UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Departamento de Sistemas de Computação

Estudo, desenvolvimento e comparação de técnicas de detecção de Deepfake

Felipe Manfio Barbosa

São Carlos - SP

Estudo, desenvolvimento e comparação de técnicas de detecção de Deepfake

Felipe Manfio Barbosa

Orientador: Fernando Santos Osório

Monografia referente ao projeto de conclusão de curso dentro do escopo da disciplina SSC0670 – Projeto de Formatura I do Departamento de Sistemas de Computação do Instituto de Ciências Matemáticas e de Computação – ICMC-USP para obtenção do título de Engenheiro de Computação.

Área de Concentração: Inteligência Artificial

USP – São Carlos
15 de Julho de 2020

“ O que todos devemos fazer é nos certificar que estamos usando a inteligência artificial de uma maneira que beneficie a humanidade, e não que a deteriore. ”

(Tim Cook)

Dedicatória

Dedico este trabalho às memórias de meu avô, Aldo Luiz Manfio, e minha bisavó, Luiza Fadoni, exemplos de bondade e dedicação à família.

Agradecimentos

Agradeço primeiramente a meus pais, Franciane e Gilson, aos quais serei eternamente grato por tudo o que fizeram e que fazem por mim, por todo o incentivo e suporte, pela ajuda em superar as dificuldades e por compartilhar as vitórias e os momentos de felicidade. Sem eles nada disso teria sido possível.

Agradeço também a todos os professores que me guiaram nessa jornada, desde os níveis mais básicos, e fundamentais, de minha educação, até o nível em que me encontro hoje. Em especial, agradeço a meu orientador, Fernando Santos Osório, que me orientou em praticamente tudo o que fiz na graduação, desde iniciações científicas até o presente projeto. Eles são a maior fonte de inspiração para meu desejo de continuar trilhando um caminho na ciência.

Finalmente, agradeço a meus amigos, em especial ao João, Leonardo, Vinícius, Thales, Hiago e Victor, que tornaram mais fáceis e agradáveis a vida longe de casa e as dificuldades da graduação.

Resumo

Os atuais avanços na área da Inteligência Artificial (IA), em especial os relacionados ao Aprendizado de Máquina e às Redes Neurais Artificiais (RNA), têm permitido a proposição de métodos cada vez mais versáteis e poderosos e também ampliado seu alcance para com a comunidade em geral, dado que o uso de tal tecnologia é hoje facilitado; um exemplo disso são os atuais aplicativos *mobile* baseados em aprendizado de máquina. Se tais métodos, porém, forem aplicados de forma mal-intencionada, diferentes esferas da sociedade podem sofrer graves consequências. Um exemplo de aplicação prejudicial são as denominadas *Deepfakes*. Combinação dos termos *Deep* (de *DeepLearning*) e *Fake* (falso), consiste na manipulação, por meio de algoritmos de aprendizado de máquina, de imagens ou vídeos com o fim de alterar as identidades ou ações das pessoas presentes nos mesmos. Este fenômeno motivou intensas pesquisas para a proposição de métodos que fossem capazes de detectar tais falsificações. Detectores baseados inicialmente na extração e análise "manual" de características estatísticas e visuais de imagens e vídeos tiveram que ser aprimorados para acompanhar o avanço na qualidade das manipulações. Passou-se, então, a utilizar arquiteturas de redes neurais para extração e análise automatizadas de características distintivas para a detecção de falsificações em imagens e vídeos. Outro resultado deste cenário é um intenso esforço de instituições públicas e privadas no sentido de incentivar a discussão a respeito do tema e a busca de novas soluções ao problema. Um exemplo disso é o *DeepFake Detection Challenge (DFDC)*, competição criada pela união de empresas como *AWS*, *Facebook* e *Microsoft* visando incentivar o estudo, desenvolvimento e a discussão a respeito de métodos de detecção por parte da comunidade. Neste trabalho, é realizado um estudo das técnicas de falsificação atualmente utilizadas, assim como o estudo e implementação das estratégias de detecção consideradas estado da arte em classificação de manipulações faciais. É utilizada uma base de dados criada pela união de diferentes bases de domínio público, constituídas por imagens e vídeos. O objetivo aqui é cobrir o número mais amplo possível de métodos de manipulação e níveis de compressão, dado que imagens e vídeos passam por esse processo quando disseminadas em redes sociais. Avalia-se, então o desempenho dos modelos com base em seu número de parâmetros e métricas de treinamento, sendo os com melhores resultados reunidos em sistema de classificação múltipla, buscando melhorar o desempenho geral de

detecção. O sistema resultante é, então, comparado aos resultados individuais dos modelos, assim como a referências de desempenho atuais na área. Por fim, são apresentadas possibilidades de extensões ao presente projeto e perspectivas a respeito do futuro da pesquisa na área.

Sumário

LISTA DE TABELAS

LISTA DE FIGURAS

CAPÍTULO 1: INTRODUÇÃO

1.1. Contextualização e Motivação

A popularização de smartphones e o crescimento das redes sociais tornaram a produção e divulgação de imagens e vídeos algo comum atualmente. De acordo com AFCHAR *et al.* (2018), há quase dois bilhões de uploads de fotos na internet e são assistidos mais de 100 milhões de horas de conteúdos em vídeo em redes sociais todos os dias. Muitas dessas fotos e vídeos retratam personalidades conhecidas, como celebridades e líderes políticos. Como mencionado por FRITH (2009), as faces têm papel central na interação humana, já que podem agregar significado ao que é dito, enfatizando uma mensagem, ou mesmo significar uma mensagem por si só. Dessa forma, faces de pessoas com grande poder de influência, como é o caso das celebridades e líderes anteriormente mencionados, carregam ainda mais poder. Sendo assim, a manipulação de faces em imagens e vídeos (ou mesmo de seus áudios) compartilhados na internet, principalmente em canais de grande alcance como as redes sociais, pode ter grande impacto na sociedade.

Paralelamente ao aumento da circulação de imagens e vídeos, ferramentas de edição se tornaram igualmente disponíveis. São bastante difundidas hoje ferramentas como o *Photoshop*, *MovieMaker*, *GIMP* e *Inkscape* – as últimas ferramentas livres. Essas, porém, são ferramentas de edição genéricas, que permitem manipular não só atributos das pessoas presentes em uma cena, mas também aqueles referentes à iluminação e objetos componentes da mesma. Além disso, a criação de manipulações de alta qualidade exige conhecimento especializado e experiência. Por outro lado, foram criadas também ferramentas automatizadas de edição, estas especializadas em um tipo de manipulação e que exigem menor grau de experiência para uso. Aqui podem ser citadas as ferramentas FaceSwap (KOWALSKI, 2018) e Face2Face (THIES; ZOLLHÖFER; NIEßNER, 2016), utilizadas para manipulações em faces e baseadas em métodos de computação gráfica para extração de características e sintetização dos resultados.

Em 2017, porém, o poder das redes neurais, mais especificamente as Redes Convolucionais (CNN, do inglês *Convolutional Neural Networks*), foi utilizado para tal

fim quando um usuário do *Reddit* com o nome “deepfakes” afirmou ter desenvolvido um algoritmo de aprendizado de máquina para substituir rostos de celebridades em filmes pornográficos (TOLOSANA *et al.*, 2016). Desde então, houve um crescimento acelerado no desenvolvimento de métodos de geração de falsificações, com melhorias na estrutura dos modelos, qualidade dos resultados finais e na facilidade de uso. Um passo importante nesse sentido foi dado com o uso das Redes Generativas Adversárias (GANs, do inglês *Generative Adversarial Networks*) (GOODFELLOW *et al.*, 2014). Desse modo, as habilidades de edição anteriormente exigidas não são mais necessárias e um usuário “comum” da internet pode gerar suas próprias falsificações sem muito esforço. A facilidade de uso e o fato de estarem disponíveis publicamente – código aberto - aumenta ainda mais sua difusão - os aplicativos *FaceApp* (FaceApp, 2017) e *ZAO* (LOUBAK, 2019), baseados na técnica de manipulação por meio de redes neurais e que viralizaram nos últimos anos, são exemplos do alcance das *deepfakes* mesmo na plataforma mobile, que tem menor poder de processamento.

Como resultado disso, a área de pesquisa conhecida como Multimídia Forense, tradicionalmente dedicada à detecção de manipulações gerais em imagens e vídeos, como adição ou subtração de regiões e duplicação de *frames*, passou a dedicar esforços para detectar manipulações faciais. Métodos incialmente baseados em padrões e análises estatísticas das imagens tornaram-se cada vez menos eficientes à medida que a qualidade das falsificações era aprimorada. Desta forma, passou-se a utilizar o poder das redes neurais como contramedida às suas equivalentes maliciosas, criando redes neurais que detectam *fakes* criados por outras redes neurais. Foram então propostos diversos modelos de detectores, com variadas premissas e arquiteturas, e com enfoques gerais ou em tipos específicos de manipulações. Estes métodos baseiam-se, contudo, no paradigma de aprendizado supervisionado, no qual é necessário um montante significativo de dados para treinamento dos modelos. Desta forma, além da proposição de novos métodos, foram propostas também novas bases de dados, assim como referências de desempenho (FaceForensics Benchmark, 2020) para os estudos e desenvolvimento subsequente na área.

Observa-se paralelamente um forte incentivo de combate aos “fakes” também por parte de empresas privadas, principalmente do ramo da tecnologia e, mais especificamente,

aquelas relacionadas a mídias sociais, como é o caso do *Facebook*. Por meio de desafios e competições, estas estimulam a discussão e geração de material a respeito do tema. Neste sentido, foi promovido recentemente o *DeepFake Detection Challenge* (FACEBOOK, 2019) (DOLHANSKY *et al.*, 2019). Este consistia de um desafio de programação, no *Kaggle* (KAGGLE, 2019), com prêmios em dinheiro para as soluções com melhores resultados para a métrica definida, e que propunha uma nova base de dados para treinamento e validação dos modelos. Finalizada a competição, seus resultados podem ser considerados como um novo marco de desempenho para futuras pesquisas.

Considerando ainda o contexto das redes sociais, um fator que pode degradar a acurácia na predição é o nível de compressão das imagens e vídeos, já que estes passam por variados tipos de compressão quando submetidas às redes sociais. Quanto maior o nível de compressão, mais difícil se torna a detecção dos artefatos deixados pelas manipulações. De mesmo modo, altos níveis de compressão podem introduzir artefatos que dificultem a classificação (NGUYEN *et al.*, 2019) (RÖSSLER *et al.*, 2019).

Sendo assim, este trabalho visa ao estudo da bibliografia atual sobre o tema, tanto relacionada à geração, quanto à detecção de falsificações em faces presentes em imagens e vídeos. No tocante à detecção de falsificações, serão estudados e implementados os métodos atualmente considerados o estado da arte na área. Para treinamento dos modelos, será utilizada uma base de dados própria, criada pela união de diferentes bases de dados atuais, de modo a garantir variedade de níveis de compressão e métodos de manipulação. Tais métodos serão então comparados segundo métricas como nível de acurácia, valor de perda e número de parâmetros. Aqueles com o melhor desempenho serão utilizados para a criação de um sistema de classificação múltipla, método final de detecção. Avalia-se, então, o desempenho do método proposto frente aos desempenhos individuais dos modelos que o compõem e frente às principais referências na área.

1.2. Objetivos

Este trabalho tem por objetivos realizar um estudo da literatura a respeito do tema da geração e detecção de manipulações faciais em imagens e vídeos, implementando os detectores estudados na linguagem *Tensorflow*, de modo a disponibilizá-los publicamente.

Busca-se propor um detector de falsificações baseado na união de um conjunto de modelos, a fim de agregar diferentes tipos de abordagens e alcançar melhor desempenho de detecção que aquele obtido pelos modelos individualmente - e que seja equiparável às referências atuais na área.

1.3. Organização do Trabalho

No Capítulo 2 são apresentados os conceitos base sobre os quais o projeto foi desenvolvido, bem como exemplos de trabalhos da literatura relacionada aos mesmos. A seguir, no Capítulo 3, são descritas as atividades realizadas durante o projeto. São apresentados e analisados os resultados obtidos por meio da metodologia empregada, com gráficos, tabelas referentes aos métodos implementados. É realizada ainda uma discussão a respeito das principais dificuldades e limitações do projeto. Finalmente, no Capítulo 5, são compiladas as principais contribuições do projeto, são apresentadas as conclusões e são feitas considerações a respeito de trabalhos futuros e perspectivas de pesquisas na área.

CAPÍTULO 2: REVISÃO BIBLIOGRÁFICA

2.1. Considerações Iniciais

Nesta seção são apresentados os principais conceitos, métodos, modelos, métricas e ferramentas utilizados no projeto e presentes nos principais trabalhos da literatura relacionada à detecção de falsificações em imagens e vídeos.

Inicialmente, é apresentada a teoria que fundamenta a geração das atuais *deepfakes*, sendo abordados: redes neurais artificiais, aprendizado profundo (*deep learning*), abordagens de aprendizado, redes neurais convolucionais e redes generativas adversárias. Posteriormente, aborda-se a geração de imagens e vídeos adulterados, apresentando os diferentes tipos de manipulação e os principais métodos, desde os iniciais até os mais atuais, utilizados para este fim. São discutidos os métodos e modelos considerados estado da arte em detecção e classificação de falsificações, assim como as principais bases de dados e referências de desempenho utilizadas na área. Finalmente, são apresentadas as métricas empregadas na análise dos resultados obtidos no presente projeto.

2.2. Redes Neurais Artificiais

Fazendo parte do paradigma de aprendizado conexionista, as redes neurais artificiais, de acordo com MONARD *et al.* (1999), são construções mate-máticas simplificadas inspiradas no modelo biológico do sistema nervoso (Figura 1), ana-logia que têm levado muitos pesquisadores a acreditar que as Redes Neurais possuem um grande potencial na resolução de problemas que requerem intenso processamento sensorial humano, tal como reconhecimento de voz e visão – escopo principal deste trabalho.

Sua unidade básica de funcionamento é o neurônio artificial (Figura 1), que aplica uma determinada função – chamada função de ativação, ou *f(x)* na Figura 1 - à soma ponderada de suas entradas ($\sum$), acrescidas ou não por um termo denominado bias (θ). Uma representação simplificada de um neurônio é exibida na Figura 1. Tais unidades são, por sua vez, organizadas em camadas, que são progressivamente “empilhadas”,

interconectando as diversas unidades, derivando desta característica o nome conexionismo (Figura 2).

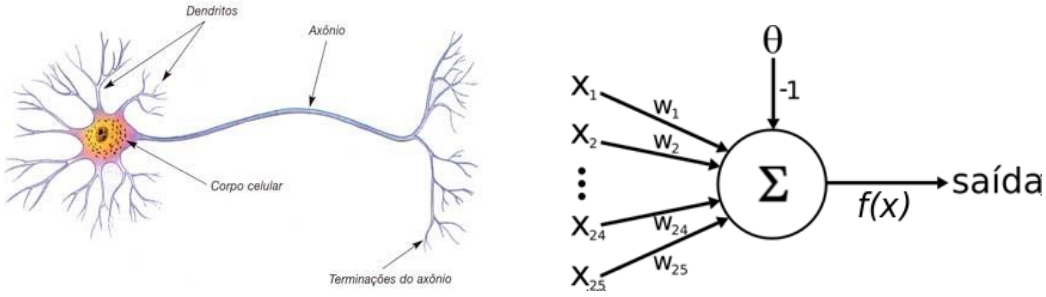

Figura 1 - Representação de uma célula de neurônio biológico e de um neurônio utilizado em redes neurais, em analogia ao biológico. Fonte: <https://bit.ly/3ezdc9y>.

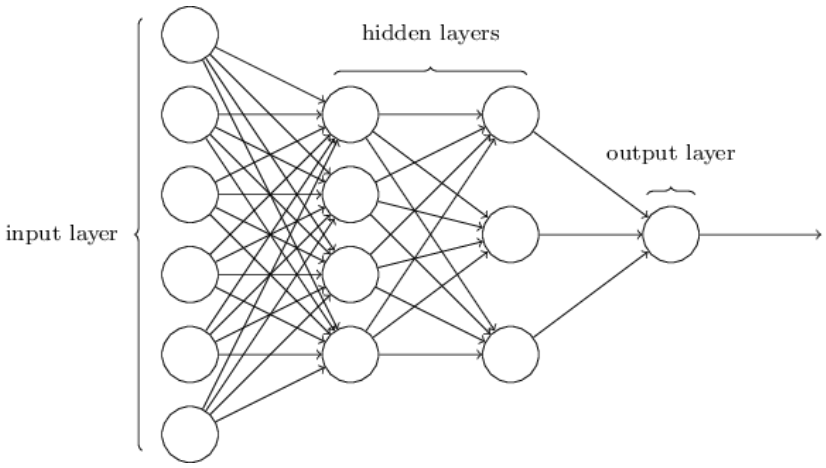

Figura 2 - Estrutura simplificada de uma rede neural. Fonte: <https://bit.ly/3fwbhns>.

