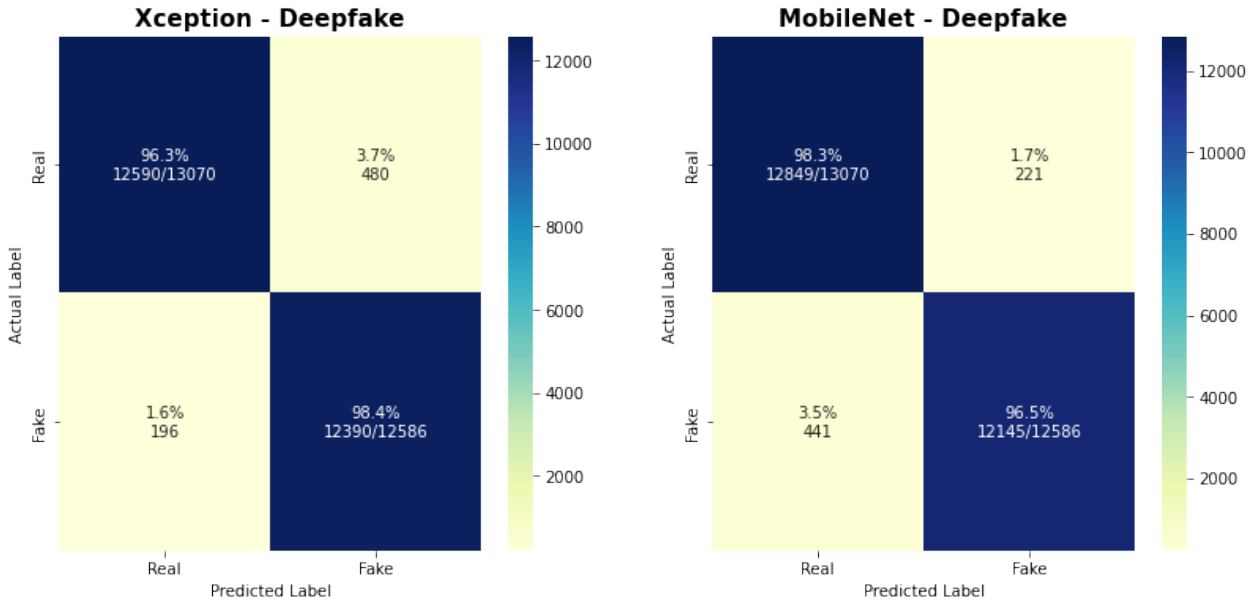


Figura 47: Matrizes de confusão das arquiteturas no *sub-dataset Deefakes*.

Deepfakes	**Xception**	**Mobile Net**
Acurácia	98.17%	94.56%
Erro do tipo 1	1.6%	3.5%
Erro do tipo 2	3.7%.	1.7%
Recall - classe real	0.96	0.98
Recall - classe *fake*	0.98	0.96
F1-Score - classe real	0.97	0.97
F1-Score - classe *fake*	0.97	0.97

Tabela 23: Valores das métricas de avaliação do modelos para o técnica *Deepfakes*.

Por fim, ao utilizar as redes neuronais (*Xception*, *Mobile Net*) para criação dos modelos de classificação do *sub-dataset NeuralTextures*, ao avaliar a performance dos dois modelos, as matrizes de confusão, tal como apresentado na Figura 48, demonstram que a *Xception* possui uma maior acurácia e maior precisão em distinguir as classes (real e *fake*). A rede neuronal *Xception* demonstrou um equilíbrio superior nos valores das métricas entre as classes, tal como apresentado na Tabela 24.

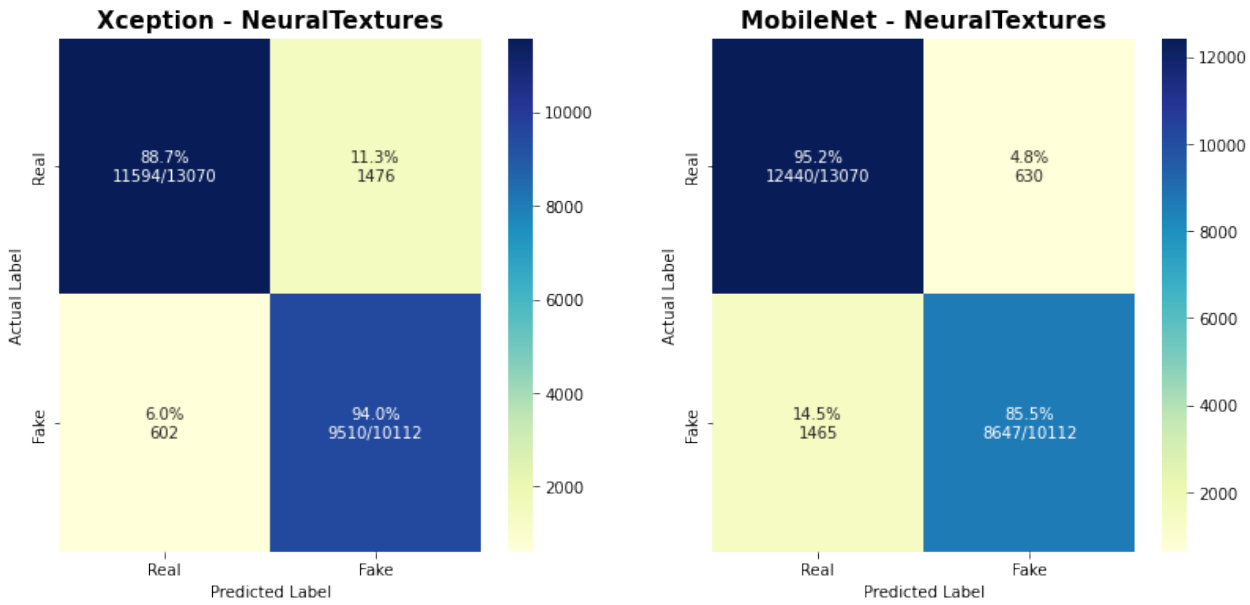


Figura 48: Matrizes de confusão das arquiteturas no *sub-dataset NeuralTextures*.

NeuralTextures	Xception	Mobile Net
Acurácia	91.04%	90.96%
Erro do tipo 1	6.0%	14.5%
Erro do tipo 2	11.3%.	4.8%
Recall - classe real	0.89	0.95
Recall - classe *fake*	0.94	0.86
F1-Score - classe real	0.92	0.92
F1-Score - classe *fake*	0.90	0.89

Tabela 24: Valores das métricas de avaliação do modelos para o técnica *NeuralTextures*.

Desta forma, a *Xception* demonstrou ser a rede neuronal mais adequada para o problema de classificação especifica de cada um dos 4 métodos de criação *deepfake*. Contudo, esta alta *performance* na classificação de dados de validação foram atingidas em dados que são oriundos do mesmo método de criação *deepfake* utilizado nos dados de treino do modelo. Posto isto, surge a seguinte questão: dado um modelo de classificação, o qual é treinado com dados gerados a partir do método A, qual será a *performance* deste ao classificar dados *deepfake* criados a partir do método B?

Para responder tal questão, foi utilizado um modelo treinado com dados criados com o método *deepfake* A. Este é utilizado para classificar dados gerados pelos métodos *deepfake* B,C,D. Desta forma,os resultados demonstram a ineficácia dos modelos ao classificar dados *deepfake* criados a partir de um determinado método *deepfake* diferente do método utilizado durante o treino do modelo. Estes resultados encontram-se apresentados na Tabela 25

		Validado com:			
	Modelo	**Deepfake**	**Face2Face**	**FaceSwap**	**NeuralTextures**
Treinado com:	*Deepfakes*	**98.17%**	54.52%	54.79%	62.60%
	Face2Face	62.81%	**97.37%**	57.85%	57.39%
	FaceSwap	50.65%	50.34%	**98.27%**	56.32%
	NeuralTextures	81.46%	62.14%	52.90%	**91.04%**

Tabela 25: Acurácia de validação nos diferentes métodos de criação *deepfake* .

No entanto, os resultados da Tabela 25 há um modelo no qual obteve uma acurácia de validação superior a 60% em dois métodos *deepfake* distintos além do próprio método utilizado no treino do modelo. Este modelo foi treinado usando os dados do método *deepfake NeuralTextures*. Para uma maior compreensão deste resultado, foram analisadas as matrizes de confusão deste modelo para com os dados de validação dos diferentes métodos *deepfake*.

Ao classificar os dados *deepfake* criados pelo método *Deepfakes*, o modelo treinado com dados do *NeuralTextures* obteve a segunda melhor acurácia de validação sendo inferior somente para classificação dos dados com o mesmo método utilizado durante o treino. O modelo conseguiu com sucesso classificar imagens não manipuladas (classe Real) em relação as imagens pertencentes a classe *Fake*, tal como pode ser observado na Figura 49. Esta apresenta uma maior ocorrência do erro tipo 1, sendo 10% a mais quando comparado com a ocorrência do erro tipo 2.