Seu objetivo principal é aprender o mapeamento entre uma determinada entrada e sua saída correspondente. O processo pode ser descrito como: uma determinada entrada, com formato a depender do domínio de aplicação (imagem, texto), é fornecida à rede pela camada de entrada; esta estrada é processada pelos neurônios das consecutivas camadas constituintes da rede num processo denominado *forward propagation*, gerando uma saída; ocorre o contraste entre a saída fornecida e a esperada de acordo com alguma métrica, resultando um valor denominado perda (do inglês, *loss*); os pesos das conexões da rede são então atualizados, com base no valor de perda (“erro”) e segundo uma função denominada otimizador, por meio de um processo denominado *backpropagation*; uma nova entrada é então fornecida à rede e repete-se o processo.

O processo descrito se repete até que um número máximo de iterações seja atingido, ou até que algum critério previamente estabelecido seja atingido pela saída da rede. O conhecimento adquirido, ou seja, o mapeamento entre entrada e saída, é armazenado na forma dos pesos das ligações na estrutura interna da rede, resultantes do processo de treinamento. Cabe observar que o ciclo de processamento descrito diz respeito ao paradigma de aprendizado supervisionado, no qual é fornecido ao algoritmo de aprendizado, ou indutor, um conjunto de exemplos de treinamento para os quais o rótulo da classe associada é conhecido (MONARD; BARANAUSKAS, 1999).

2.2.1. Aprendizado Profundo (*Deep Learning*)

Redes Neurais com poucas camadas escondidas (*hidden layers*) são conhecidas como Redes Rasas, ou *shallow networks*. À medida em que mais camadas são adicionadas, a rede se torna gradualmente mais profunda, sendo então denominada Rede Profunda. Tarefas de aprendizado baseadas em modelos de redes profundas se enquadram, portanto, na abordagem de Aprendizado Profundo.

O aumento na profundidade da arquitetura utilizada traz também novos benefícios e preocupações. Por um lado, modelos profundos de redes neurais têm a capacidade de mapear funções mais complexas enquanto, por outro lado, há maiores chances de ocorrência dos fenômenos de *overfitting* e *vanishing gradient*, além de geralmente ocupar mais espaço de armazenamento e exigir maior tempo de treinamento, devido ao seu maior número de parâmetros.

2.2.1.1. Overfitting

Ao induzir conhecimentos, a partir dos exemplos disponíveis, é possível que a hipótese seja muito específica para o conjunto de treinamento utilizado. Como o conjunto de treinamento é apenas uma amostra de todos os exemplos do domínio, é possível induzir hipóteses que melhorem seu desempenho no conjunto de treinamento, enquanto pioram o desempenho em exemplos diferentes daqueles pertencentes ao conjunto de treinamento. Nesta situação, o erro em um conjunto de teste independente evidencia um desempenho

ruim da hipótese. Neste caso, diz-se que a hipótese se ajusta em excesso ao conjunto de treinamento ou que houve um *overfitting* (MONARD; BARANAUSKAS, 1999).

2.2.1.2. Vanishing Gradient

De acordo com HE *et al.* (2016) redes neurais profundas naturalmente integram características de baixo, médio e alto níveis, e esses níveis podem ser ainda mais enriquecidos conforme mais camadas "empilhamos" em nosso modelo. Dessa forma, é natural esperar que quanto mais profunda a rede, melhor seu desempenho. Essa concepção, contudo, é barrada pelo problema do *vanishing gradient*. Este fenômeno dificulta a convergência de modelos profundos pois, à medida em que sua profundidade aumenta, a atualização dos pesos é dificultada. Isso pode ser explicado pelo fato de as atualizações envolverem o cálculo de gradientes, que implicam em multiplicações. O fenômeno descrito ocorre quando os termos envolvidos em tais multiplicações se tornam tão pequenos, tendendo a 0, que a atualização do modelo é dificultada, portando não convergindo.

2.2.2. Redes Neurais Convolucionais

Redes neurais convolucionais (RNCs ou CNNs) são uma classe de redes neurais profundas comumente utilizada em aplicações de análise de imagens e que representaram um grande avanço no processo de extração e reconhecimento de padrões.

Inicialmente, o processo de extração de características em dados de imagens era realizado de forma manual, passando posteriormente por uma etapa de classificação por meio de algoritmo específico. Com a adoção das CNNs, tais características são automaticamente extraídas por meio do aprendizado neural, a partir de exemplos de treinamento - paradigma de aprendizado supervisionado, já descrito.

O uso de CNNs é especialmente poderoso quando aplicado em tarefas de reconhecimento de padrões em imagens, sendo seu pilar de funcionamento a operação de convolução, descrita a seguir.

2.2.2.1. A Operação de Convolução

A operação de convolução captura a natureza 2D de uma imagem, usando os chamados *kernels* de convolução para varrer a mesma. A varredura (escaneamento) é realizada deslocando o *kernel*, ou filtro, por toda a imagem. Realiza-se, a cada passo de deslocamento, também denominado *stride*, a multiplicação dos pesos do *kernel* pelos valores presentes nos canais de uma dada sub-região da imagem. Posteriormente, somam-se os resultados destas multiplicações. Neste processo são gerados os chamados mapas de características, descrições sumarizadas da imagem. Cada filtro de convolução gera um mapa de características. O procedimento descrito é ilustrado na Figura 3.

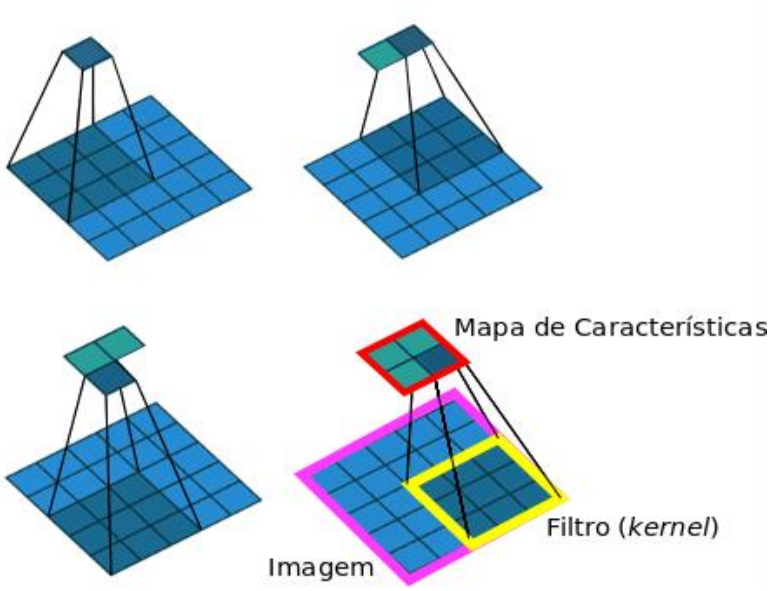

Figura 3 - Operação de convolução em imagens. Fonte: autoria própria.

Os filtros de convolução são adaptados por aprendizado e organizados em camadas de convolução, de modo que o número de filtros de cada camada é igual ao número de mapas de características retornados pela mesma. As camadas iniciais da rede aprendem conceitos gerais a qualquer tipo de domínio de imagens, como retas, curvas, cantos e bordas. As camadas seguintes, por sua vez, combinam o que foi aprendido (mapas de características) nas camadas anteriores em conceitos e estruturas mais complexas, e assim sucessivamente. Assim, a medida que nos aprofundamos na estrutura da rede, a informação

principalmente estrutural das primeiras camadas se torna progressivamente informação semântica do domínio (Figura 4).

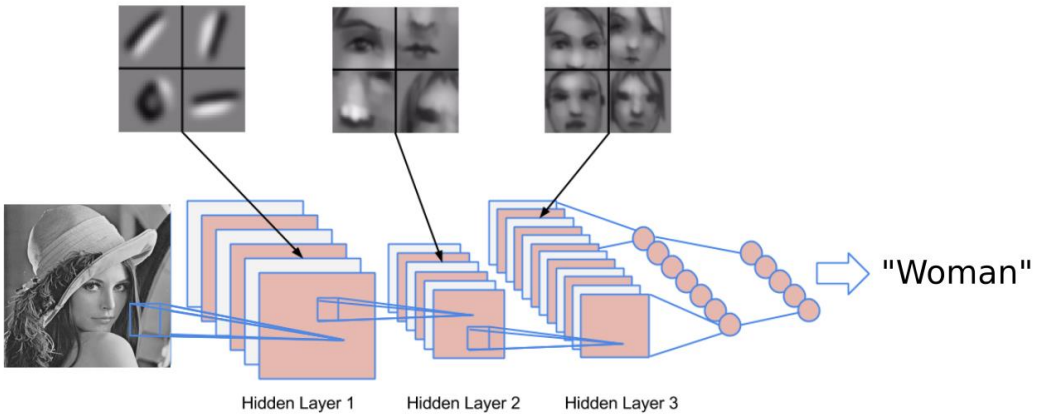

Figura 4 - Processamento de uma imagem através de uma CNN. Adaptado de: <https://bit.ly/38TIcjf>.

Uma variante da convolução padrão anteriormente descrita é a convolução 3D. Esta é agora composta por filtros com três dimensões. Desta forma, no processo de varredura da imagem, as multiplicações são realizadas considerando agora *kernels* que são volumes (três dimensões), aplicados também a volumes (canais das imagens). Esta difere da convolução 2D aplicada a múltiplos *frames* pois, diferentemente desta, a saída é também um mapa de características com volume (Figura 5). Esta característica auxilia no aprendizado de relações espaço-temporais entre as entradas fornecidas ao modelo, conforme demonstrado por (TRAN *et al.*, 2015).

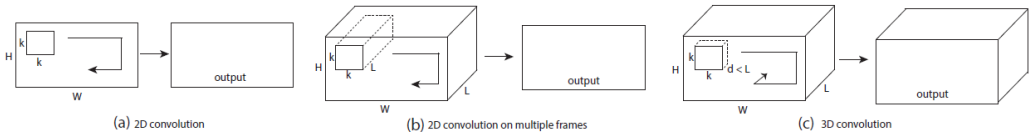

Figura 5 – Convolução 2D (a) conforme ilustrado na Figura 3, convolução 2D aplicada a múltiplos canais (b) e convolução 3D (c). Fonte: (TRAN *et al.*, 2015).

2.2.2.2. A Operação de Subamostragem

A operação de subamostragem (*pooling* ou *subsampling*) consiste na redução dimensional de um dado mapa de características, preservando apenas a informação mais relevante. Pode ser realizada principalmente por dois métodos, a saber: *Max Pooling* e *Average Pooling*.

No método denominado *Max Pooling*, é mantido apenas o valor máximo de uma dada sub-região do mapa, enquanto no método *Average Pooling* é calculada a média dos elementos da mesma (Figura 6).

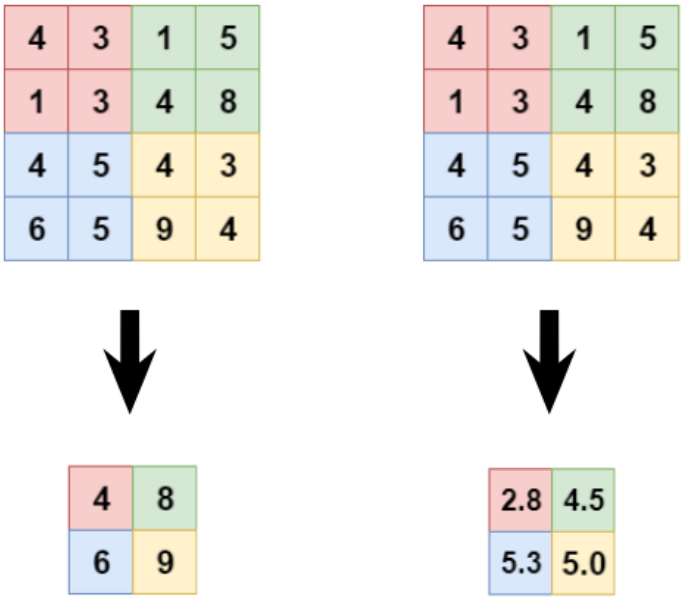

Figura 6 – Subamostragem por *Max Pooling* e *Average Pooling*, considerando um filtro de tamanho 2x2 e sem intersecções. Adaptado de: <https://bit.ly/3esZNQ3>.

A operação de subamostragem tem fundamental importância para as CNNs pois as convoluções introduzem muitos parâmetros na rede e se as dimensões fossem mantidas durante todo o processamento da rede, este processamento seria extremamente custoso computacionalmente. Desta forma, a amostragem, mantendo apenas características relevantes torna o treinamento deste tipo de rede praticável.

A adoção das CNNs cresceu de forma exponencial recentemente graças à grande disponibilidade de bases de dados rotulados para treinamento e validação, e também,

devido aos avanços no processamento paralelo e por meio de GPUs, que aumentou grandemente as capacidades de aprendizado e inferência. Modelos considerados marcos em tarefas de classificação (CHOLLET, 2017), detecção (REDMON *et al.*, 2016), segmentação (BADRINARAYANAN; KENDALL; CIPOLLA, 2017) e reconstrução (ISOLA *et al.*, 2017) são baseados em redes convolucionais.

2.2.3. Redes Neurais Residuais

Propostas como uma forma de solução ao problema do *vanishing gradient* (seção 2.2.1.2) as redes neurais residuais apresentam como principal inovação os blocos residuais (Figura 7). São compostas por um ramo direto e um ramo paralelo que funciona como uma espécie de passagem direta da entrada do bloco para a saída. Essa informação é adicionada ao resultado do processamento pelo ramo principal, passando então por uma função de ativação e sendo propagada pelo restante da rede.

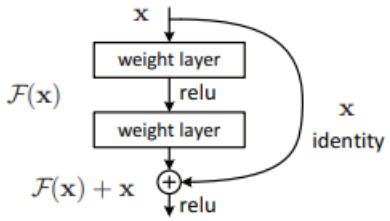

Figura 7 - Bloco residual. Fonte: (HE *et al.*, 2016).

Como pode ser observado na Figura 7, a conexão residual implementa a função de identidade, sem a adição de parâmetros nem de complexidade computacional. O mapeamento da identidade permite que um dado bloco opte por manter a conexão residual ou descartá-la - caso seu desempenho já seja ótimo considerando a tarefa sendo desenvolvida. Isso minimiza o fenômeno do *vanishing gradient* no sentido em que as conexões residuais permitem o fluxo do gradiente das camadas mais profundas até as mais rasas da rede.

Em HE *et al.* (2016), é demonstrada a efetividade do método por meio da comparação de estruturas de redes rasas e suas correspondentes profundas em diferentes bases de dados.

2.2.4. Redes Neurais Recorrentes

O funcionamento padrão de Redes Neurais envolve o processamento de entradas de maneira sequencial, contudo sem capturar a relação de dependência entre as mesmas – a sequência dos *frames* de um vídeo, por exemplo. As redes neurais recorrentes surgem como uma alternativa de solução para tal situação, capturando aspectos temporais.

Em Redes Neurais Recorrentes, a saída gerada é função não apenas da entrada fornecida, mas também de um estado interno que armazena informação a respeito de entradas anteriores. Sendo assim, é armazenada também informação referente a contexto, baseado em entradas/saídas anteriores (VENKATACHALAM, 2019). Em resumo, redes neurais recorrentes podem ser pensadas como possuindo laços de repetição, que permitem que a informação de passos anteriores persista. Alternativamente, tal estrutura de repetição pode ser pensada como múltiplas cópias da mesma rede, cada uma passando uma mensagem para seu sucessor (OLAH, 2015). A Figura 8 ilustra as ideias anteriormente apresentadas, sendo *x* a entrada, *A* o estado interno e *h* a saída da rede em uma determinada iteração.

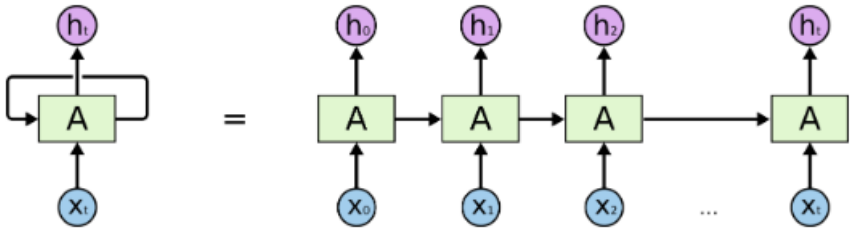

Figura 8 - Redes Neurais Recorrentes podem ser pensadas como Redes Neurais com um laço de repetição ou como cópias interligadas da mesma rede. Fonte: (OLAH, 2015).

Nesta estrutura uma mesma entrada pode gerar diferentes saídas, a depender de entradas anteriores na série de dados. Esta estrutura revela que este tipo de rede está intimamente ligado ao processamento de sequências e listas. Um exemplo pode ser dado considerando os *frames* de um vídeo, que preservam relação temporal/espacial uns com os outros.

Os principais avanços relacionados a Redes Neurais Recorrentes são, contudo, devidos a um tipo específico de arquitetura, as LSTM (do inglês, *Long Short Term Memory*) (HOCHREITER, 1997) – atualmente tem-se ainda uma variação da mesma, chamada GRU (do inglês, *Gated Recurrent Unit*) (CHO *et al.*, 2014).

2.2.5. Redes Generativas Adversárias

Redes Generativas Adversárias (GANs, do inglês *Generative Adversarial Networks*) (GOODFELLOW *et al.*, 2014) são uma classe de redes neurais profundas convolucionais utilizadas para tarefas de tradução de imagens (ISOLA *et al.*, 2017) (ZHU *et al.*, 2017). São basicamente compostas por duas sub redes, o Gerador e o Discriminador.

No processamento padrão em uma GAN, são fornecidos pares *entrada/saída esperada* para o modelo gerador. Este realiza o processamento da entrada de modo a gerar uma tradução da mesma para um outro domínio. O discriminador, por sua vez, tendo acesso ao par *saída gerador/saída esperada*, busca classificar as mesmas como reais ou falsas (sintetizadas). O erro de classificação, juntamente com uma medida do erro entre a sintetização e a saída esperada (erro de reconstrução) são combinados em uma medida conhecida como "perda adversária". Esta, finalmente, é utilizada no passo da retro propagação em ambos os modelos, ou seja, na atualização dos pesos. O processo continua até que um valor suficientemente pequeno de erro ou do valor de perda seja atingido. A Figura 9 exemplifica o processo descrito.

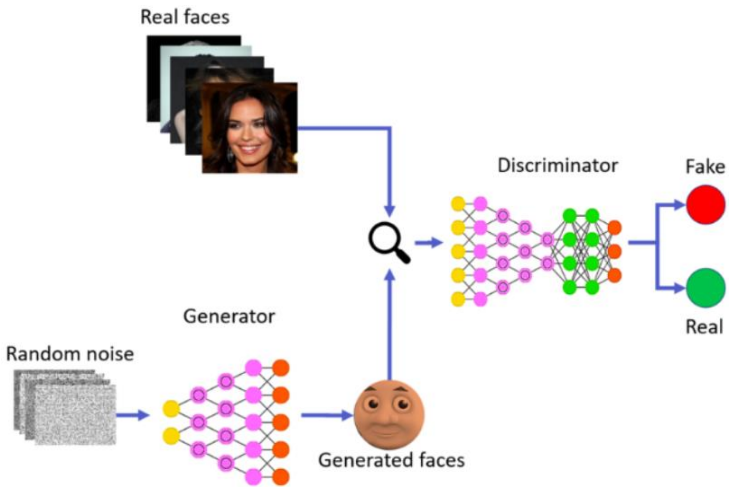

Figura 9 - Arquitetura simplificada de uma GAN. Fonte: <https://bit.ly/306ougg>.

Resumindo, durante o treinamento, o modelo gerador progressivamente sintetiza traduções mais fiéis ao resultado esperado, enquanto o discriminador se especializa em identificar cada vez melhor tais falsificações. O aprendizado neural tem fim quando o gerador fornece sintetizações de qualidade boa o suficiente para "enganar" o discriminador.

2.2.6. Transferência de Aprendizado

Em situações em que não há a possibilidade de treinar um modelo totalmente "do zero", seja por falta de dados suficientes, ou pela complexidade temporal/computacional exigida, é possível utilizar a prática conhecida como transferência de aprendizado (do inglês, *transfer learning*).

Esta técnica permite que o conhecimento adquirido pelo modelo em uma dada tarefa possa ser utilizado em outro domínio. Por exemplo, pode-se utilizar um modelo treinado em uma base de dados como a ImageNet (DENG *et al.*, 2009) e adequar o modelo a uma nova tarefa, digamos, distinguir entre rostos reais e rostos manipulados. Geralmente, o treinamento inicial, também chamado de pré-treinamento, é realizado considerando uma base de dados menor que a base de dados da tarefa em questão.

A calibração do modelo, ou *fine-tuning*, ocorre geralmente em duas etapas: calibração das camadas de classificação e re-treinamento do modelo. Na calibração das camadas de classificação, as camadas iniciais do modelo são mantidas inalteradas, enquanto o bloco de classificação tem seus pesos atualizados. Utiliza-se esta abordagem pois, como mencionado na seção 2.2.2.1, as camadas inicias correspondem a características comuns a qualquer tipo de imagem, de modo que podemos mantê-las inalteradas em um primeiro momento e apenas treinar a parte da rede que tenha sofrido alteração estrutural – considerando o exemplo anterior, como a rede treinada no ImageNet distinguia 1000 classes e agora o faz para apenas duas, sua camada final de classificação deve ser modificada e calibrada para se adequar ao novo domínio.

Calibradas as camadas do topo do modelo, ele é então treinado como um todo, seja tornando todos os níveis da estrutura treináveis ou ainda preservando alguns dos iniciais (MARTIN, 2019).

2.2.7. Sistema de Classificação Múltiplo (*Ensemble*)

Um classificador do tipo *ensemble*, também chamado de sistema de classificadores múltiplo, consiste em um conjunto de classificadores treinados individualmente, sendo suas saídas combinadas de alguma maneira para gerar a resposta final do sistema. O principal objetivo da união de diversos classificadores é reduzir a variância inerente às redes neurais. De fato, diferentes execuções, nas mesmas condições de treinamento (hiperparâmetros e arquitetura), podem resultar em modelos com pesos diferentes, a depender de fatores como a inicialização dos mesmos. Desta forma, a união de diferentes modelos em um sistema conjunto de predição tende a reduzir possíveis comportamentos inesperados derivados dessa aleatoriedade.

Neste projeto, é proposto um sistema múltiplo de classificação por meio da união das saídas de 3 modelos. Objetiva-se a agregar diferentes métodos de detecção de *deepfakes* utilizando como forma de combinação dos resultados um sistema simples de "voto majoritário", ou seja, a decisão com maioria dos votos (REAL ou FAKE) vence.

2.3. Geração de Falsificações

A seguir, são apresentados os tipos mais comuns de falsificações, assim como os métodos mais utilizados para a geração das mesmas.

2.3.1. Tipos de Falsificações

Os tipos de falsificações atualmente podem ser classificados, de acordo com o nível de manipulação utilizado, em 4 tipos principais (DANG *et al.*, 2019). TOLOSANA *et al.* (2016) apresenta uma descrição detalhada de cada um deles, os quais são descritos sucintamente a seguir, em ordem crescente de nível de manipulação.

2.3.1.1. Expressões Faciais

Consiste na modificação da expressão facial de uma pessoa. A modificação pode ocorrer pela transferência da expressão de uma pessoa para outra (AVERBUCH-ELOR *et al.*, 2017). Este tipo de manipulação pode ainda ser dividido em *source-to-target* e *self-reenactment*: no primeiro, as expressões de uma pessoa são transmitidas a outra; no segundo, a mesma imagem é fonte e destino da manipulação. A técnica mais popular de geração deste tipo de manipulação é conhecida como Face2Face (THIES *et al.*, 2016). A Figura 10 exemplifica tal tipo de manipulação.

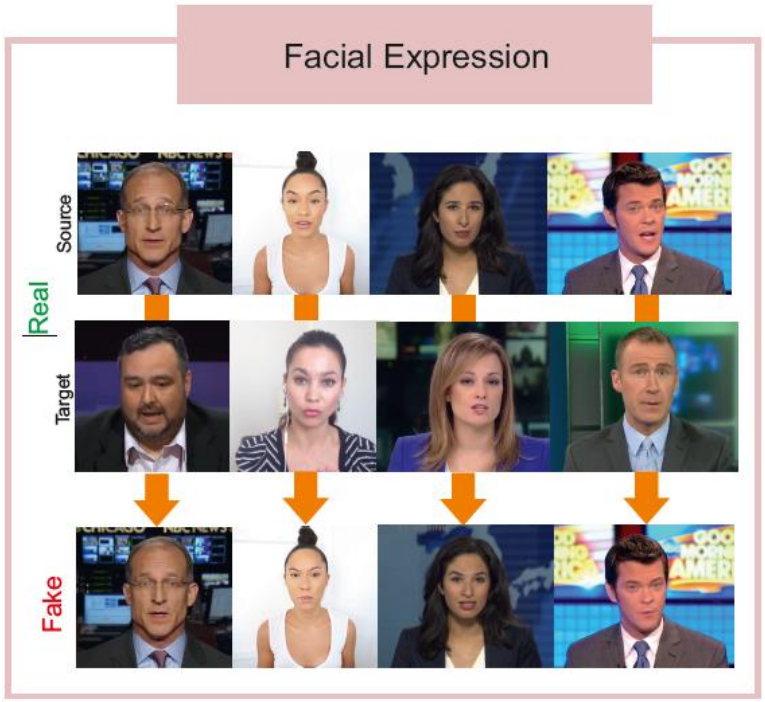

Figura 10 - Manipulação de expressões faciais. Fonte: (TOLOSANA *et al.*, 2016).

2.3.1.2. Atributos faciais

Consiste em modificar atributos como a cor da pele ou cabelo, o gênero, a idade, ou mesmo adicionar acessórios, como óculos. Geralmente usa-se GANs (seção 2.2.5) para gerar este tipo de manipulação (CHOI *et al.*, 2018). Um exemplo de aplicativo que opera segundo este tipo de manipulação é o popular *FaceApp* (FaceApp, 2017). A Figura 11 exemplifica tal tipo de manipulação.

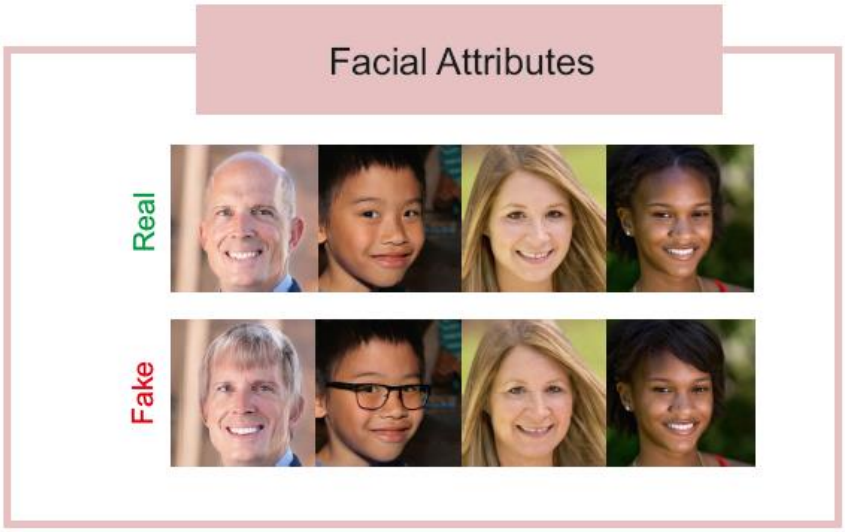

Figura 11 - Manipulação de atributos faciais. Fonte: (TOLOSANA *et al.*, 2016).

2.3.1.3. Troca de Identidades (*Face Swap)*

Talvez este seja o tipo de manipulação mais conhecido, dado que foi o precursor do atual contexto com relação à pesquisa em *deepfakes* e por estar amplamente disponível em vídeos do *YouTube*, como o conteúdo disponível no canal do brasileiro Bruno Sartori, que utiliza as manipulações para criar sátiras políticas e se entitula um "*deepfaker"*(SARTORI, 2012). Consiste em substituir a face de uma pessoa pela face de outra. Duas abordagens podem ser utilizadas para gerar tal tipo de manipulação: i) abordagem clássica baseada em técnicas de computação gráfica, como a técnica *FaceSwap* (KOWALSKI, 2018) e ii) abordagem recente utilizando técnicas de *deep learning* conhecida como Deepfakes. Um exemplo de aplicação comercial disponível ao público e que utiliza tal tipo de tecnologia é o já mencionado aplicativo ZAO (LOUBAK, 2019). A Figura 12 exibe exemplos deste tipo de manipulação.

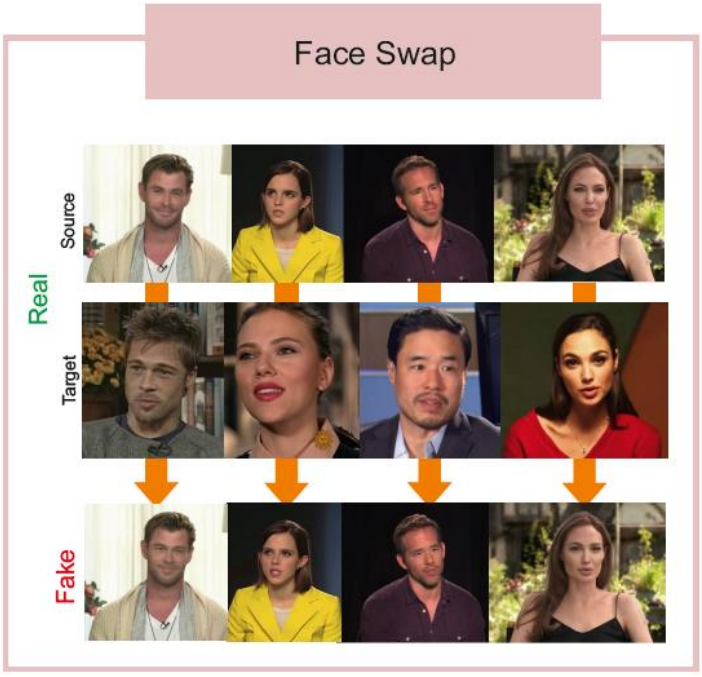

Figura 12 - Manipulação do tipo troca de identidades. Fonte: (TOLOSANA *et al.*, 2016).

2.3.1.4. Síntese facial

Este tipo de manipulação cria faces inteiramente novas, ou seja, sintetiza faces que não existem. Tal tipo de manipulação geralmente é gerado com o uso de redes GAN (KARRAS; LAINE; ALLA, 2019). Os resultados obtidos por este tipo de manipulação têm um nível surpreendente de qualidade e realismo (WEST; BERGSTROM, 2019) (WANG, 2019). A Figura 13 exibe exemplos deste tipo de manipulação.

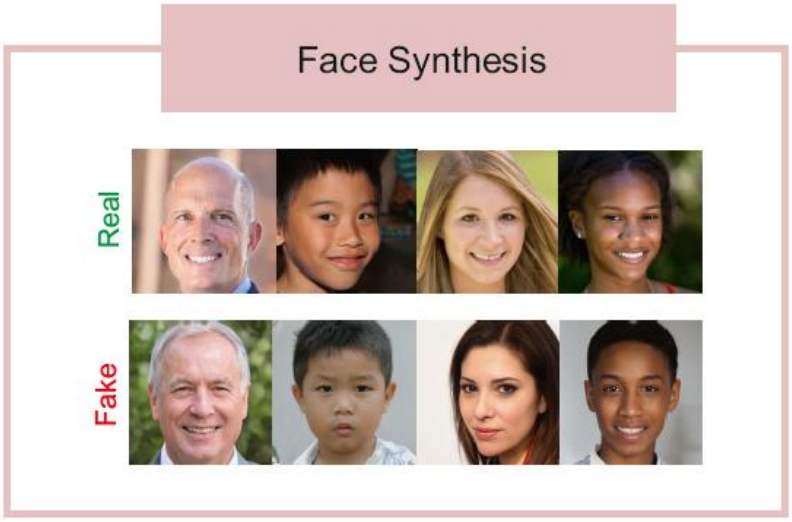

Figura 13 - Manipulação do tipo síntese facial. Fonte: (TOLOSANA *et al.*, 2016).