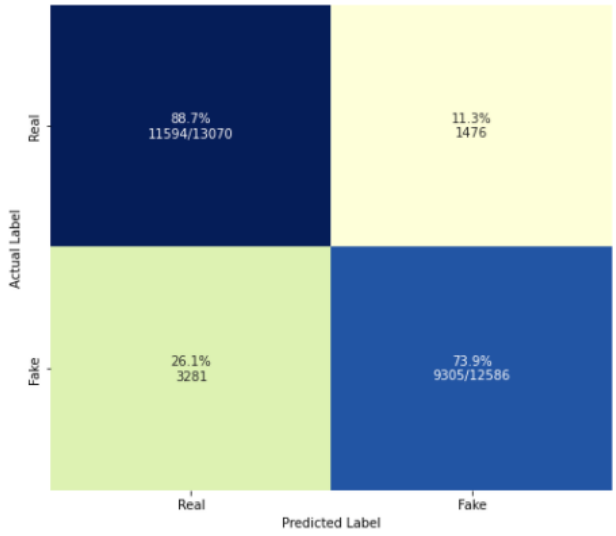


Figura 49: Matriz de confusão nos dados do método *Deepfakes*.

Na Figura 50 apresenta a matriz de confusão dos dados criados com o método *Face2Face*, no qual o modelo *NeuralTextures* obteve uma acurácia de validação de 61.14%. No entanto, a matriz demonstra uma incapacidade do modelo em distinguir com eficácia as classes (Real e *Fake*) dos dados apresentado a ele, *Face2Face*. O erro do tipo 1 ocorre seis vezes mais em relação ao erro do tipo 2, o que se traduz numa *recall* da classe *Fake* muito inferior a *recall* da classe real.

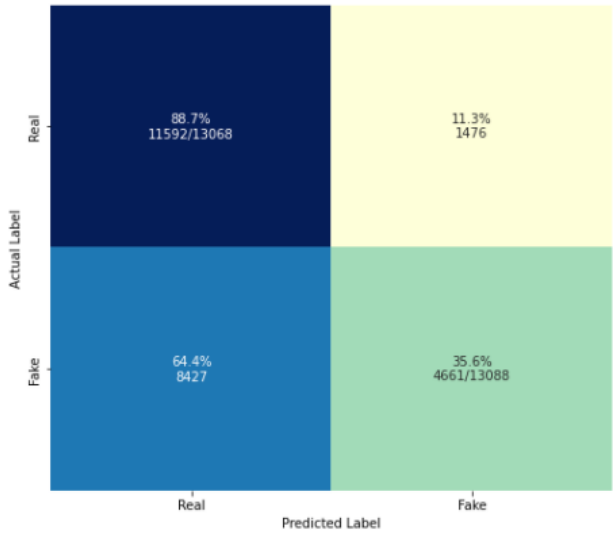


Figura 50: Matriz de confusão nos dados do método *Face2Face* .

Por fim, ao analisar a matriz de confusão do modelo ao classificar os dados do método *FaceSwap*, apresentado na Figura 51, obteve-se a menor acurácia de todos os 4 métodos, corroborando a incapacidade do modelo treinado com o método A ser capaz de detetar com eficiência conteúdo *fake* quando apresentado a novos dados gerados com método de criação *deepfake* do tipo B.

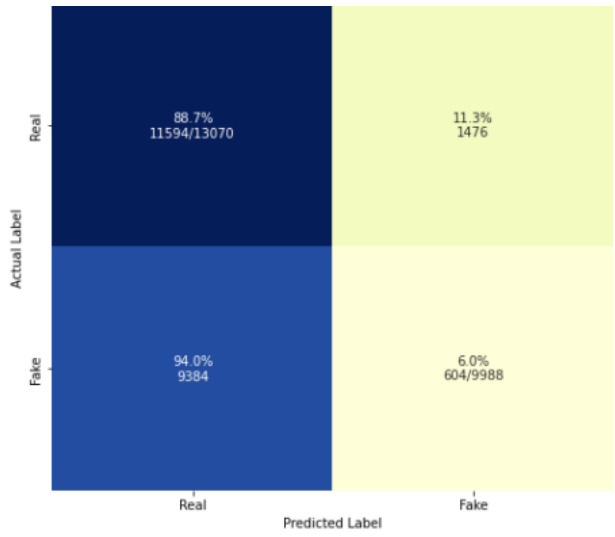


Figura 51: Matriz de confusão nos dados do método *FaceSwap*.

O modelo treinado com os dados *deepfake* do método *NeuralTextures* apresentou resultados ineficientes ao classificar dados oriundos de outros métodos *deepfake*, mesmo este sendo o mais eficiente na validação cruzada com dados de diferentes métodos da geração de *deepfake*. Desta forma, foi realizado um processo de otimização do modelo *NeuralTextures* com o propósito de reduzir a ineficiência do modelo ao classificar dados *deepfake* criados a partir de outro método diferente do utilizado nos dados de treino.

Este processo pode ser realizado com a configuração dos hiperparâmetros da rede neuronal ou a alterar os processos de preparação de dados de treino, designado *Data augmentation*. Este trabalho

realizou um processo de *Data augmentation*, o qual proporciona uma ampla gama de alternativas para alteração dos dados de treino (e.g, rotação da imagem, borrá-la, alternância de zoom, distorcê-la, alterar a luminosidade da imagem, entre outras opções).

Há trabalhos que apresentam uma melhoria dos modelos ao utilizar o *Data augmentation*, através da inserção de ruído nos dados de treino. Desta forma, foi utilizado uma técnica introduzida pelo trabalho [80], designada *Cutout*. Esta técnica demonstrou que uma regularização simples como o ato de mascarar aleatoriamente regiões nos dados de entrada durante o treino, pode ser utilizada para melhorar a robustez e o desempenho geral das redes neuronais que utilizam camadas CNN. Dessa forma, foi utilizado esta técnica nos dados de treino para otimizar o modelo, tal como apresentado na Figura 52.

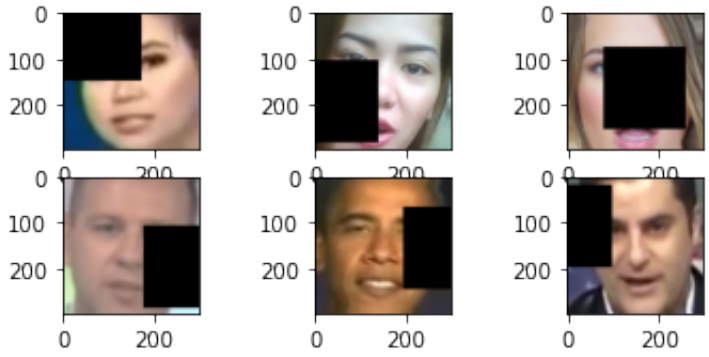


Figura 52: Aplicação da técnica de *cutout* nos dados de treino.

O modelo treinado sem a aplicação da técnica apresentava uma acurácia de validação de 91.04%, já o modelo a partir da utilização da técnica de *cutout* obteve uma acurácia de 56.91%. Este resultado demonstrou que após o processo de otimização, houve um decrescimento no desempenho da capacidade de classificação do modelo. Desta forma, a experiência posterior utilizou a abordagem multi-classificação.

3.4.2 Multi-classificação

Utilizou a arquitetura de rede neuronal *Xception*, pois essa obteve melhores resultados nas experiências anteriores, para a criar um modelo multi-classificação. Desta forma, foram utilizados os dados dos quatro *sub-datasets deepfake* bem como os dados não manipulados do *sub-dataset youtube*. Desta forma, foram utilizados os dados pré-processados das experiências posteriores.

Para isso, a rede neuronal foi adaptada para um problema de multi-classificação, apresentada na Figura 53, aonde foram utilizadas camadas CNN da *Xception* seguida de uma camada *GlobalMaxPool2D* e 4 camadas densas com 1024,512,128,64 neurónios respetivamente, com a função de ativação *Relu*. Dentre elas, foi aplicado um *Dropout* de 0,5. A ultima camada densa tem a quantidade de classes (i.e., 5) a qual se quer classificar.

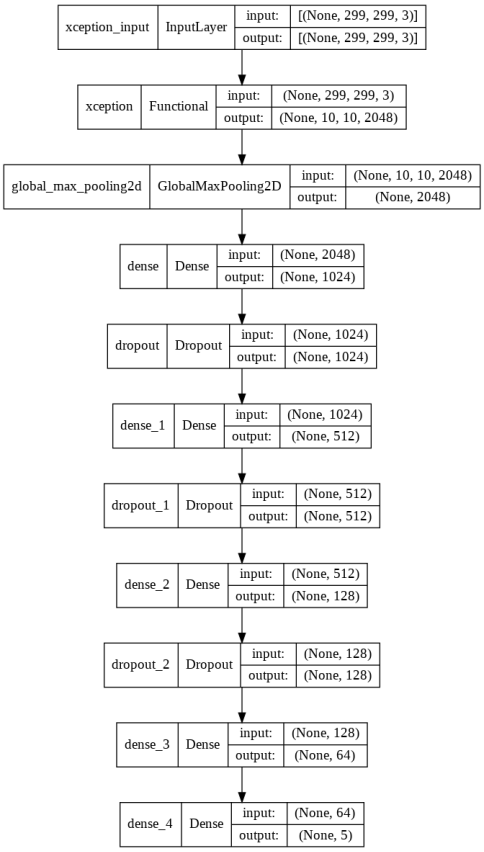


Figura 53: Arquitetura neuronal para multi-classificação.

Ao avaliar a performance do modelo com dados de validação, foi obtido uma acurácia de 51.03%. Este resultado é inferior aos modelos da classificação binária da experiência anterior. No entanto, para compreender melhor este resultado, analisou-se a matriz de confusão, apresentada na Figura 54.

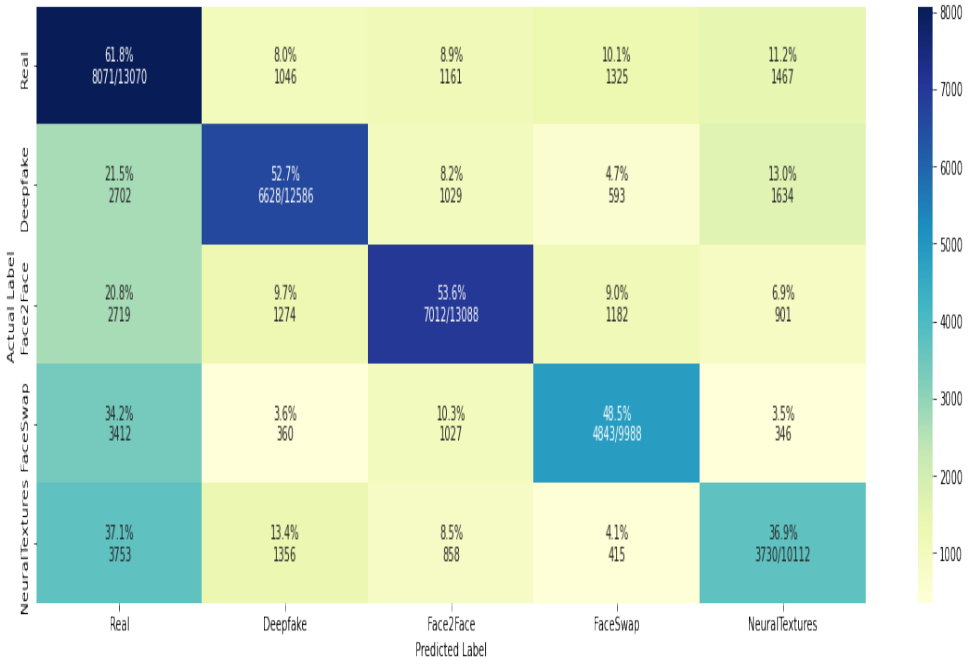


Figura 54: Matriz de confusão para o modelo de multi-classificação.

Há uma ligeira dificuldade do modelo em distinguir os métodos *deepfake* entre si e também em relação aos dados não manipulados(e.g, classe real). Ao verificar as métricas de performance do modelo, a métrica *recall* da classe real obteve um valor superior. No entanto, o *F1-score* apresentou resultado superior na classe *Face2Face*, tal como apresentado na Tabela 26.

Classe	Precision	Recall	F1
Real	0.39	0.62	0.48
Deepfakes	0.62	0.53	0.57
Face2Face	0.63	0.54	0.58
FaceSwap	0.58	0.48	0.53
NeuralTextures	0.46	0.37	0.41

Tabela 26: Métricas de avaliação de performance do modelo multi-classe.

No entanto, apesar do modelo criado ser de multi-classificação, o objetivo desse trabalho é a classificação binária (real ou *fake*), se a imagem extraída do vídeo em questão foi manipulada ou não. Para isso foi sintetizado os resultados para compreender a situação num contexto de classificação binária. Após sintetizar, a classe real continua apresentar resultados inferiores nas métricas de avaliação, tal como apresentado na Tabela 27.

Classe	Precision	Recall	F1
Real	0.39	0.59	0.47
Fake	0.86	0.74	0.80

Tabela 27: Métricas do modelo multi-classes em classificação binária.

Consequentemente foi analisado a matriz de confusão dos resultados sintetizados para uma classificação binária, tal como apresentada na Figura 54. Aonde se conclui que o modelo tem uma maior pré-disposição a classificar uma imagem como *fake*.

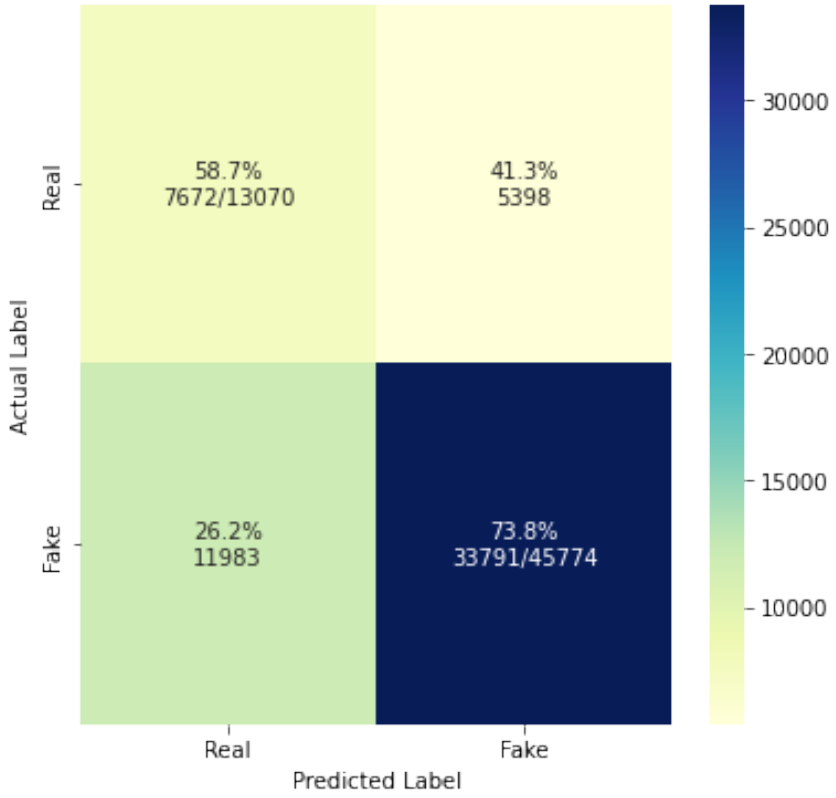


Figura 55: Matriz de confusão multi-classificação convertida para binária.

3.5 Validando o modelo em vídeos

Após as experiências realizadas anteriormente, o modelo escolhido para validar foi o modelo multi-classificação. Ao utilizar este modelo, foi necessário transformar sua classificação final numa classificação binária, uma vez que o propósito desse trabalho é ser capaz de distinguir se o vídeo analisado é ou não produto da aplicação de uma técnica *deepfake* (ao invés de saber qual a técnica propriamente utilizada para tal fim).

Para tal, foram selecionados vídeos oriundos dos *sub-datasets* de dados originais (*youtube*) e manipulados(*deepfakes*). Os dados selecionados de cada *sub-dataset* foram os que possuíam uma configuração de Frames Por Segundos (FPS) diferente dos dados utilizados no treino e validação do modelo multi-classificação, ou seja, todos os dados que possuem um FPS diferente de 30FPS. Desta forma, o total de vídeos retirados de cada *sub-dataset* foram 520 vídeos, tal como demonstrado na Tabela 15, resultando num total de 2.600 vídeos a serem analisados. Destes 98.54% (2.562 vídeos) foram de fato analisados nesta experiência, esta quantia foi composta por: 515 vídeos do *youtube*, 517 *Face2Face*, 511 *FaceSwap*, 507 *Deepfakes*, 512 *NeuralTextures*.

Os vídeos foram analisados integralmente, imagem a imagem, onde o modelo recebe como entrada cada imagem e classifica-a (*fake* ou real). Ao classificar N imagens que compõe um determinado vídeo, é apresentado a percentagem destas que foram preditas como da classe real ou *deepfake*, tal como apresentado nas Figuras 56 e 57. A Figura 56 é resultado da classificação do vídeo (youtube_458.mp4), este sem manipulação em seu conteúdo, enquanto que a Figura 57 apresenta a classificação do vídeo (Face2Face_458_722.mp4) do qual possui manipulação com a técnica *deepfake Face2Face*.