2.3.2. Principais Métodos de Manipulação

Considerando os principais tipos de manipulação apresentados na seção anterior, RÖSSLER *et al.* (2019) apresenta uma descrição dos principais métodos de falsificação utilizados atualmente, descritos a seguir.

2.3.2.1. FaceSwap

Abordagem baseada em computação gráfica, *FaceSwap* diz respeito a uma técnica para transferência da região da face de um vídeo de origem para um vídeo de destino. Inicialmente, cria um modelo 3D do rosto a ser usado como substituto. O modelo é então agregado à imagem de destino por meio da minimização do erro entre o modelo criado e as referências calculadas para a imagem de destino. Finalmente, é aplicado um processo de correção de coloração. A aplicação é leve e pode ser executada por meio de CPU.

2.3.2.2. DeepFakes

Apesar de ser utilizado para denominar um conjunto amplo de tipos de manipulações faciais, o termo *deepfakes* diz respeito a um método específico de

manipulação. No trabalho de NGUYEN *et al.* (2019) é descrita de forma clara e simplificada a ideia geral do processo de geração de deepfakes.

O modelo inicialmente utilizado se baseava em uma estrutura de autoencoder-decoder. O autoencoder realiza a extração de características latentes (representações em baixa resolução) das imagens dos rostos, enquanto o decoder reconstrói tais imagens a partir de sua representação latente. Para realizar a troca das faces são necessários dois decoders, cada um treinado em uma das bases de dados (faces de A e faces de B), e que compartilham o mesmo encoder. Essa estratégia, ainda segundo NGUYEN *et al.* (2019), permite que o encoder aprenda as similaridades entre os conjuntos de imagens, como olhos, nariz e posição da boca. Finalmente, os decoders são trocados, de modo que a partir das características extraídas para uma dada face A, seja gerada uma face com as características de B. Essa abordagem é aplicada em diversos trabalhos, como por exemplo o DeepFaceLab (PEROV, 2018). O processo descrito é exemplificado na Figura 14, a seguir.

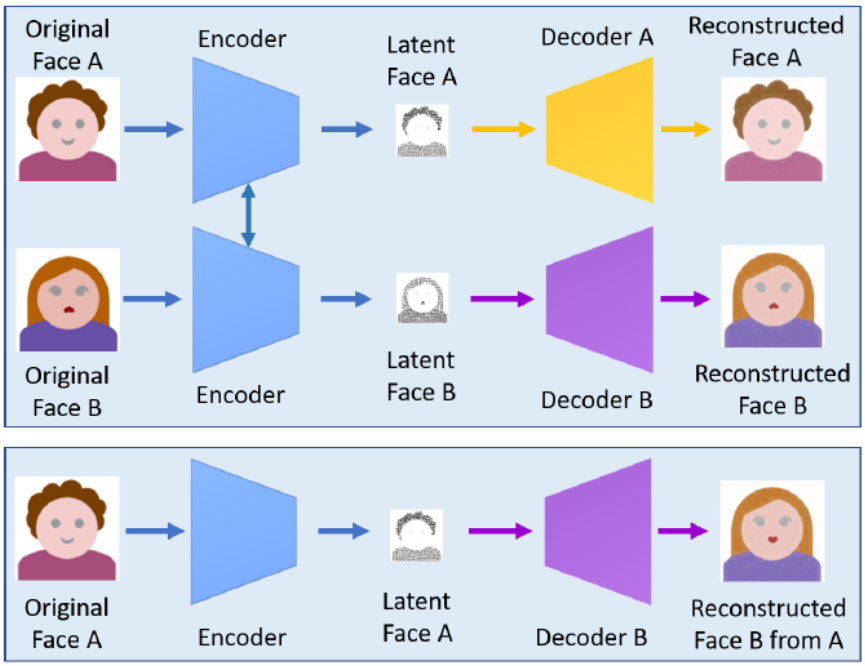

Figura 14 - O método Deepfakes. Fonte: (NGUYEN *et al.*, 2019).

2.3.2.3. Face2Face

A técnica chamada Face2Face diz respeito a um sistema de manipulação facial que transfere as expressões de um vídeo de origem para um vídeo de destino, mantendo a identidade da pessoa alvo. Sua implementação é baseada em duas entradas de vídeos, com seleção manual de *frames*-chave. Estes *frames* são então utilizados para gerar uma reconstrução densa da face, podendo então ser utilizada para re-sintetizar a face com diferentes expressões e em diferentes condições de iluminação.

2.3.2.4. NeuralTextures

Proposta no trabalho de THIES *et al.* (2019), esta técnica objetiva à manipulação de expressões faciais. Utiliza dados do vídeo original para aprender a "textura neural" da pessoa alvo, que é manipulada e posteriormente re-agregada por uma rede de renderização.

2.4. Detecção de Falsificações

A detecção de falsificações em imagens e vídeos tem sido tema de pesquisas na área de Multimídia Forense, ou mais especificamente Multimídia Forense Digital, desde antes do advento das *deepfakes*. Tinham por objetivo identificar manipulações como recorte, clonagem (cópia e colagem) de regiões em imagens e duplicação de *frames* em vídeos (FARID, 2019). Os métodos se baseavam na análise de características estatísticas, de irregularidades no pipeline de formação das imagens, ou ainda de distorções deixadas por manipulações (FERRARA *et al.*, 2012) (STAMM; WU, 2013) (JULLIAND; NOZICK; TALBOT, 2016) (BARNI *et al.*, 2017) (BONDI *et al.*, 2017). O surgimento das *deepfakes*, contudo, estimulou o aumento das pesquisas na área, e trouxe a necessidade do desenvolvimento de técnicas específicas para a identificação de manipulações relacionadas agora a faces. Este novo sub-ramo da pesquisa digital forense pode ser divido em detecção de imagens e detecção de vídeos falsos (NGUYEN *et al.*, 2019).

Com relação à detecção em imagens, foram propostas abordagens baseadas em artefatos e inconsistências "sutis" deixados pelos métodos de manipulação, como discrepâncias de cor, textura, pose da cabeça (YANG; LI; LYU, 2019) ou ainda

inconsistências nos olhos (LI; CHANG; LYU, 2018). Contudo, à medida que os métodos de manipulação evoluíram e passaram a gerar resultados mais realistas, métodos baseados nestes tipos de artefatos tiveram perda considerável de desempenho. Isto fez com que métodos baseados em *deep learning* para a detecção de inconsistências mais gerais fossem propostos (AFCHAR *et al.*, 2018). Estes últimos, contudo, ainda não abordam a influência da compressão de dados no desempenho da detecção. Uma análise neste sentido é considerada em RÖSSLER *et al.* (2019). Recentemente, no contexto do DFDC, o primeiro e terceiro colocados utilizaram como estratégia a classificação *frame* a *frame* por meio de um *ensemble* de diferentes variantes da recentemente proposta *EfficientNet* (MINGXING; QUOC, 2019). Uma abordagem semelhante é apresentada em BONETTINI *et al.* (2020).

Com relação à detecção em vídeos, métodos de detecção em imagens únicas não podem ser utilizados por conta da grande degradação introduzida nos *frames* pela compressão do vídeo. Além disso, vídeos apresentam características temporais que variam entre conjuntos de *frames*, as quais são difíceis de serem detectadas por métodos projetados para imagens (NGUYEN *et al.*, 2019). Desta forma, passa-se a analisar conjuntos de quadros, também chamados de janelas, na busca por características temporais referentes a manipulações.

A detecção baseada em características temporais leva em consideração o fato de os métodos de manipulação não considerarem a coerência temporal entre *frames*, já que são executadas quadro a quadro. Deste modo, artefatos produzidos pelos métodos de manipulação se manifestam como inconsistências temporais entre *frames*. Os principais métodos neste sentido se baseiam na associação de redes neurais convolucionais 2D e 3D (DOGONADZE; OBERNOSTERER; HOU, 2020), ou ainda de redes convolucionais com redes recorrentes (GÜERA; DELP, 2018) (SABIR *et al.*, 2019).

Apesar dos grandes avanços feitos na detecção de falsificações, métodos de geração evoluem de maneira igualmente rápida. Isso é ainda mais evidente quando consideramos o já mencionado atual grau de realismo e qualidade das sintetizações geradas pelas redes GAN. Porém, mesmo abordagens utilizando GANs deixam marcas características que podem ser aproveitadas no desenvolvimento de novos métodos de detecção (MCCLOSKEY; ALBRIGHT, 2018) (ZHANG; KARAMAN; CHANG, 2019).

2.6. Bases de Dados

Um modelo de classificação tem desempenho tão bom quanto os dados que lhe são fornecidos durante o treinamento. Dessa forma, uma parte importante do processo de desenvolvimento de um método de detecção de deepfakes baseado em *deep learning* é a escolha da base de dados. Com o aquecimento das pesquisas em detecção de manipulações faciais, houve também a criação de novas bases de dados, de modo a fomentar o desenvolvimento da área. Tais bases abrangem os diversos tipos de manipulação descritos anteriormente, assim como consideram diferentes fatores de compressão a que uma dada imagem ou vídeo pode ser submetido quando de sua divulgação por meio de mídias sociais. Algumas bases fornecem até mesmo as máscaras utilizadas na geração das falsificações.

Um ponto a se observar é a crescente preocupação com relação aos direitos de imagem das pessoas retratadas na base. Ao passo que as primeiras bases de dados neste sentido foram construídas pela união de vídeos disponíveis em plataformas como o *YouTube*, bases mais recentes, como a disponibilizada no *DeepFake Detection Challenge*, consistem de vídeos gerados por atores pagos.

As bases de dados utilizadas no presente projeto são descritas, por ordem cronológica, na Tabela 1, a seguir.

Tabela 1 - Bases de dados para treinamento em detecção de deepfakes.

Base	**Data**	**Composição**	**Características**
FaceForensics (RÖSSLER *et al.*, 2018)	2018	1004 vídeos, já somados reais e manipulados.	Vídeos reais extraídos do YouTube. Falsificações geradas com o uso do método Face2Face.
MesoNet (AFCHAR *et al.*, 2018)	2019	11509 imagens extraídas de vídeos reais e 7884 imagens manipuladas.	Imagens reais extraídas de vídeos do *YouTube*. Manipulações geradas por 2 métodos: DeepFakes e Face2Face

			(extraídas da base de dados FaceForensics).
FaceForensics++ (RÖSSLER *et al.*, 2019)	2019	1000 vídeos reais e 4000 vídeos resultantes de falsificações. Também são fornecidas as máscaras referentes às manipulações.	Vídeos reais extraídos do YouTube. Extensão da base FaceForensics, as manipulações são agora geradas a partir de 4 métodos: *Deepfakes*, *Face2Face*, *FaceSwap* e *NeuralTextures*.
DFDC (DOLHANSKY *et al.*, 2019)	2019	1131 vídeos reais e 4113 vídeos resultantes de falsificações.	Vídeos reais gerados por atores pagos. Manipulações de dois tipos, mas não é mencionado quais e nem os métodos utilizados para gerá-las. Base de dados adotada no "*DeepFake Detection Challenge*"
Celeb-DF (v2) (LI; YANG; SUN, 2019) (LI, 2020)	2019	890 vídeos reais e 5639 vídeos resultantes de falsificações.	Vídeos reais extraídos do YouTube, mostrando diferentes celebridades. Manipulações geradas com o uso de uma versão modificada do método *DeepFakes*.

A cada base de dados anteriormente descrita é associada uma referência de desempenho, ou *baseline*. Tal medida é definida a partir da performance obtida por modelos de referência quando treinados e avaliados em tais bases, segundo diferentes métricas de desempenho. A seção seguinte apresenta, de forma detalhada, tanto as métricas quanto as referências utilizadas no contexto do presente projeto.

2.7. Avaliação de Desempenho

A seguir são descritas as métricas a serem utilizadas para a avaliação dos modelos implementados. São apresentadas, também, as principais referências de desempenho na área.

2.7.1. Métricas de Desempenho

2.7.1.1. LogLoss/Categorical Crossentropy

LogLoss (Equação 1) é uma medida do erro de classificação de um modelo, cuja saída é um valor entre 0 e 1. O objetivo dos modelos de aprendizado de máquina é minimizar esse valor, de modo que um modelo perfeito apresentaria valor 0 para esta métrica. Seu valor aumenta à medida que a probabilidade prevista diverge da classificação esperada.

$$\text{LogLoss} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

Equação 1 – *LogLoss*.

Em que:

- *n* é o número de imagens/vídeos sendo preditos;
- *ŷi* é a probabilidade predita da imagem/vídeo ser da classe FAKE;
- *yi* é 1 se a imagem/vídeo é da classe FAKE, 0 da classe REAL;
- *log()* é o logaritmo natural (base *e*).

Categorical Crossentropy (Equação 2) é uma métrica equivalente à *LogLoss*, porém utilizada quando se tem mais de uma saída na rede, isto é, há uma saída para cada classe a ser predita. Neste caso, cada saída representa a probabilidade de a entrada pertencer à classe em questão, de modo que a soma das saídas deve ser igual a 1.

$$CCE = -\frac{1}{N}\sum_{i=0}^{N}\sum_{j=0}^{J} y_j \cdot log(\hat{y}_j) + (1 - y_j) \cdot log(1 - \hat{y}_j)$$

Equação 2 - *Categorical Crossentropy*.

Em que:

- *N* é o número de imagens/vídeos sendo preditos;
- *J é o número de classes possíveis (número de saídas);*
- *ŷj* é a probabilidade predita da imagem/vídeo ser da classe *j*;
- *yj* é 1 se a imagem/vídeo é da classe *j*, 0 caso contrário;
- *log()* é o logaritmo natural (base *e*).

No contexto do presente projeto, a classe a ser detectada é a FAKE, sendo, portanto, rotulada como [0, 1] e a classe REAL como [1, 0].

2.7.1.2. Matriz de Confusão

Uma forma de avaliar o desempenho de um modelo de classificação é por meio da matriz de confusão. Segundo MONARD *et al.* (1999), "a matriz de confusão de uma hipótese *h* oferece uma medida efetiva do modelo de classificação, ao mostrar o número de classificações *versus* as classificações preditas para cada classe, sobre um conjunto de exemplos *T*. Os resultados são organizados em duas dimensões: classes verdadeiras e classes preditas, para *k* classes diferentes" (Tabela 2).

Tabela 2 - Matriz de Confusão de um classificador. Fonte: (MONARD; BARANAUSKAS, 1999).

Classe	predita C_1	predita C_2	$\cdots$	predita C_k
verdadeira C_1	$M(C_1, C_1)$	$M(C_1, C_2)$	$\cdots$	$M(C_1, C_k)$
verdadeira C_2	$M(C_2, C_1)$	$M(C_2, C_2)$	$\cdots$	$M(C_2, C_k)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
verdadeira C_k	$M(C_k, C_1)$	$M(C_k, C_2)$	$\cdots$	$M(C_k, C_k)$

Para o presente projeto, a matriz de confusão será composta por 2 linhas e 2 colunas, já que um dado exemplo pode ser classificado como verdadeiro ou *fake*. Sendo assim, teremos uma representação conforme a Figura 15, a seguir.

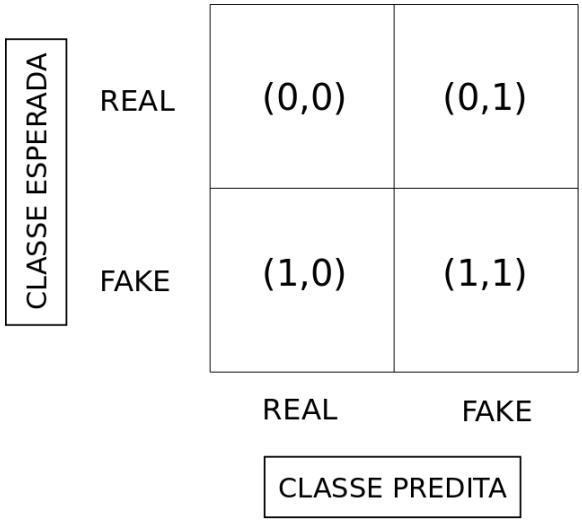

Figura 15 - Matriz de Confusão considerando as classes Real e *Fake*. Fonte: autoria própria.