Figura 56: Vídeo real *youtube*

Figura 57: Vídeo *deepfake Face2Face*

Ao classificar os dados para validar o modelo, determinar se um vídeo é *deepfake* utiliza-se a maior percentagem de classificação após a analisar o vídeo integralmente. A exemplo a Figura 56 a qual representa um vídeo considerado pelo detetor como real, enquanto que o vídeo representado da Figura 57 classificado como um vídeo *deepfake*. Analisados os 2.562 vídeos, resultou num custo computacional de 214 horas, em tempo de processamento, foram criados meta-dados num ficheiro do formato *Comma-Separated Values (CSV)*, tal como apresentado na Figura 58, no qual contém as seguintes informações de cada analise:

- *Name*: o respetivo nome do vídeo que foi analisado;
- *Sub_dataset*: origem de qual o *sub-dataset* foi retirado;
- *Is_label*: a classe que o vídeo analisado pertence;
- *rated_target*: a classe estimada pelo modelo;
- *FN*: número de imagens que compõem o vídeo;
- *FPS*: número de Frames Por Segundos (FPS) do vídeo;
- *Time_to_sort*: tempo de processamento do vídeo para classificação;
- *Fake%*: percentagem da quantidade de *frame*(imagem) estimada pelo modelo com a classe *fake*;
- *Real%* percentagem da quantidade de *frame*(imagem) estimada pelo modelo com a classe real.

Name	Sub_dataset	Is_label	rated_target	FN	FPS	Time_to_sort	Fake%	Real%
11_youtube_924.mp4	youtube	Real	Real	1247	25.0	13	36	64
8_Face2Face_035_036.mp4	Face2Face	DeepFake	DeepFake	336	25.0	3	100	0
8_FaceSwap_035_036.mp4	FaceSwap	DeepFake	DeepFake	336	25.0	3	98	2
188_Deepfakes_303_309.mp4	Deepfakes	DeepFake	DeepFake	570	25.0	5	86	14
47_NeuralTextures_128_896.mp4	NeuralTextures	DeepFake	DeepFake	530	25.0	5	82	18

Figura 58: Amostra dos meta-dados provenientes da classificação dos vídeos.

Ao termino da analise dos 2.562 vídeos e a geração dos meta-dados, observou-se como o detetor se comportou ao classificar os dados. Para tal, foi utilizada a matriz de confusão, onde o detetor demonstrou uma maior eficiência ao classificar vídeos dos *sub-datasets deepfake*, tal como apresentado na Figura 59.

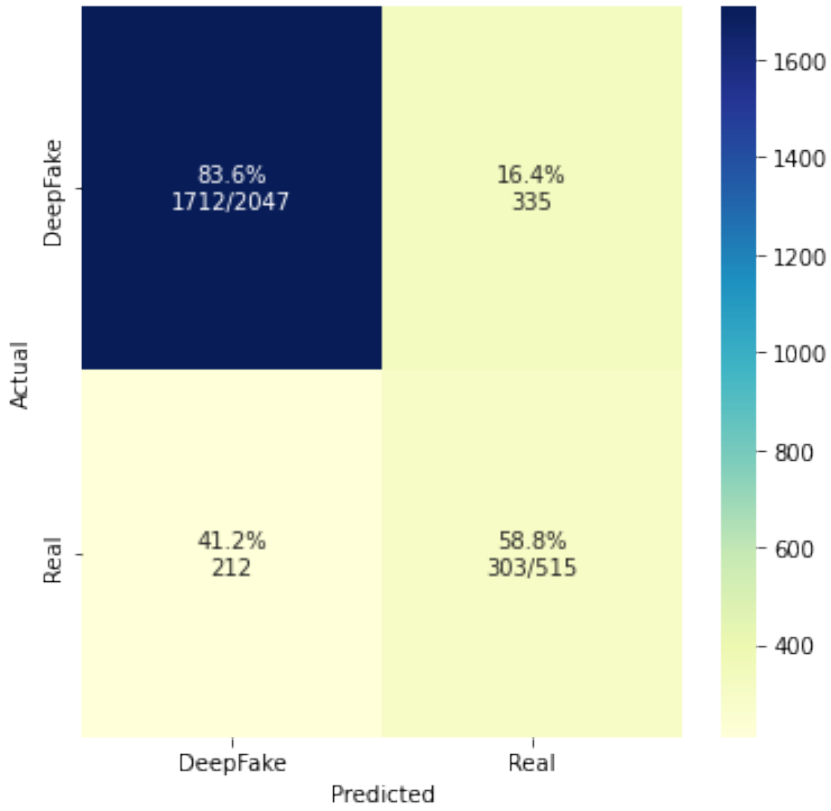


Figura 59: Matriz de confusão dos 2.562 vídeos.

Consequentemente, o erro de classificação para com a classe real é significativamente maior, em termos de percentagem, pois foram 212 (41.2%) vídeos classificados como *deepfake* quando na verdade eram dados referentes a classe real. Contudo, os 335 (16.4%) vídeos dos quais foram classificados como real pelo detetor, por meio dos meta-dados verificou-se que a quantidade de classificações erradas para cada *sub-dataset deepfake* foram de 114 vídeos para *NeuralTextures*, 110 para *FaceSwap*, 67 para o *Face2Face* 67 e o 54 para o *Deepfakes*, tal como apresentado na Figura 60.

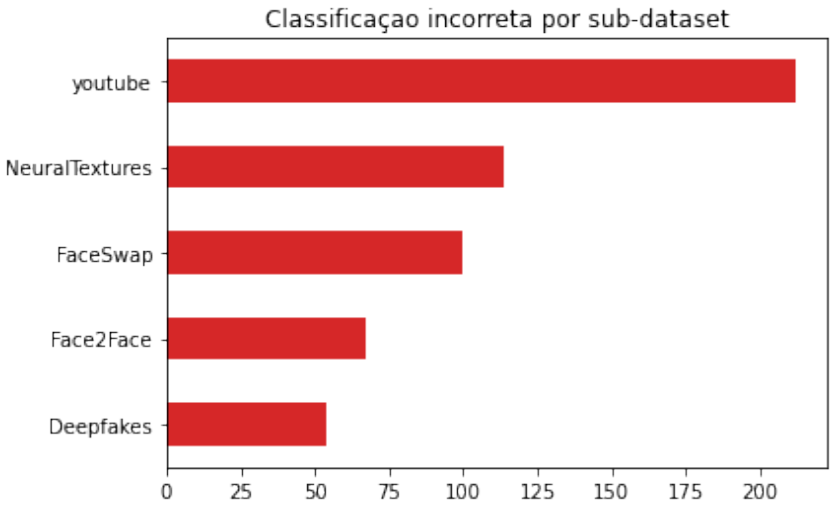


Figura 60: Erro de classificação por sub-dataset.

No entanto, ao rotular um vídeo a partir da maior percentagem de imagens classificadas de uma determinada classe não garante que o vídeo em questão recebeu o rótulo da classe apropriada. E.g., um determinado vídeo ao ser classificado com uma percentagem final de 50% *fake* versus 50% real, possui a mesma probabilidade de um evento do lançamento de uma moeda ao ar.

Como tal, foram estabelecidos valores para proporcionar um nível de confiança na classificação final, da qual o detetor utiliza para rotular o vídeo numa determinada classe, onde a percentagem de classificação maioritária for igual ou menor a 50% o nível de confiança da classificação é definido como **"Não Confiável"**; se a percentagem for maior que 50% e menor que 70% é considerado como **"Pouco Confiável"**; já quando a percentagem for igual ou superior a 70%, essa classificação é considerada como **"Muito Confiável"**. Os valores foram estabelecidos por via de uma analise empírica, a partir das matrizes de confusão de cada nível de confiança, tal como apresentado na Figura 61.

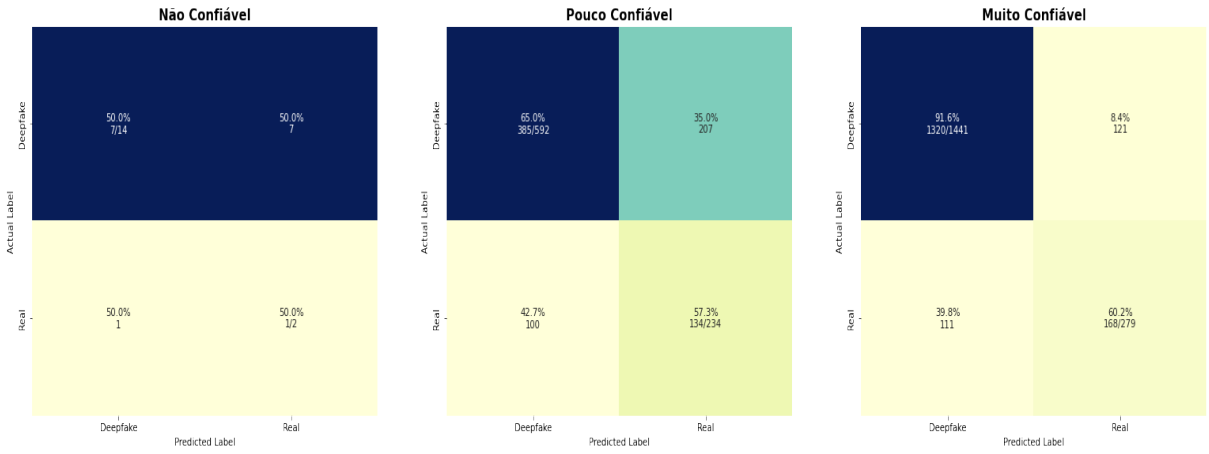


Figura 61: Matrizes de confusão por cada nível de confiança.