O elemento da posição (0,0) da matriz representa o número de exemplos que foram classificados corretamente como reais (*TN*). O elemento (1,0) representa o número de exemplos que foram classificados como reais, mas que, na verdade, pertencem à classe falsa (*FN*). O elemento (0,1) representa o número de exemplos que foram classificados como falsos, mas que, na verdade, pertencem à classe real (*FP*). Por fim, o elemento (1,1) representa o número de exemplos que foram classificados corretamente como falsos (*TP*).

Observamos que a diagonal principal da matriz representa o número de classificações corretas, enquanto os demais elementos constituem erros.

Cada um dos elementos acima descritos recebe ainda um nome, a saber: verdadeiros positivos, falsos negativos, falsos positivos, verdadeiros negativos. Porém, como se objetivou a detecção de manipulações, a classe FAKE será considerada como positivo. Desta forma, a partir deste ponto a nomenclatura utilizada será:

- (0,0): verdadeiros negativos (*TN*);
- (1,0): falsos negativos (*FN*);
- (0,1): falsos positivos (*FP*);
- (1,1): verdadeiros positivos (*TP*).

Utilizando tais medidas, é possível ainda extrair outras métricas a partir da matriz de confusão, como a acurácia, precisão e recall.

2.7.1.3. Receiver Operator Characteristic Curve (ROC) e *Area Under the Curve (AUC)*

Análise ROC (do inglês, *Receiver Operating Characteristic*) é um modelo gráfico para avaliação, organização e seleção de sistemas de diagnóstico e/ou predição. Gráficos permitem uma melhor visualização da multidimensionalidade do problema de avaliação. O gráfico ROC (Figura 16) é baseado na probabilidade de detecção, ou taxa de verdadeiros positivos (*tpr*, do inglês *True Positive Rate*) (Equação 3) e na taxa de falsos positivos (*fpr*, do inglês *True Positive Rate*) (Equação 4). Para se construir o gráfico ROC plota-se *fpr* no eixo das ordenadas – eixo x - e *tpr* no eixo das abcissas – eixo y. (PRATI; BATISTA; MONARD, 2008)

$$tpr = \frac{TP}{TP+FN}$$

Equação 3 - Taxa de Verdadeiros Positivos (*tpr*).

$$fpr = \frac{FP}{TN+FP}$$

Equação 4 - Taxa de Falsos Positivos (*fpr*).

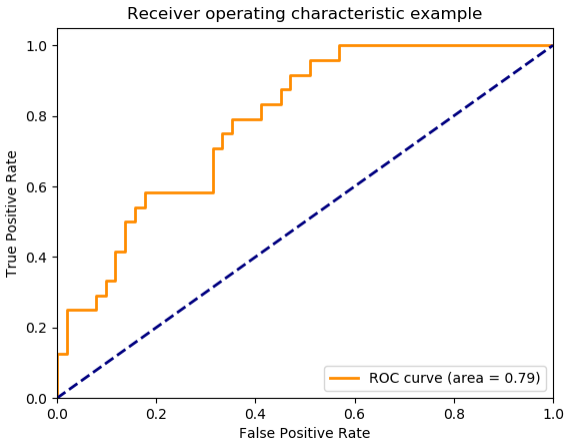

Figura 16 – Gráfico curva ROC. Fonte: <https://bit.ly/2OmnyP5>.

Cada ponto na curva ROC corresponde a um dado limiar de escolha/aceitação. Por exemplo, se a saída da rede é 0.6 para Fake, podemos usar um limiar 0.5, indicando que valores acima de 0.5 são considerados Fake, mas também podemos ser mais "exigentes" indicando que somente saídas acima de 0.8 serão consideras a saída da rede como uma indicação de Fake. A curva é construída variando-se tal limiar, de modo que a análise seja feita independentemente da escolha de um limiar específico. Ainda segundo PRATI *et al.* (2008), quanto mais distante a curva estiver da diagonal principal, melhor será o desempenho do sistema de aprendizado para aquele domínio.

Uma maneira de simplificar a análise da curva ROC a apenas um número é analisar a Área Abaixo da Curva (AUC, do inglês *Area Under Curve*). A AUC é numericamente igual à probabilidade de que, dados dois exemplos de classes distintas, o exemplo positivo seja ordenado primeiramente que um exemplo negativo. Uma vez que a área abaixo da curva ROC é uma fração da área de um quadrado de lado um, o seu valor está sempre entre 0 e 1. Quanto maior seu valor, melhor o desempenho do modelo. (PRATI; BATISTA; MONARD, 2008)

As métricas anteriormente descritas são utilizadas como forma de avaliação, em bases de dados específicas, de modelos de detecção considerados estado da arte em tarefas detecção de falsificações. Por meio desse processo são geradas as chamadas referências de desempenho da área, sendo as principais apresentadas a seguir.

2.7.2. Referências de Desempenho (*Baselines*)

Referências de desempenho representam valores atingidos por métodos considerados o estado da arte na tarefa considerada. São limiares segundo os quais novos métodos propostos são avaliados. Representam, assim, um elemento importante para a constante evolução da área de pesquisa. Neste sentido, juntamente com as bases de dados, são definidas referências de desempenho. Estas são geradas pela avaliação do desempenho de diferentes métodos quando submetidos às bases em questão. Neste projeto, a análise dos resultados sé feita considerando três destas referências.

2.7.2.1. FaceForensics

Propõem uma referência de desempenho baseada na acurácia de classificação considerando um conjunto de 1000 imagens. Aqui, não apenas é avaliada a distinção entre as classes real e *fake*, mas também entre os tipos de manipulação utilizados. A Figura 17 exibe a configuração do placar de líderes no momento da redação deste documento.

FaceForensics Benchmark

This table lists the benchmark results for the Binary Classification scenario.

Method	Info	Deepfakes	Face2Face	FaceSwap	NeuralTextures	Pristine	Total
PredictFake		0.973	0.847	0.913	0.820	0.894	**0.887**
StableForensics		**0.991**	0.847	0.951	0.787	0.884	0.883
ATDETECTOR		0.955	0.796	0.922	0.780	0.898	0.875
RobustForensics		0.973	0.832	0.942	0.760	0.854	0.859
Firefly		0.955	0.730	0.874	0.667	**0.920**	0.855

Figura 17 - Referência de Desempenho FaceForensics. Fonte: < https://bit.ly/38VwaWm>.

2.7.2.2. Celeb-DF

Utiliza como métrica o valor *AUC* (seção 2.7.1.3) em termos percentuais – *AUC (%)*. Faz uma extensa análise, considerando os principais modelos de detecção e as bases de dados atuais para pesquisa em *deepfakes*. A Tabela 3, retirada de (LI; YANG; SUN, 2019), exibe a compilação dos resultados apresentados no referenciado trabalho.

Tabela 3 - Referência de Desempenho Celeb-DF. Fonte: (LI; YANG; SUN, 2019).

Methods↓ Datasets→	UADFV [53]	DF-TIMIT [25]		FF-DF [40]	DFD [15]	DFDC [14]	**Celeb-DF**
		LQ	HQ				
Two-stream [54]	85.1	83.5	73.5	70.1	52.8	61.4	53.8
Meso4 [6]	84.3	87.8	68.4	84.7	76.0	75.3	54.8
MesoInception4	82.1	80.4	62.7	83.0	75.9	73.2	53.6
HeadPose [53]	89.0	55.1	53.2	47.3	56.1	55.9	54.6
FWA [28]	97.4	99.9	93.2	80.1	74.3	72.7	56.9
VA-MLP [33]	70.2	61.4	62.1	66.4	69.1	61.9	55.0
VA-LogReg	54.0	77.0	77.3	78.0	77.2	66.2	55.1
Xception-raw [40]	80.4	56.7	54.0	**99.7**	53.9	49.9	48.2
Xception-c23	91.2	95.9	94.4	99.7	**85.9**	72.2	65.3
Xception-c40	83.6	75.8	70.5	95.5	65.8	69.7	**65.5**
Multi-task [34]	65.8	62.2	55.3	76.3	54.1	53.6	54.3
Capsule [36]	61.3	78.4	74.4	96.6	64.0	53.3	57.5
DSP-FWA	**97.7**	**99.9**	**99.7**	93.0	81.1	**75.5**	64.6

2.7.2.3. DFDC

Utiliza como métrica de avaliação a função *LogLoss* (seção 2.7.1.1). No decorrer da competição, os participantes tinham acesso ao placar de líderes público (*Public Leaderboard* - Figura 18) e após o seu encerramento foi divulgada a versão privada do mesmo (*Private Leaderboard*) - ainda que possa não ser definitivo (QUACH, 2020), o resultado até o momento (top 5) é apresentado também na Figura 18.

Esta pode ser considerada uma referência para futuros projetos, já que simula um ambiente real de aplicação dos métodos de detecção e demonstra o quão desafiadora pode ser a tarefa de detecção de *deepfakes* – houve uma queda de desempenho considerável quando a avaliação foi realizada na base de testes privada, com o modelo vencedor atingindo 65% de acurácia na base de avaliação (VINCENT, 2020).

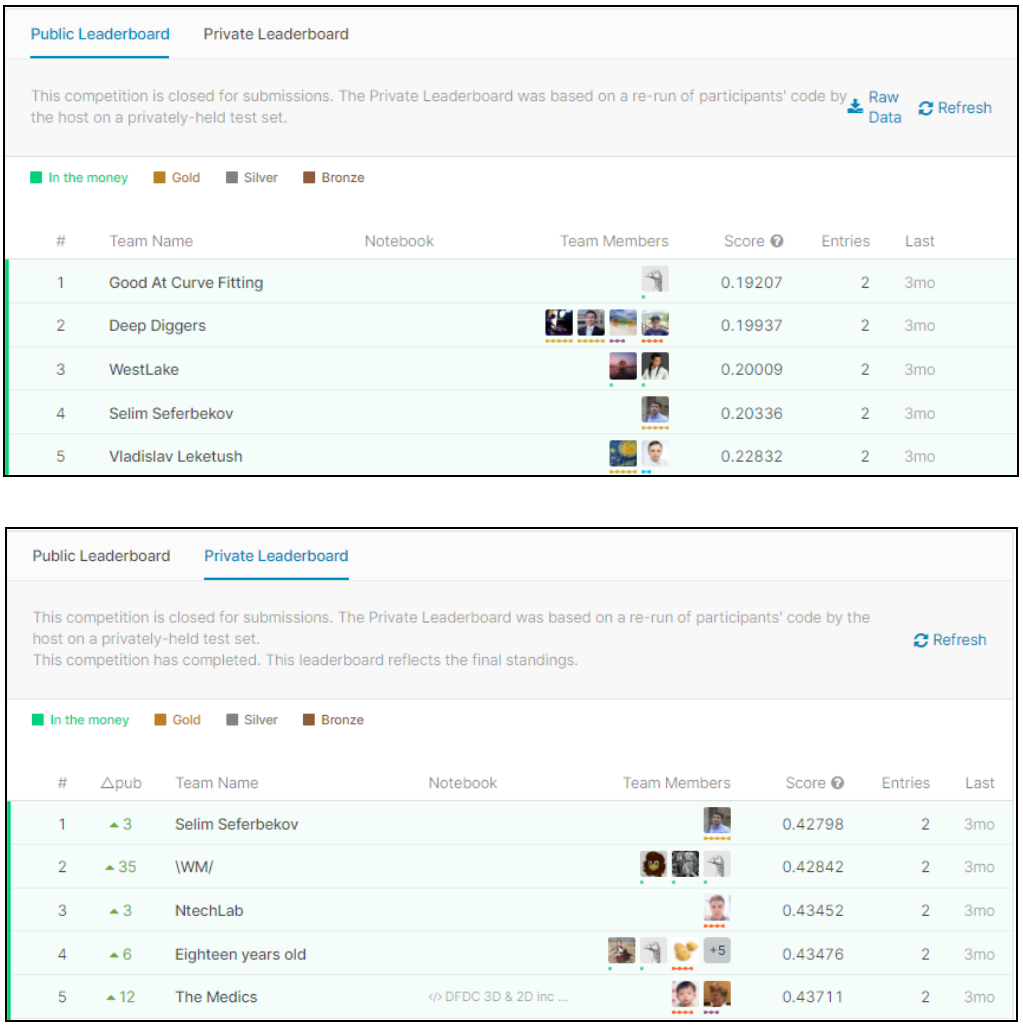

Figura 18 - Placares de Líderes Público e Privado DFDC. Fonte: <https://bit.ly/3emGGas>.

2.8. Considerações Finais

Neste capítulo foram apresentados os principais conceitos relacionados à área de detecção de Deepfakes. Inicialmente apresentou-se os conceitos básicos de redes neurais, assim como algumas das arquiteturas mais utilizadas atualmente. Posteriormente, foram discutidos os principais tipos de falsificação atuais e os métodos utilizados para gerar cada um deles. Foram apresentadas, ainda, as principais abordagens de detecção de manipulações faciais em imagens. Introduziu-se algumas das métricas mais utilizadas na avaliação de modelos de classificação. Finalmente, discorreu-se a respeito das principais

bases de dados utilizadas na área, assim como as referências de desempenho atuais da área. No capítulo seguinte, os passos de desenvolvimento do projeto são apresentados de forma detalhada.

CAPÍTULO 3: DESENVOLVIMENTO DO TRABALHO

3.1. Considerações Iniciais

Neste capítulo serão descritos os detalhes de implementação do projeto. Será descrita a metodologia adotada, assim como as ferramentas utilizadas. Posteriormente, serão apresentadas as etapas de desenvolvimento das diferentes abordagens de detecção implementadas, incluindo pré-processamento dos dados, implementação dos algoritmos, execução, compilação e análise do aprendizado adquirido pelos mesmos. Posteriormente, é proposto um sistema de classificação múltiplo composto pelos modelos com melhor desempenho individual. O desempenho do sistema é então comparado aos desempenhos individuais de seus componentes e às referências apresentadas na seção 2.7.2. Finalmente, são discutidas as principais dificuldades encontradas no decorrer do projeto, assim como suas principais limitações.

3.2. Projeto

O presente projeto tem como principal objetivo propor um método para detecção de *deepfakes*, com base nos modelos atuais de detecção. Tendo acompanhado a evolução e desfecho do desafio DFDC (SEFERBEKOV, 2020) (DAVLETSHIN, 2020), assim como após pesquisas em trabalhos atuais da área (BONETTINI *et al.*, 2020), o método de classificação escolhido foi um *ensemble* de modelos, também chamado de sistema de classificação múltiplo. Esta abordagem foi proposta visando agregar diferentes modelos, baseados em diferentes premissas, de forma a criar um modelo de caráter mais geral (independente do método de manipulação) e com melhor desempenho que seus componentes individualmente.

Primeiramente, foi realizada uma ampla pesquisa a respeito da área de detecção de falsificações em imagens e vídeos. Buscou-se, por meio de tal, selecionar um conjunto de modelos de aprendizado de máquina, baseados em *deep learning*, a serem testados no

contexto do projeto. Foram selecionados tanto modelos de detecção por quadro único, quanto modelos de detecção baseados na análise de janelas de quadros.

Posteriormente, foi realizada a busca de bases de dados destinadas à pesquisa em detecção de *deepfakes*. As bases selecionadas foram agrupadas em uma única base buscando agregar as individualidades de cada uma. Buscou-se, com isso, simular um contexto de aplicação real de um sistema de detecção de falsificações, que deve buscar ser independente do tipo e grau de falsificação, assim como do grau de compressão aplicado sobre os dados. Para isso, foram inicialmente aplicados passos de pré-processamento, utilizando a linguagem *Python*, de modo a padronizar a forma de representação dos dados nas bases.

Em seguida, os modelos de detecção incialmente selecionados foram implementados utilizando a linguagem *Python* e o *framework Keras/Tensorflow*. Cabe observar que modelos de classificação baseados no processamento de uma janela de *frames* apresenta entrada diferente daqueles baseados em quadros únicos, sendo, portanto, necessária a adequação das bases de dados para os mesmos.

Os modelos implementados foram então treinados considerando a base criada pela composição das bases individuais. O desempenho de cada um é avaliado, segundo a análise de sua matriz de confusão, a métrica *Categorical Crossentropy*, curva ROC e valor AUC, assim como a relação Acurácia *versus* Número de Parâmetros. A avaliação aqui é feita extraindo um conjunto de teste de cada base, de forma a avaliar o grau de generalização atingido.