Ao estabelecer os valores de cada nível de confiança, surgiu a seguinte questão: qual a percentagem,

dentre as classificações realizadas nos respetivos *sub-dataset*, para cada nível de confiança? Sem considerar se a classificação estimada pelo detetor era correspondente a classe ao qual o vídeo pertencia. A resposta para esta questão demonstra que em todos os *sub-datasets* possui um elevado número de vídeos classificados com o nível de confiança "Muito Confiável", tal como demonstrado na Figura 62.

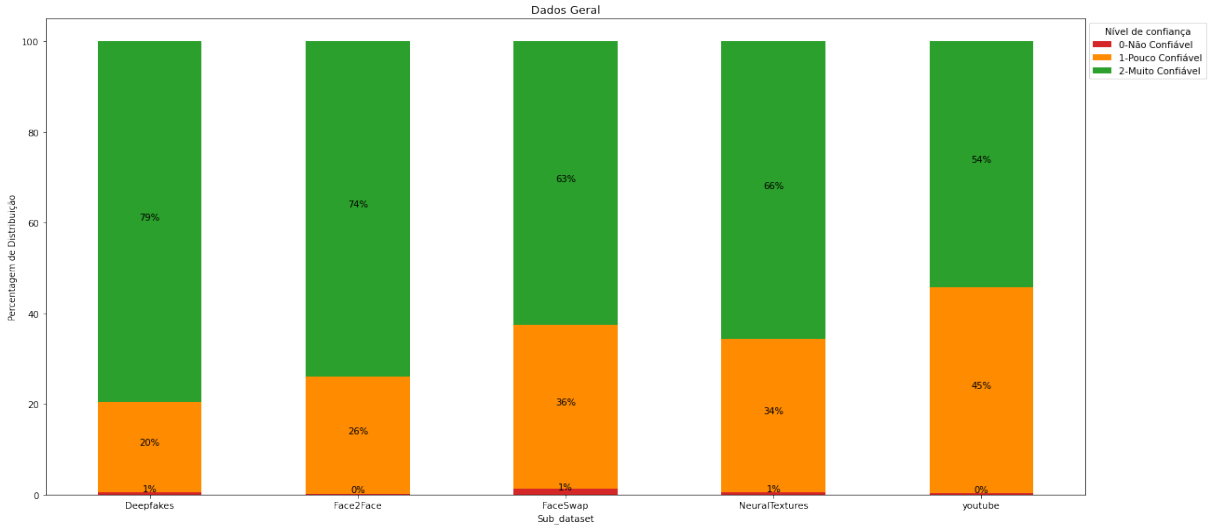


Figura 62: Distribuições dos dados de cada *sub-dataset* por nível de confiança.

No entanto, mesmo com a resposta, foi realizado uma investigação de maior precisão em cada *sub-dataset*. Tendo o conhecimento da classe estimada pelo detetor e classe de fato ao qual o vídeo analisado pertence, é possível saber quanto da estimações foram corretas ou não. Desta forma, ao analisar o *sub-dataset youtube* constatou-se que das classificações realizadas pelo detetor neste *sub-dataset* apresentou 279 vídeos classificações com nível de confiança "Muito Confiável", sendo destes 60% foram classificados corretamente. Enquanto que 234 vídeos foram classificados com um nível de confiança "Pouco Confiável", destes 47% tiveram sua classe estimada errada. Por fim, apenas 2 deles tiveram as classificações com o nível de confiança "Não Confiável", tal como apresentado na Figura 63.

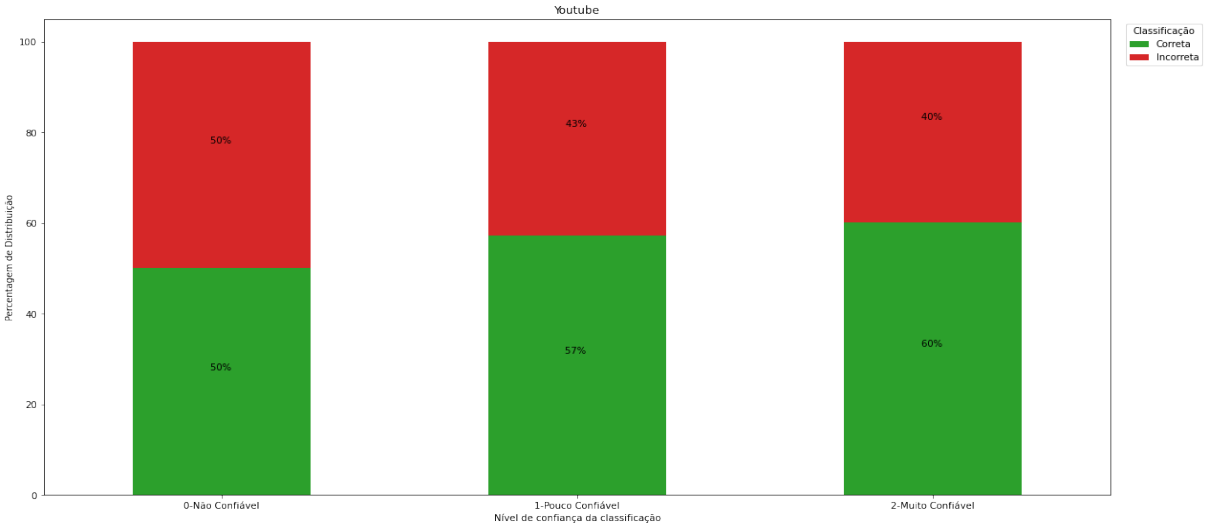


Figura 63: Classificações por nível de confiança do sub-dataset youtube.

Ao verificar as distribuições das classificações por nível de confiança atribuído a elas, conclui-se que quantidade de classificações estimadas pelo detetor corretamente no nível de confiança "Muito Confiável"é superior aos demais níveis de confiança (Não Confiável e Pouco Confiável). Esse fato demonstra que há mais assertividade na classificação quando a mesma possuem um nível de confiança "Muito Confiável"no que se refere aos dados originais.

Contudo é preciso perceber se o mesmo comportamento de assertividade acontece novamente em dados *deepfake*, independente da técnica de geração utilizada, consequentemente foi necessário uma analise a cada *sub-dataset deepfake*. Desta forma, foi analisado as classificações que o detetor realizou no *sub-dataset Deepfakes*. Destas classificações 403 vídeos foram classificados com o nível de confiança "Muito Confiável", sendo que destes 95% foram realizadas corretamente. Enquanto, que 101 vídeos foram classificados com o nível de confiança "Pouco Confiável", sendo que 33% o detetor errou ao estimar a classe. Apenas 3 vídeos foram classificados como "Não Confiável", tal como apresentado na Figura 64, corroborando ao fato que a precisão do detetor ao classificar vídeos do *sub-dataset Deepfakes* com nível de confiança "Muito Confiável"é superior as classificações com outros níveis de confiança.

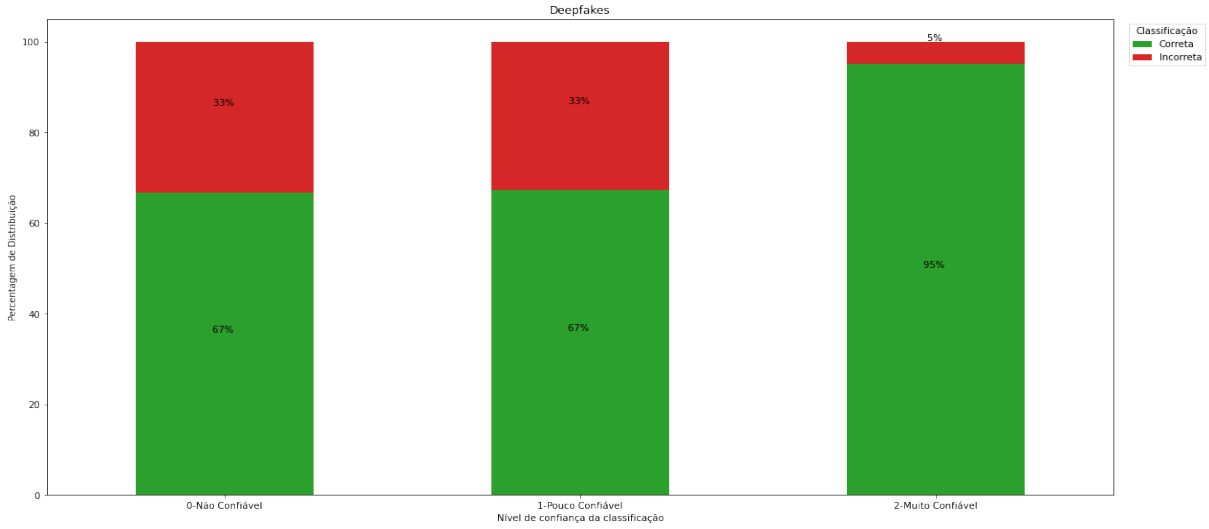


Figura 64: Classificações por nível de confiança do sub-dataset Deepfakes.

Quando o *sub-dataset* analisado é o *Face2Face* apresenta 382 classificações com nível de confiança "Muito Confiável", sendo que 92% com as classes estimadas corretamente. E o nível de confiança "Pouco Confiável"teve 134 vídeos, dos quais 72% tiveram a estimação da classe correta. Por fim, o nível de confiança "Não Confiável"teve apenas 1 vídeo. O nível de confiança neste *sub-dataset* é igual ao anteriores, no entanto, apresenta uma alta taxa de estimação da classes também nas classificações "Pouco COnfiável", tal como apresentado 65.

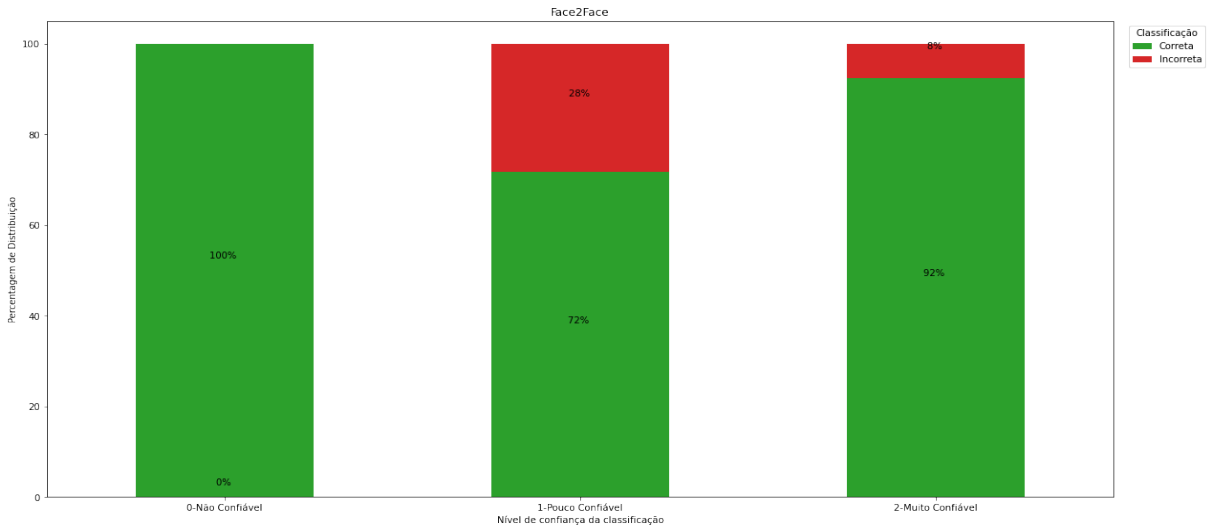


Figura 65: Classificações por nível de confiança do sub-dataset Face2Face.

Consequentemente no *sub-dataset FaceSwap* apresentou no nível de confiança "Muito Confiável"320 vídeos, sendo 86% dos quais o detetor estimou corretamente. Já o nível de confiança "Pouco Confiável"para este *sub-dataset* obteve 184 vídeos, dos quais 38% tiveram a sua classe estimada errada pelo detetor. Quando observa-se o nível de confiança "Não Confiável", este apresenta apenas 7 vídeos dos

quais 67% tiveram sua classe estimada erradamente também pelo detetor. Desta forma, é possível perceber que as classificações incorretas decorrem em maior percentagem nos níveis de confiança menor, tal como demonstrado na Figura 66.

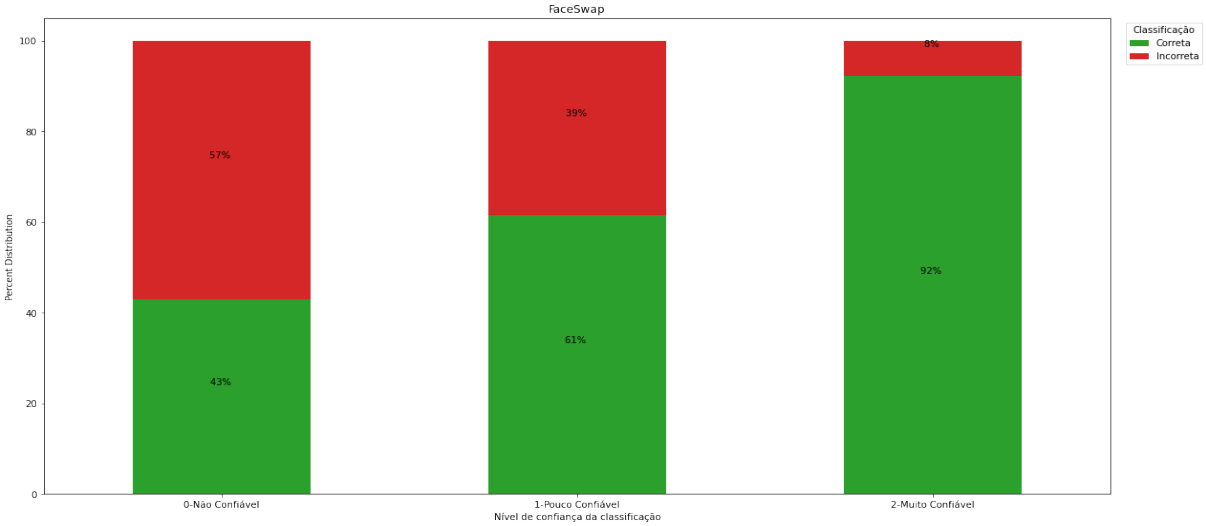


Figura 66: Classificações por nível de confiança do sub-dataset FaceSwap.

Por fim, o *sub-dataset NeuralTexures* apresentou 336 classificações, das quais possuem o nível de confiança "Muito Confiável", destas 86% foram estimadas corretamente. Enquanto que o nível de confiança "Pouco Confiável"possuem 173 vídeos classificados com este nível, destes 62% tiveram a classe estimada corretamente. Apenas 3 vídeos foram classificados como um nível de confiança "Não Confiável", tal como apresentada na Figura 67.

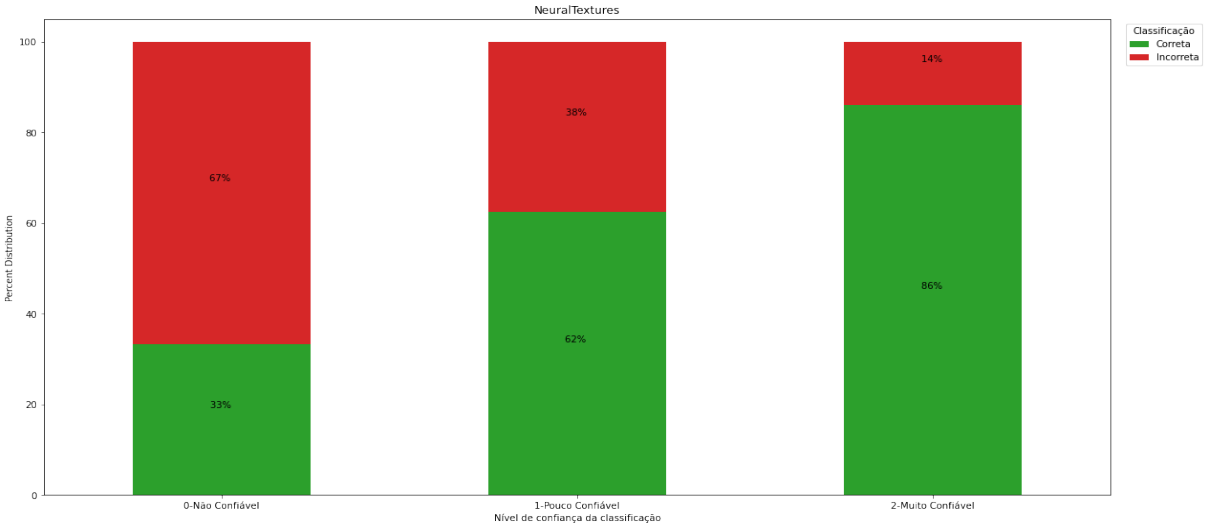


Figura 67: Classificações por nível de confiança do sub-dataset NeuralTexures.