Os 3 modelos com melhor desempenho na maioria das análises realizadas foram então agregados de modo a compor o sistema de classificação múltiplo. Seu desempenho é então comparado àqueles obtidos individualmente pelos modelos componentes do sistema.

3.3. Descrição das Atividades Realizadas

Nesta seção serão descritos, de forma detalhada, os passos elencados na seção anterior.

3.3.1. Ferramentas de Desenvolvimento

3.3.1.1. Python

Python é uma linguagem voltada para a prototipação e desenvolvimento ágil, com rápida curva de aprendizado. Vem sendo muito utilizada em diversas plataformas, para as mais variadas tarefas e por um grande número de empresas nacionais e empresas de renome mundial, como *Google* e *NASA* (LIMA, 2018). Alguns dos fatores que contribuem para o seu sucesso são a eficiência no desenvolvimento de *machine learning*, Inteligência Artificial e ciência, gestão e análise de dados.

3.3.1.2. Tensorflow e Keras

Tensorflow (TENSORFLOW, 2020) é uma biblioteca *open source* desenvolvida pelo time *Google Brain* para aprendizado de máquina. Com ela é possível criar e treinar redes neurais para as mais diversas finalidades. *Keras*, por sua vez, é um *framework* de mais alto nível, construído sobre as funcionalidades fornecidas pelo *Tensorflow*. Sendo assim, permite que um usuário sem muita experiência possa utilizar as capacidades do *Tensorflow*, porém com um nível maior de abstração.

3.3.1.3. Google Colab

Google Colab (GOOGLE, 2020) é uma ferramenta que permite escrever e executar programas *Python* no navegador. Seu funcionamento é baseado no chamado *Colab notebook*, um ambiente interativo de edição e execução de scripts *Python*. Com ele, é possível organizar código e comentários/documentação a respeito do mesmo por meio de elementos chamados células. Para utilizá-lo basta possuir e ter efetuado *login* em uma

conta *Google*. O ambiente fornece um montante razoável de memória RAM e acesso gratuito, porém limitado, a aceleradores como GPUs.

3.3.2. Pesquisa de Modelos de Detecção de Falsificações

A pesquisa em modelos de detecção/classificação de falsificações foi realizada considerando principalmente trabalhos publicados na área e as discussões realizadas pela comunidade durante a evolução do DFDC. Optou-se pela escolha dos modelos mais citados na literatura, aqueles utilizados na definição das referências de desempenho na área e os utilizados pelos participantes com melhores resultados nos placares divulgados pela organização do DFDC. Os modelos escolhidos são apresentados a seguir. São descritos de forma sucinta seus princípios de funcionamento.

MesoNet é o nome dado ao modelo proposto em AFCHAR *et al.* (2018). Modelo utilizado como referência de desempenho em diversos trabalhos na área, aborda a detecção por meio de características mesoscópicas das imagens, isto é, informações com nível semântico e estrutural médios. Desta forma, o modelo é composto por poucas camadas e apresenta treinamento rápido. Os autores ainda apresentam uma avaliação de seu desempenho em duas bases de dados, uma composta por manipulações geradas pelo método *Face2Face* e outra, esta criada pelos próprios autores, gerada com o método *Deepfake*. São relatadas acurácias acima de 90% para ambos os métodos de manipulação. O código referente ao modelo, implementado em *Keras*, está disponível em (AFCHAR, 2020).

Xception é uma arquitetura de rede recentemente proposta por François Chollet (CHOLLET, 2017), que é também o criador do *Keras*. Este modelo tem como principal característica as convoluções separáveis por profundidade (*depthwise separable convolutions*). Neste tipo de convolução, o processo é dividido em duas etapas: na primeira, cada filtro é aplicado a um canal por vez, separadamente, enquanto na segunda os resultados das multiplicações são combinados por meio de convoluções ponto a ponto (*kernels* de dimensão 1x1). Isto diminui a complexidade computacional do modelo. É utilizada como estado da arte no trabalho que introduz a base de dados *FaceForensics++*

(RÖSSLER *et al.*, 2019). O *framework Keras* disponibiliza uma implementação da Xception, pré-treinada na base ImageNet.

ResNet diz respeito a uma arquitetura proposta por HE *et al.* (2016) para reconhecimento de imagens. Sua principal característica são os blocos residuais, já explicados em seções anteriores deste trabalho. O modelo foi escolhido para teste por permitir um treinamento prolongado com risco reduzido de *overfitting*. O *framework Keras* fornece implementações dos modelos ResNet50, ResNet101 e ResNet152 (valores relacionados ao número de camadas), todos pré-treinados na base ImageNet.

EfficientNet é um dos modelos mais recentes na área de classificação (MINGXING; QUOC, 2019). A principal proposta dos autores é estudar o balanceamento da profundidade, largura e resolução quando da escalabilidade do modelo, de modo a levar a um desempenho ótimo. É proposta uma família de modelos, de nomes B0 a B7, a depender de seu fator de escala. Tanto o *Keras* quanto os próprios autores fornecem implementações do modelo. Além de ser um modelo atual e com desempenho considerado estado da arte em tarefas de classificação, o mesmo foi utilizado nas soluções do primeiro e segundo colocados do DFDC. No projeto, foi utilizado o modelo EfficientNetB4, por apresentar uma boa relação entre o número de parâmetros e o nível de acurácia, e por ser implementado no trabalho que motiva o uso de um sistema de classificação múltipla como detector final neste projeto (BONETTINI *et al.*, 2020).

Os modelos a seguir abordam o problema da detecção de *deepfakes* com um processamento por janela de quadros.

GÜERA *et al*. (2018) propõe um sistema de classificação de falsificações que utiliza uma CNN para extrair características a nível de quadro. Posteriormente, tais características são utilizadas para treinar uma RNN (LSTM) de forma a extrair características temporais relacionadas aos quadros em sequência, utilizando-as, finalmente, para a classificação da mesma. Os autores avaliam ainda a influência do tamanho da sequência de entrada (largura da janela de quadros) no desempenho do modelo.

De forma semelhante, SABIR *et al.* (2019) realiza uma análise extensa do desempenho de uma estrutura composta por um modelo extrator de características a nível

de quadro (CNN) e um modelo extrator de características temporais (RNN), considerando a tarefa de classificação de manipulações geradas pelos métodos *Deepfake*, *Face2Face* e *FaceSwap*. Os autores analisam diversas configurações para o modelo CNN e RNN, mantendo, porém, a célula GRU como componente fixo do modelo RNN. São relatadas acurácias acima de 95%.

Os últimos modelos selecionados são apresentados em DOGONADZE *et al* (2020). Seguindo basicamente a mesma configuração dos modelos anteriores, os autores propõem um extrator de características espaciais por meio de uma CNN e duas configurações de extratores de características temporais: uma rede composta por convoluções 3D e uma rede LSTM. A classificação de um dado quadro é feita considerando também os quadros de sua vizinhança. A base de dados utilizada é a *FaceForensics++*, tendo o modelo proposto figurado entre os primeiros colocados da referência de desempenho definida por tal base.

3.3.3. Pesquisa de Bases de Dados de *Deepfakes*

Assim como foi feito com relação aos modelos selecionados, a pesquisa por bases de dados relacionadas às *deepfakes* foi feita pelo estudo das principais publicações na área, assim como pelo acompanhamento da discussão no DFDC. Após tal pesquisa, as bases de dados já descritas na seção 2.6 foram selecionadas.

3.3.4. Pré-processamento dos Dados

As bases de dados utilizadas, exceto a referente à MesoNet, são originalmente compostas por arquivos de vídeo, com aproximadamente 10s de duração cada. Os modelos de classificação, porém, realizam o processamento baseado em imagens. Além disso, como demonstrado em RÖSSLER *et al.* (2019), a alimentação dos modelos com apenas a região de interesse, no caso as faces, melhora o desempenho geral. Desta forma, a primeira etapa de processamento dos dados dizia respeito à separação dos vídeos em *frames* – a uma taxa

de 3 *fps* - e então a extração dos rostos presentes nestes. Para isso foram usadas as bibliotecas OpenCV[1] e Face Recognition[2].

A primeira se trata de uma biblioteca de código aberto para operações em imagens e vídeos. A segunda se trata de uma biblioteca para reconhecimento de faces: dada a imagem de uma face, o retorno do processamento são pontos chave presentes no rosto, assim como os limites da região retangular que contém a face em questão – região de interesse (ROI, do inglês *Region of Interest*). No contexto do projeto, foram utilizados apenas a informação referente à ROI, à qual foi adicionada um fator de aumento, de modo a capturar a cabeça inteira, e não apenas o rosto da pessoa. Isso foi feito pois alguns artefatos resultantes do processo de manipulação podem estar presentes em regiões fora da área da face, como na divisão do rosto com o cabelo, como ilustrado em WEST *et al*. (2019) (Figura 19).

Figura 19 - Artefatos gerados pelo processo de falsificação fora da região do rosto. Fonte: (WEST; BERGSTROM, 2019).

Posteriormente, foi realizada uma etapa de filtragem manual das faces extraídas, de modo a remover possíveis falhas de detecção, como exemplificado na Figura 20, a seguir.

[1] Disponível em: <https://docs.opencv.org/master/d6/d00/tutorial_py_root.html>. Acessado em 12/07/2020 às 18:04.

[2] Disponível em: <https://pypi.org/project/face-recognition/>. Acessado em 12/07/2020 às 18:04.

Figura 20 - Falhas resultantes da detecção de faces. Fonte: autoria própria.

Finalmente, os dados foram organizados em uma estrutura de diretórios de modo a permitir o carregamento correto das bases para treinamento e teste dos modelos. O processo descrito é ilustrado na figura 21, a seguir.

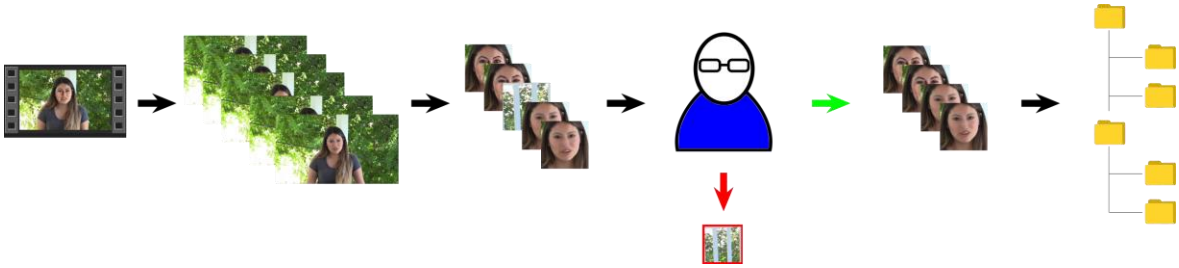

Figura 21 - Resumo dos passos de pré-processamento das bases de dados. Fonte: autoria própria.

3.3.5. Implementação dos Modelos de Detecção

Como já mencionado, a implementação dos modelos selecionados se deu no ambiente do *Google Colab* e foi realizada na linguagem Python, utilizando as facilidades proporcionadas pela biblioteca de aprendizado de máquina *Keras*. A organização do projeto foi realizada utilizando o *Google Drive*, por permitir uma interação direta com o *Google Colab*. Desta forma, tanto as bases de dados, quanto os arquivos referentes à estruturação dos modelos e às funcionalidades gerais do projeto foram mantidos na plataforma online.

3.3.6. Treinamento dos Modelos de Detecção

Os parâmetros utilizados no treinamento de cada um dos modelos são elencados a seguir, na Tabela 4. Em todos os casos, foi utilizado o otimizador Adam (KINGMA; BA, 2015).

Tabela 4 - Parâmetros de treinamento dos modelos implementados.

Modelo	*MesoNet*	*Xception*	*ResNets*	*EfficientNet*	*DenseNet+GRU, InceptionV3+LSTM, InceptionV2+3DResnet*
Taxa de aprendizado inicial	1e-3	1e-3	1e-3	1e-3	3e-3
Tamanho de Batch	76	32	32	32	8
Dimensões da Entrada	256x256x3	256x256x3	224x224x3	224x224x3	160x160x3
Largura Janela de Quadros	X	X	X	X	5

As bases de dados são mantidas em disco de forma separada, sendo então reunidas no momento do treinamento dos modelos. Uma porcentagem de cada base é utilizada para compor a base de treinamento, conforme a distribuição exibida nas Figuras 23 e 24.

Cabe observar que, na abordagem de detecção por análise de janela de *frames*, cada exemplo consiste em uma sequência de, no caso, cinco *frames*. A saída esperada para cada exemplo corresponde à classificação do *frame* intermediário da sequência. Desta forma, a classificação de um dado quadro é realizada considerando dois quadros anteriores e dois quadros posteriores ao mesmo, ou ainda, dois quadros no "passado" e dois quadros no "futuro" (Figura 22).

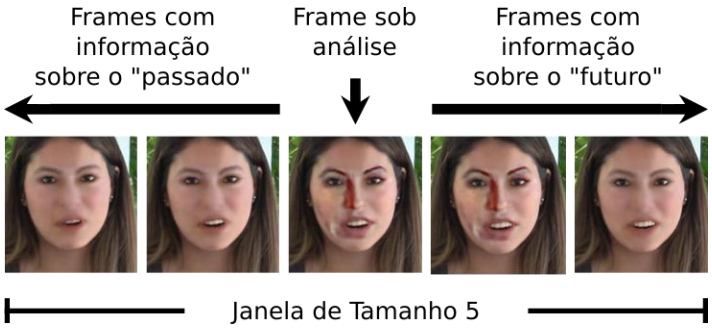

Figura 22 – Ilustração de um exemplo do conjunto de dados utilizado na abordagem de detecção por análise de janela de quadros. Fonte: autoria própria.

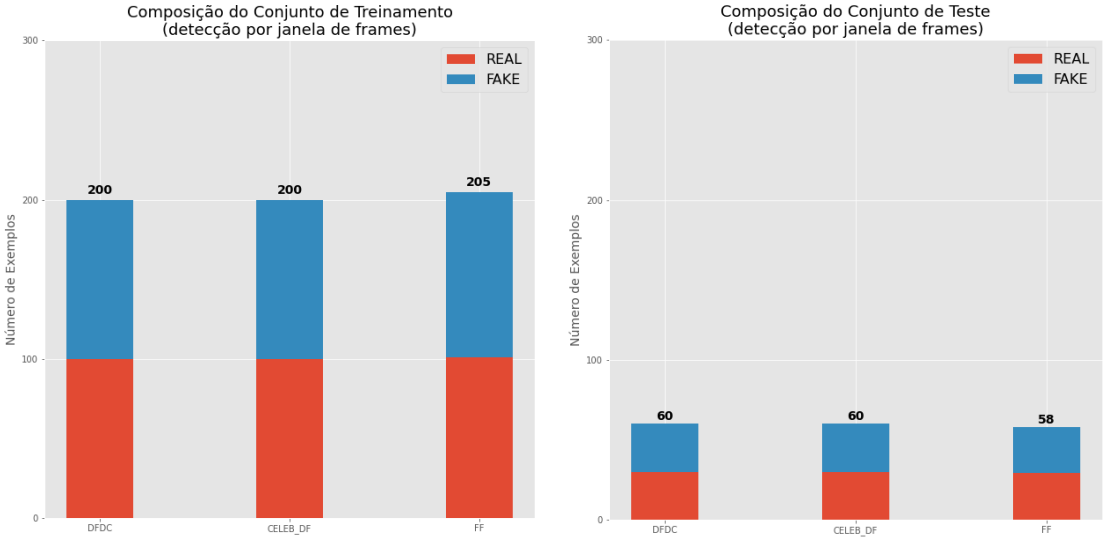

Figura 23 - Número de dados dos conjuntos de treinamento e teste para a abordagem de detecção por janela de *frames*.