Os resultados apresentados acima demonstram que a classificação dos vídeos *deepfake* nunca antes vistos pelo modelo, independente da técnica utilizada para criá-los, pois ao analisar as classificações nos

sub-datasets, foi possível verificar que as classificações estimadas corretamente pelo detetor é equivalente em todos os *sub-datasets*, sejam eles compostos por dados originais ou *deepfake*, havendo uma maior assertividade ao estimar a classe quando o nível de confiança atrelado a classificação é configurado como "Muito Confiável".

3.6 Discussão dos resultados

Das experiências realizadas para o problema da classificação de vídeos *deepfake*, foram realizadas experiências exploratórias, das quais demonstram ineficiência mesmo com o aumento do volume de dados, a rede neuronal foi incapaz de distinguir entre as classes (*fake* ou real). Em esperical a classe *fake*, pois os dados desta classe foram extraídas de vídeos de diferentes técnicas de criação *deepfake*. Com o propósito de verificar se a ineficiência das experiências exploratória foram resultados da junção de imagens extraídas de vídeos *deepfake* de diferentes técnicas *deepfake*, foram realizados experiências posteriores.

Desta forma, foi realizada uma experiência, da qual foi gerado um modelo para cada técnica de criação *deepfake*. Estes modelos demonstraram uma alta performance das redes (*Xception e Mobile Net*) tanto no treino quanto na classificação dos dados de teste das respetivas técnica *deepfake*. A rede neuronal *Xception* apresentou resultados ligeiramente superiores nas 4 técnicas *deepfakes* analisadas, de acordo com os critérios estabelecidos nesse trabalho.

Uma vez que os modelos apresentaram uma alta performance ao classificar dados de cada respetiva técnica *deepfake*, foi realizada uma validação cruzada, i.e., foi aplicado um modelo do qual teve seu treino com os dados de uma técnica *deepfake* X, para classificar dados de teste de uma técnica *deepfake* Y. Os resultados desta validação cruzada demonstraram que o modelo obtinha valores muito inferiores, quando classificava dados oriundos de uma técnica *deepfake* diferente da qual foi utilizada durante o seu treino. Posteriormente, foi escolhido o modelo *NeuralTextures*, o qual foi aplicado um processo de *Data augmentation* com o intuito de otimizar os resultados ao analisar dados de outras técnicas de geração *deepfake*. Todavia, não se obteve o resultado esperado.

Consequentemente, a quarta experiência foi a criação de um modelo multi-classificação, o qual apresentou resultados para os dados de teste oriundos de diferentes técnicas *deepfake*. Contudo, este estudo demonstrou um ligeiro *bias* para os dados *deepfake* ao sintetizar os resultados em classificação binária. Para analisar o impacto desse *bias* na classificação de vídeos, foi realizado uma quinta experiência da qual teve o propósito de validar o detetor e o modelo ao classificar 2.562 vídeos.

Ao analisar um vídeo é classificado todas as imagens que o compõe, obtendo uma percentagem de imagens classificadas como *fake* ou real. Desta forma, ao final da classificação do vídeo a percentagem maioritária define o rotulo que o mesmo receberá (i.e, *deepfake ou real*). Ao analisar a matriz de confusão da classificação de todos os vídeos ainda apresentava um ligeiro *bias* para classificar como *deepfake*.

Consequentemente, foram criados valores para estabelecer níveis de confiança para as classificações. Estes valores foram escolhidos empiricamente, com os níveis de confiança houve maior precisão ao classificar um vídeo, pois quando a classificação realizada pelo detetor possui um nível de confiança

"Muito Confiável"a probabilidade de acerto médio para classe *deepfake* é estimada em 91% e 60% para classe real. Em relação aos vídeos classificados com nível de confiança "Pouco Confiável", estes podem ser utilizados para trabalhos futuros num processo de *active learning* com o intuito de melhoraria da capacidade de classificação do modelo, de forma autónoma.

O modelo em questão tem possibilidades de melhorias, em especial quando se refere aos dados originais, no entanto, este apresenta uma capacidade de classificar vídeos *deepfake* independente da técnica utilizada para criá-los. Este fato é importante, pois os algoritmos apresentados no estado da arte possuem elevados valores para classificação em dados oriundos de um único método apenas.

Capítulo 4

Conclusão

O surgimento dos conteúdos *deepfake* têm alertado a sociedade democrática, devido ao crescimento do nível de desinformação nos meios digitais, consequentemente a colocar em risco a integridade moral até a física de indivíduos a exemplo da popularização de vídeos pornográficos *deepfake* com celebridades [6, 81] ou expondo instituições e Estados democráticos tal como o caso dos políticos responsáveis por várias capitais europeias, onde foram enganados em uma vídeo chamada por um individuo que utilizou-se do *deepfake* para se passar pelo Vitali Klitschko, autarca de Kiev [16].

Desta forma, ao verificar o estado da arte, diversos trabalhos de investigações focam-se nas técnicas de DL para detetar conteúdos *deepfake*, como forma de mitigar os problemas que estes trazem à sociedade democrática como um todo. Um crescente número de investigações com diferentes abordagens, as quais utilizam distintas arquiteturas neuronais para alcançar o objetivo da deteção de *deepfake*, através do uso de arquiteturas neuronais baseadas em CNN, ou em redes *Transformers*. No entanto, estes estudos apresentam modelos com excelente desempenho, a validar apenas com dados de uma única técnica de geração de *deepfake*.

No desenvolvimento desse trabalho foram realizadas experiências das quais foram testadas de acordo com os conhecimentos extraídos do estado da arte, como forma de explorar diferentes estratégias para mitigar o problema de classificação de vídeos *deepfake*. Desta forma, foram realizadas comparações de duas redes neuronais(*Xception* e *Mobile Net*) para criar modelos distintos para cada método de criação de vídeos *deepfake*. A partir destes modelos foram explorados uma solução mais generalista de classificação.

Por fim, com o desenvolvimento de um algoritmo de multi-classificação para detetar vídeos *deepfake*, foi possível uma classificação de vídeos *deepfake* independentemente da técnica de geração *deepfake*. Ao validar este algoritmo, foram criados níveis de confiança junto a classificação, nos quais proporcionaram maior precisão ao estimar uma classe ao vídeo.

4.1 Principais Limitações

Esse trabalho teve algumas limitações durante seu desenvolvimento. Uma das principais limitações foi referente a recursos computacionais, onde para solucionar tal limitação foi realizada a aquisição da

assinatura do *Colab Pro*. Outra limitação presente ao decorrer desse trabalho foi o aceder aos dados armazenados no *Google Drive* via *Colab*, devido a alta latência na comunicação entre os servidores de armazenamento do *Google Drive* e o servidor do *Colab*.

4.2 Trabalhos Futuros

Nos últimos anos, a utilização de *DL* para classificar conteúdos *deepfake* entre os investigadores tem vindo a crescer para mitigar o problema do alto número de casos crimes relacionados a fraudes, crime contra honra, de ataque ao Estado democrático, entre outros, dos quais aplicam conteúdo multimédia *deepfake* para fins ilícitos e disseminar desinformação nos meios digitais.

O resultado desse trabalho tem como objetivo mitigar a lacuna existente no estado da arte, o qual propõe algoritmos capazes classificar apenas vídeos gerados por uma técnica *deepfake* X a qual foi treinado para detetar. Entretanto, há espaço para melhorias nos resultados, tendo em consideração a plataforma de classificação e redução do tempo e recursos computacionais para execução do modelo; aplicação do grau de confiança para automatizar um pipeline de aprendizagem ativa / automática; desenvolver um detetor de anomalias para detetar *deepfake* (para colmatar futuras técnicas que venham a surgir para geração de vídeos Deep Fake).

Bibliografia

[1] Raquel Recuero. A conversação em rede: comunicação mediada pelo computador e redes sociais na internet. *Porto Alegre: Sulina*, 201, 2012.

[2] Adam M Enders, Joseph E Uscinski, Michelle I Seelig, Casey A Klofstad, Stefan Wuchty, John R Funchion, Manohar N Murthi, Kamal Premaratne, and Justin Stoler. The relationship between social media use and beliefs in conspiracy theories and misinformation. *Political behavior*, pages 1–24, 2021.

[3] Oscar de Lima, Sean Franklin, Shreshtha Basu, Blake Karwoski, and Annet George. Deepfake detection using spatiotemporal convolutional networks. *arXiv preprint arXiv:2006.14749*, 2020.

[4] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[5] Alex Hern. Reddit bans 'deepfakes' face-swap porn community, feb 2018. URL `https://www.theguardian.com/technology/2018/feb/08/reddit-bans-deepfakes-face-swap-porn-community`. The Guardian.

[6] Tyrone Kirchengast. Deepfakes and image manipulation: criminalisation and control. *Information & Communications Technology Law*, 29(3):308–323, 2020.

[7] Miles Brundage, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, Paul Scharre, Thomas Zeitzoff, Bobby Filar, et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*, 2018.

[8] Shadrack Awah Buo. The emerging threats of deepfake attacks and countermeasures. *arXiv e-prints*, pages arXiv–2012, 2020.

[9] The Wall Street Journal. Fraudsters used ai to mimic ceo's voice in unusual cybercrime case, aug 2019. URL `https://on.wsj.com/39jKEBu`.

[10] Craig Silverman. This analysis shows how viral fake election news stories outperformed real news on facebook, nov 2016. URL https://www.buzzfeednews.com/article/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook.

[11] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. The spread of low-credibility content by social bots. *Nature communications*, 9(1):1–9, 2018.

[12] Mika Westerlund. The emergence of deepfake technology: A review. *Technology Innovation Management Review*, 9(11), 2019.

[13] Ian Sample. What are deepfakes – and how can you spot them? https://www.theguardian.com/technology/2020/jan/13/what-are-deepfakes-and-how-can-you-spot-them, jan 2020.

[14] Andrei OJ Kwok and Sharon GM Koh. Deepfake: a social construction of technology perspective. *Current Issues in Tourism*, 24(13):1798–1802, 2021.

[15] MIT Technology Review. Novo aplicativo de inteligência artificial coloca mulheres em vídeos pornôs com um clique. https://bityli.com/NbULUgFU, oct 2021.

[16] Luke Harding. European politicians duped into deepfake video calls with mayor of kyiv, jun 2022. URL https://www.theguardian.com/world/2022/jun/25/european-leaders-deepfake-video-calls-mayor-of-kyiv-vitali-klitschko. (accessed Nov. 20, 2021).

[17] Omkar Prabhune, Pradnya Sabale, DN Sonawane, and CL Prabhune. Image processing and matrices. In *2017 International Conference on Data Management, Analytics and Innovation (ICDMAI)*, pages 166–171. IEEE, 2017.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[19] Peter Russell, Stuart J e Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[20] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[21] Karlijn Willems. Keras tutorial: Deep learning in python, dec 2019. URL https://www.datacamp.com/community/tutorials/deep-learning-python. (accessed Dec. 25, 2021).

[22] Mustapha Belaissaoui and Yassine Maleh. Machine learning techniques optimized by practical swarm optimization for intrusions detection in iot. *Journal of Information Assurance & Security*, 16(3), 2021.

[23] Jason Brownlee. When to use mlp, cnn, and rnn neural networks, jul 2018. URL `https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/`. (accessed Nov. 15, 2021).

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] Nura Aljaafari. Ichthyoplankton classification tool using generative adversarial networks and transfer learning, 02 2018.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[27] Jason Brownlee. *Generative adversarial networks with python: deep learning generative models for image synthesis and image translation*. Machine Learning Mastery, 2019.

[28] Vladimir Bok and Jakub Langr. *GANs In Action*. Manning Publications, 2019.

[29] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks, 2017. URL `https://arxiv.org/abs/1701.00160`.

[30] Barak Turovsky. Found in translation: More accurate, fluent sentences in google translate, nov 2016. URL `https://blog.google/products/translate/found-translation-more-accurate-fluent-sentences-google-translate/`. (accessed Jan. 20, 2022).

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[32] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[33] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[34] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[35] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *EASE*, volume 8, pages 68–77, 2008.

[36] Jorge Biolchini, Paula Gomes Mian, Ana Candida Cruz Natali, and Guilherme Horta Travassos. Systematic review in software engineering. *System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES*, 679(05):45, 2005.

[37] Mônica Cecilia De-la Torre-Ugarte, Renata Ferreira Takahashi, Maria Rita Bertolozzi, et al. Revisão sistemática: noções gerais. *Revista da Escola de Enfermagem da USP*, 45(5):1260–1266, 2011.

[38] Barbara Kitchenham. Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26, 2004.

[39] Abhijith Punnappurath and Michael S Brown. Learning raw image reconstruction-aware deep image compressors. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):1013–1019, 2019.

[40] Thanh Thi Nguyen, Cuong M Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, and Saeid Nahavandi. Deep learning for deepfakes creation and detection: A survey. *arXiv preprint arXiv:1909.11573*, 2019.

[41] Pearl Brereton, Barbara A Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of systems and software*, 80(4):571–583, 2007.

[42] Javier Hernandez-Ortega, Ruben Tolosana, Julian Fierrez, and Aythami Morales. Deepfakeson-phys: Deepfakes detection based on heart rate estimation. *arXiv preprint arXiv:2010.00400*, 2020.

[43] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3207–3216, 2020.

[44] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) preview dataset. *arXiv preprint arXiv:1910.08854*, 2019.

[45] Shahroz Tariq, Sangyup Lee, and Simon S Woo. A convolutional lstm based residual network for deepfake video detection. *arXiv preprint arXiv:2009.07480*, 2020.

[46] Davide Cozzolino, Justus Thies, Andreas Rössler, Christian Riess, Matthias Nießner, and Luisa Verdoliva. Forensictransfer: Weakly-supervised domain adaptation for forgery detection. *arXiv preprint arXiv:1812.02510*, 2018.

[47] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–11, 2019.

[48] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[49] Faceapp, 2019. URL `https://www.faceapp.com`. Accessed: 2021-01-20.

[50] Fakeapp, 2018. URL https://www1.folha.uol.com.br/mercado/2018/03/aplicativo-fakeapp-permite-manipulacao-realista-de-imagens.shtml. Accessed: 2021-01-20.

[51] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[52] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[53] Pavel Korshunov and Sébastien Marcel. Deepfake detection: humans vs. machines. *arXiv preprint arXiv:2009.03155*, 2020.

[54] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[55] Daniel Mas Montserrat, Hanxiang Hao, Sri K Yarlagadda, Sriram Baireddy, Ruiting Shao, János Horváth, Emily Bartusiak, Justin Yang, David Guera, Fengqing Zhu, et al. Deepfakes detection with automatic face weighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 668–669, 2020.

[56] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) preview dataset, 2019.

[57] Christian Keimel, Julian Habigt, Clemens Horch, and Klaus Diepold. Qualitycrowd - a framework for crowd-based quality evaluation. In *Picture Coding Symposium 2012 (PCS2012)*, pages 245–248, May 2012. doi: 10.1109/PCS.2012.6213338. URL ;.

[58] Irene Amerini, Leonardo Galteri, Roberto Caldelli, and Alberto Del Bimbo. Deepfake video detection through optical flow based cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[59] Luciano Alparone, Mauro Barni, Franco Bartolini, and Roberto Caldelli. Regularization of optic flow estimates by means of weighted vector median filtering. *IEEE transactions on image processing*, 8 (10):1462–1467, 1999.

[60] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995.

[61] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.

[62] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[63] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[64] Deng Pan, Lixian Sun, Rui Wang, Xingjian Zhang, and Richard O Sinnott. Deepfake detection through deep learning. In *2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, pages 134–143. IEEE, 2020.

[65] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[66] Dimensions. app.dimensions.ai/deepfake, jan 2020. URL `https://app.dimensions.ai/discover/publication?search_mode=content&search_text=deepfake&search_type=kws&search_field=full_search&or_facet_publication_type=article`. Accessed: 2021-01-20.

[67] Luisa Verdoliva. Media forensics and deepfakes: an overview. *IEEE Journal of Selected Topics in Signal Processing*, 14(5):910–932, 2020.

[68] Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. *arXiv preprint arXiv:1812.08685*, 2018.

[69] Deepfakes github, dec 2020. URL `https://github.com/deepfakes/faceswap`.

[70] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016.

[71] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[72] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[73] Google Research Nick Dufour and Jigsaw Andrew Gully. Contributing data to deepfake detection research. https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html, sep 2019.

[74] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge dataset, 2020.

[75] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu. Celeb-df: A large-scale challenging dataset for deepfake forensics. In *IEEE Conference on Computer Vision and Patten Recognition (CVPR)*, 2020.

[76] Lingzhi Li, Jianmin Bao, Hao Yang, Dong Chen, and Fang Wen. Faceshifter: Towards high fidelity and occlusion aware face swapping. *arXiv preprint arXiv:1912.13457*, 2019.

[77] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *CoRR*, abs/1604.02878, 2016. URL `http://arxiv.org/abs/1604.02878`.

[78] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.", 2008.

[79] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[80] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[81] MIT Technology Review. Novo aplicativo de inteligência artificial coloca mulheres em vídeos pornôs com um clique, oct 2021. URL `https://bityli.com/NbULUgFU`. (accessed Nov. 20, 2021).