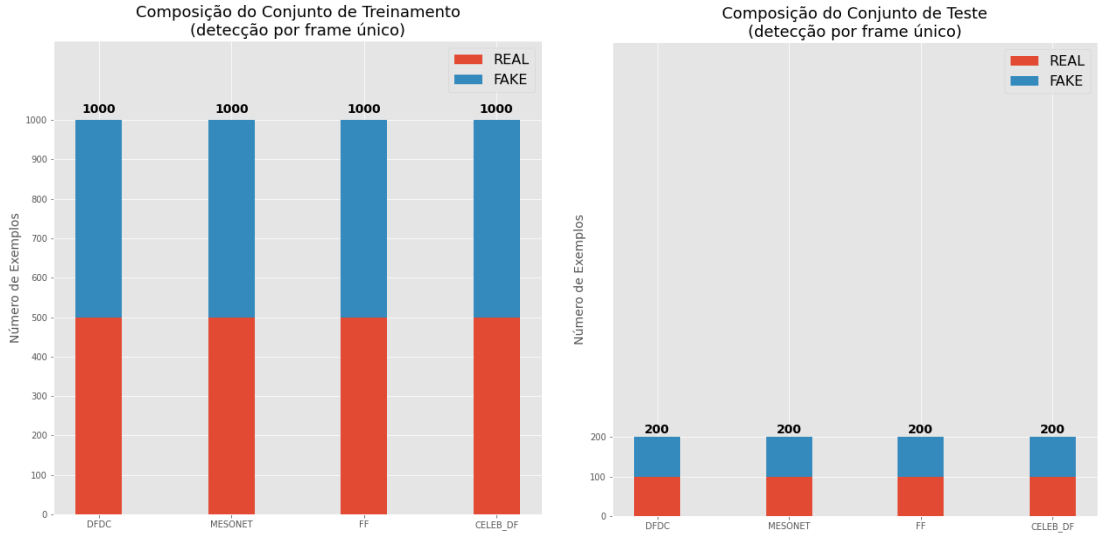

Figura 24 - Número de dados dos conjuntos de treinamento e teste para a abordagem de detecção por *frame* único.

O modelo MesoNet foi treinado com pesos inicializados aleatoriamente. Todos os demais foram treinados utilizando *transfer learning*. Nestes, os pesos utilizados são provenientes do pré-treinamento na base de dados ImageNet. A evolução do aprendizado neural de cada um dos modelos, com relação aos seus valores de acurácia e perda, é exibida a seguir, nas Figuras 25 e 26.

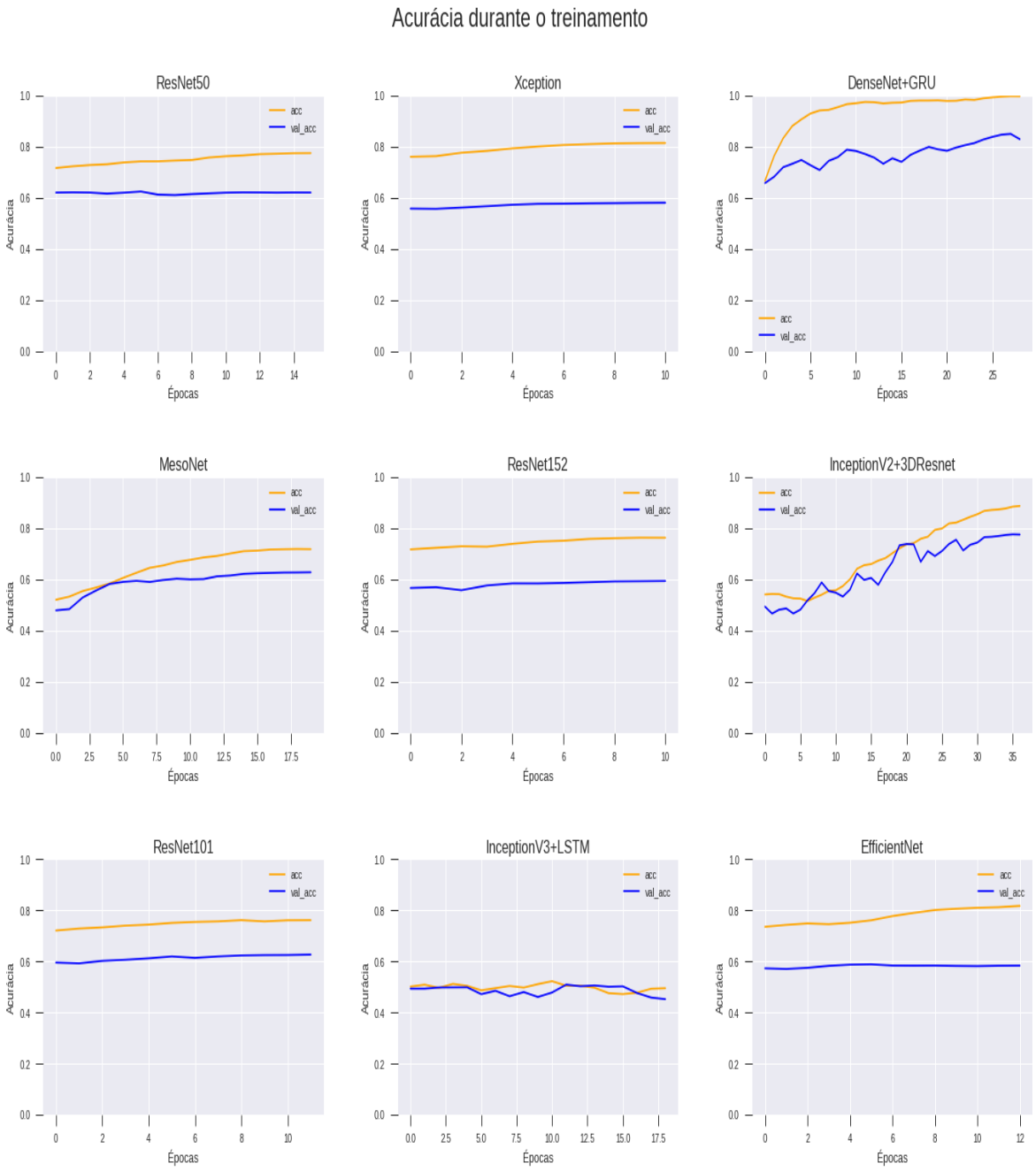

Figura 25 - Evolução do aprendizado neural medido pelo nível de acurácia de cada um dos modelos treinados.

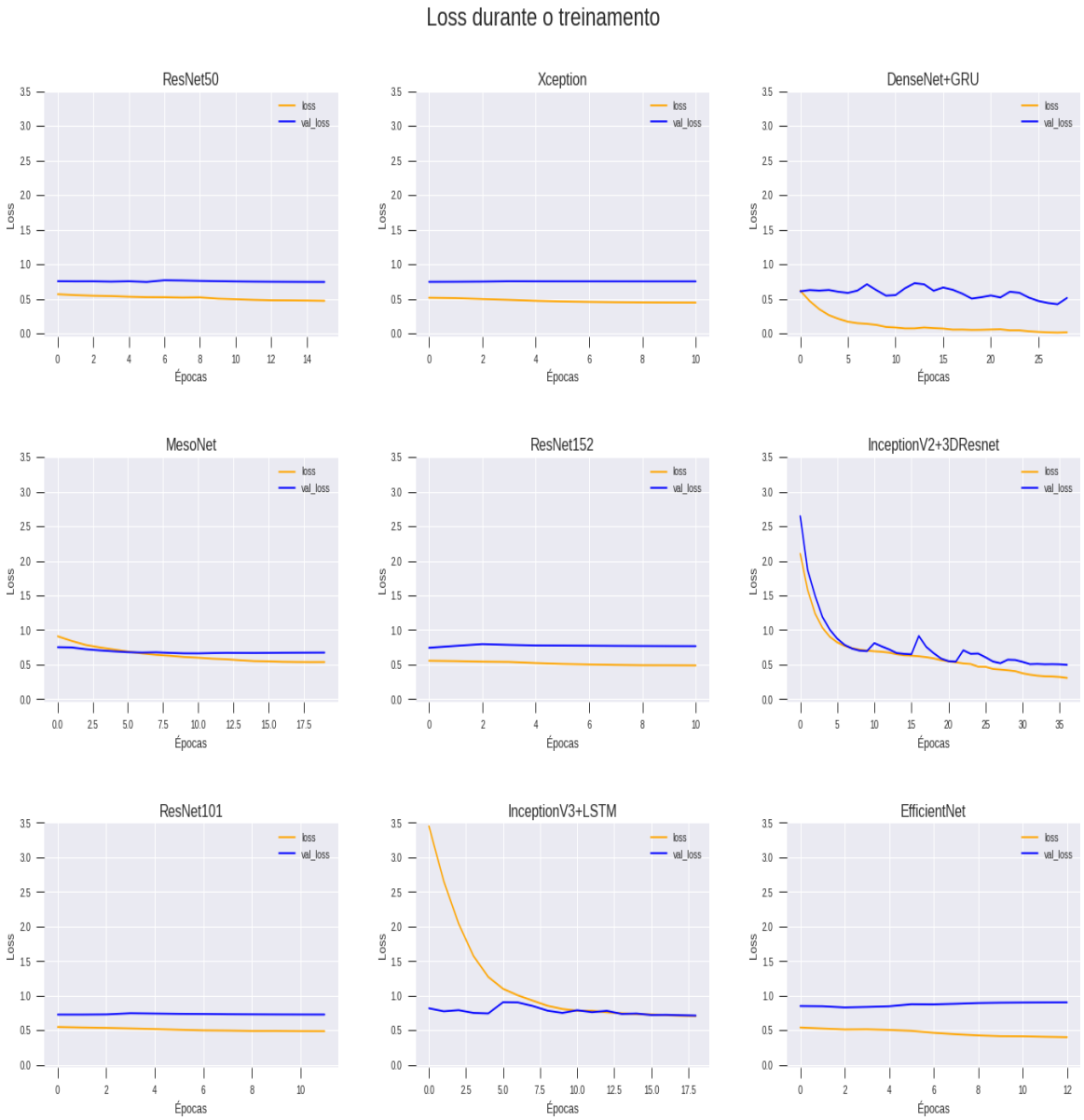

Figura 26 - Evolução do aprendizado neural medido pelo valor de perda de cada um dos modelos treinados.

Podemos observar que a maior parte dos modelos não obteve ganhos significativos de desempenho durante o treinamento. O modelo com variações mais consideráveis foi o composto pelas redes InceptionV2 e 3DResnet. Em seguida, vem os modelos DenseNet+GRU e Mesonet, porém com taxa de aprendizado bem inferior à apresentada pelo primeiro modelo.

A observação apenas da evolução do aprendizado neural, contudo, não dá uma visão tão clara do real desempenho dos modelos. Dessa forma, como auxílio à interpretação dos resultados, seguem as matrizes de confusão e curvas ROC dos modelos testados (Figuras 27 e 28).

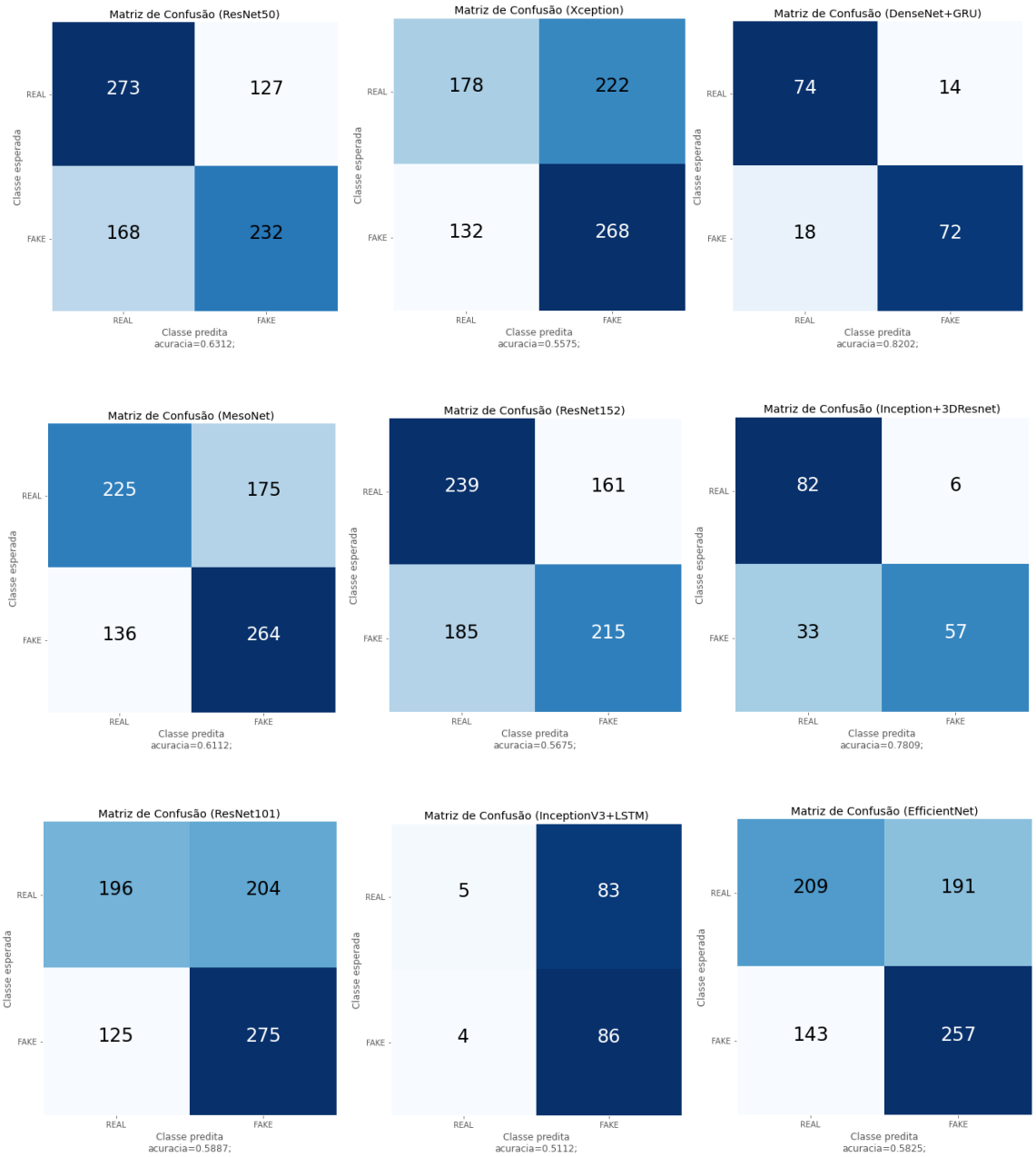

Figura 27 - Matrizes de confusão dos modelos treinados.

A análise da Figura 27 elucida melhor o comportamento do modelo, quando comparada à curva de aprendizado. Os modelos que havíamos observado ter o melhor comportamento pela análise da evolução do aprendizado confirmaram tal hipótese pela matriz de confusão – houve apenas uma troca entre as posições dos métodos de detecção por janela de *frames*. Fica claro, porém, que o modelo InceptionV3+LSTM não foi capaz de aprender com o treinamento, tendo comportamento similar "ao de um chute", sempre apostando em classificar as imagens como *fake*. Os modelos ResNet se mantiveram entre estes dois extremos.

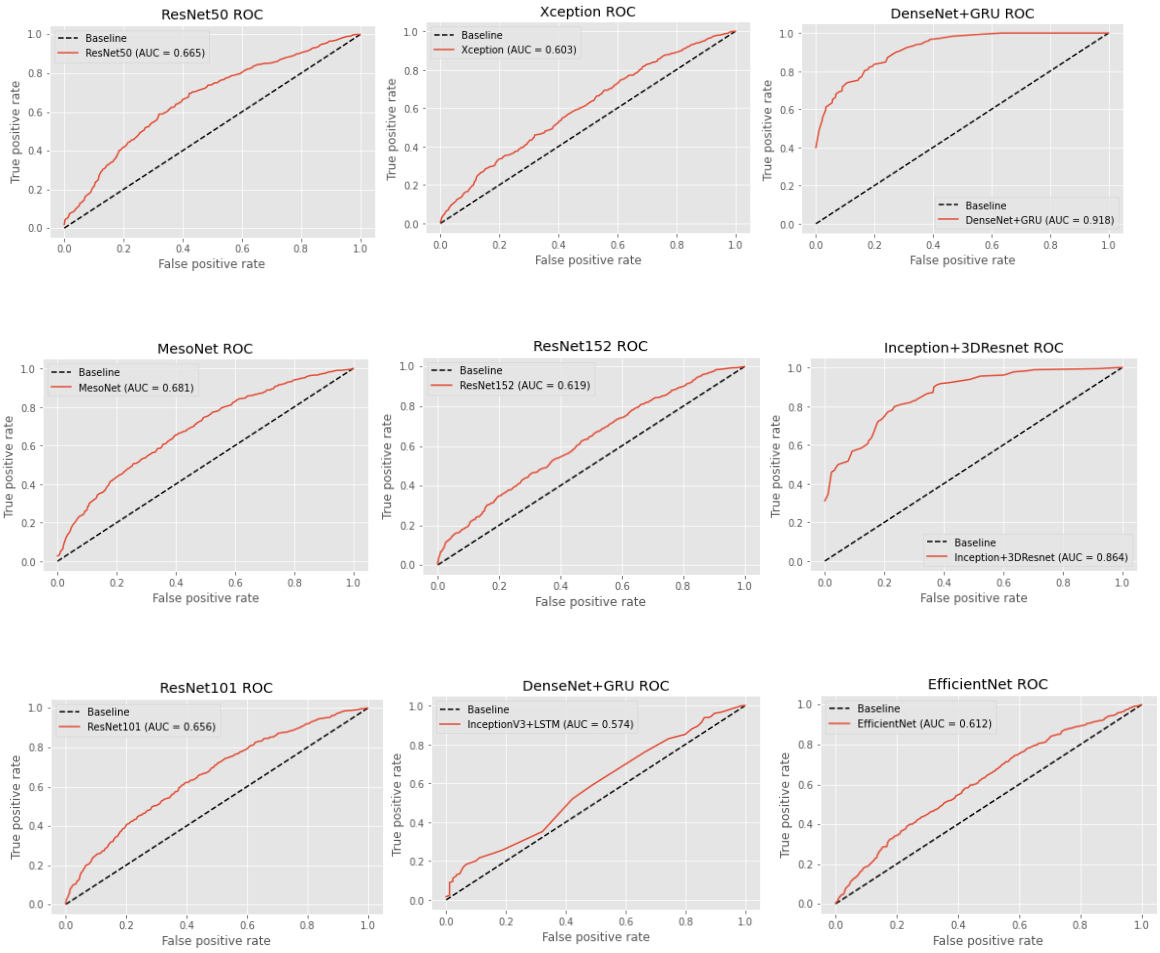

Figura 28- Curvas ROC e valores AUC para os modelos treinados.

A análise da curva ROC parece ser igualmente desafiadora e um pouco abstrata quando comparada à matriz de confusão, que fornece uma medida direta do desempenho dos modelos. Contudo, por uma rápida análise das curvas ROC apresentadas na Figura 28,

podemos constatar que, como nos casos anteriores, os modelos InceptionV2+3DResnet e DenseNet+GRU apresentam melhor desempenho, enquanto o modelo composto pelas redes InceptionV3+LSTM apresentam o pior resultado. Os demais casos podem ser avaliados considerando o valor AUC, que auxilia na distinção entre os desempenhos dos modelos intermediários, já que suas curvas ROC são visualmente muito próximas.

Finalmente, para avaliar a influência das bases de dados no desempenho dos modelos, são exibidos os níveis de acurácia para cara uma das bases utilizadas no treinamento, assim como o valor obtido pelo teste na base conjunta (Figura 29).

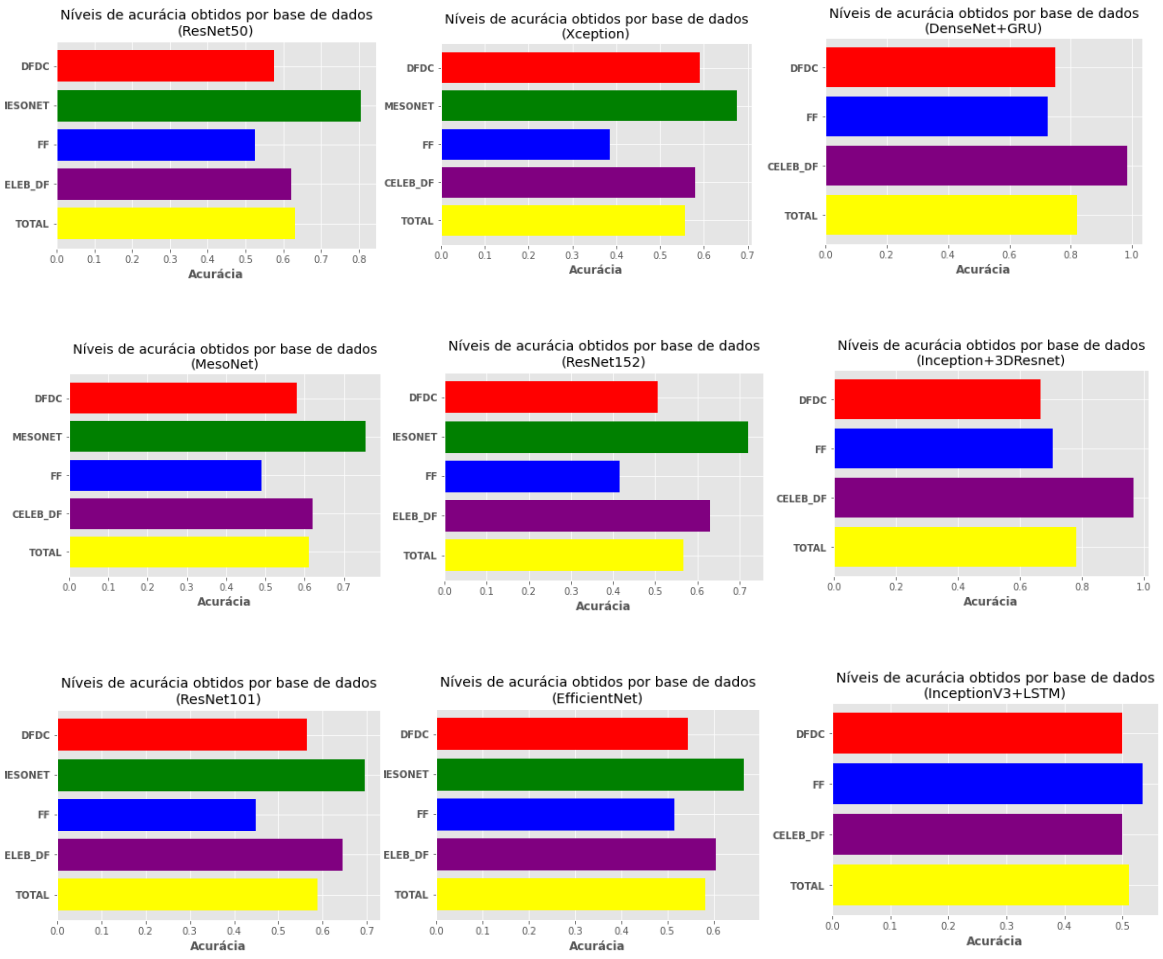

Figura 29 - Níveis de acurácia por base de dados.

Como podemos observar, pela análise da Figura 29, os modelos baseados na detecção por análise de *frame* único apresentam o maior nível de acurácia quanto realizando predições a respeito da base de dados MesoNet. Modelos baseados na análise de

um conjunto de *frames* apresentam melhor desempenho na base de dados Celeb-DF – base esta que resulta no segundo melhor desempenho entre os modelos de classificação por *frame* único. De forma geral, os métodos não conseguiram uma boa generalização para a base de dados *FaceForensics*.

Cabe observar que não foi possível utilizar a base referente à MesoNet para a detecção por conjunto de *frames*. Por isso ela não aparece nos gráficos referentes a tais métodos de detecção.

3.3.7. Composição do Sistema de Detecção Múltipla

Visando compor um sistema de detecção múltiplo, os modelos anteriormente analisados foram comparados segundo seu nível de acurácia/perda *versus* seu número de parâmetros (Figuras 30 e 31). Por fim, foram compiladas também as acurácias obtidas a partir da avaliação dos modelos (tanto o sistema de classificação múltiplo, quanto seus componentes individualmente) nos conjuntos de teste referentes a cada base de dados (Figura 33).

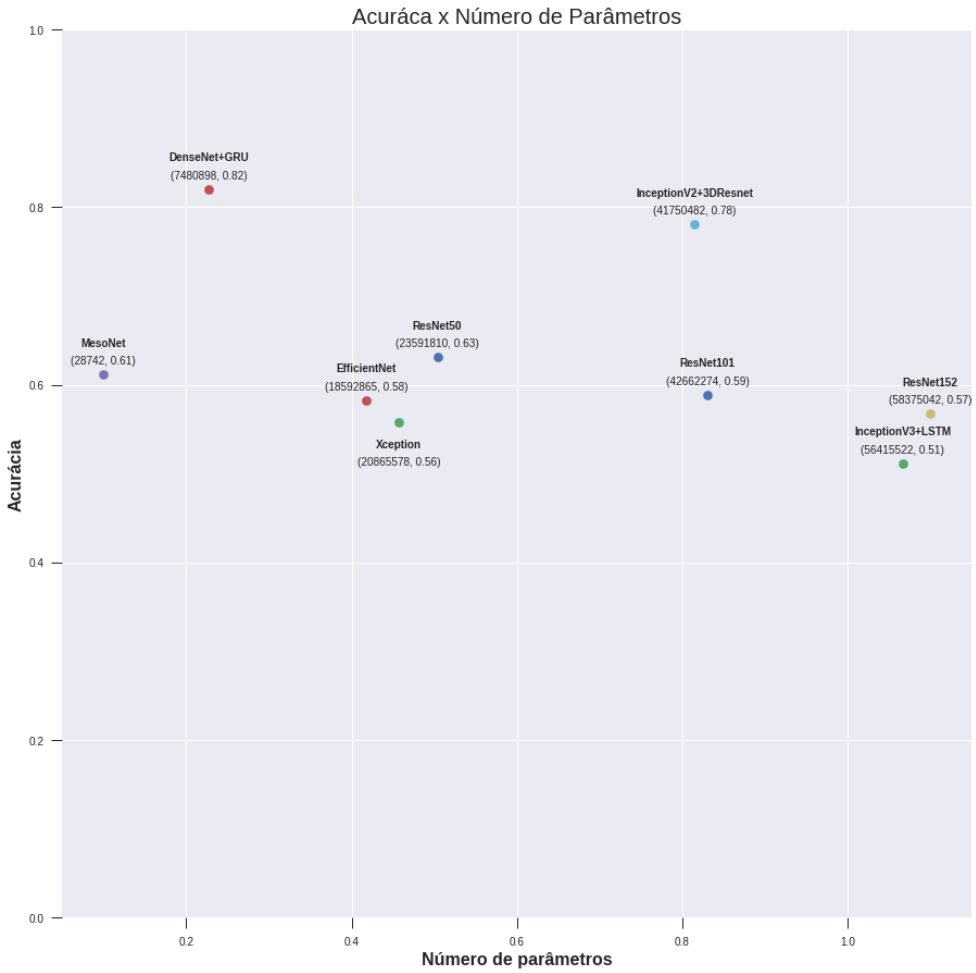

Figura 30 - Acurácia versus Número de Parâmetros.

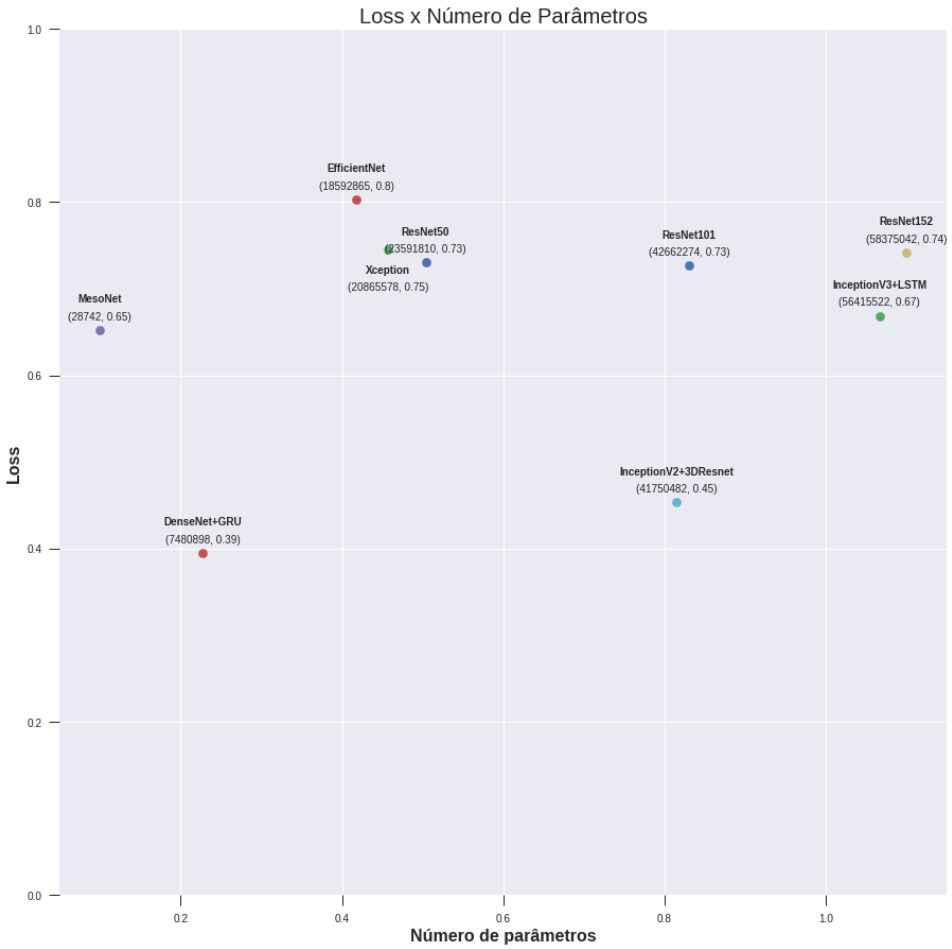

Figura 31 - Perda (Loss) versus Número de Parâmetros.

A interpretação dos gráficos de Acurácia e Loss *versus* o Número de Parâmetros é ilustrada na Figura 32. Os quadrantes verdes correspondem às regiões onde os melhores modelos se encontram alocados, enquanto os quadrantes vermelhos correspondem aos piores casos em cada um dos quesitos considerados. Isto é, no primeiro caso, os modelos com melhor desempenho são aqueles com um número pequeno de parâmetros e alta acurácia. No segundo caso, continuam sendo desejados modelos com menos parâmetros e também com as menores perdas.

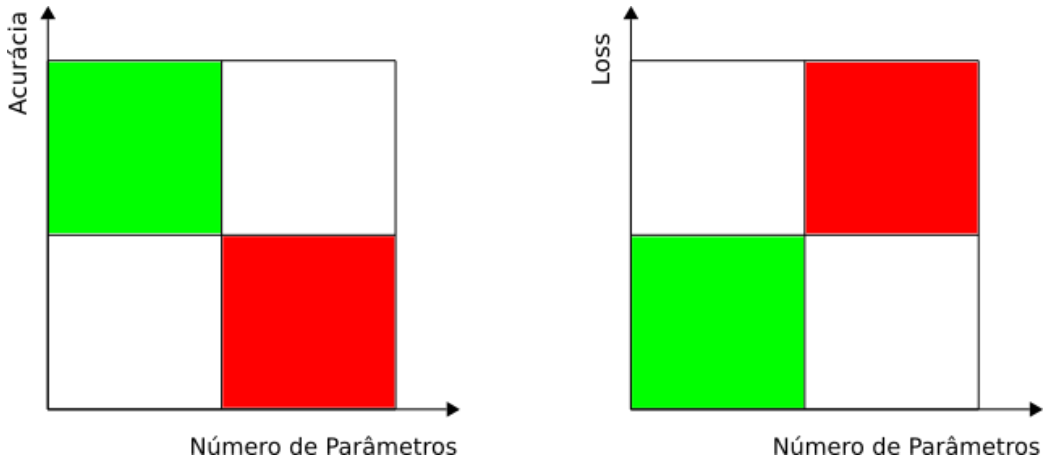

Figura 32 - Interpretação para as figuras 30 e 31. Fonte: autoria própria.

Tendo por base as métricas apresentadas, e as análises realizadas, foram selecionados os três modelos com o melhor desempenho, a saber: DenseNet+GRU, MesoNet e EfficientNet. Contudo, o primeiro modelo apresentou problemas quando de sua recuperação para uso em novas classificações. Dessa forma, foi escolhido o modelo InceptionV2+3DResnet como seu substituto. A escolha foi baseada no desempenho deste modelo nas análises anteriores, e também na abordagem utilizada pelo mesmo, já que é muito semelhante ao modelo substituído. Desta forma, foi criado um sistema com, além da diversidade de modelos, diversidade de estratégias.

As predições dos modelos foram combinadas de forma simples e direta, por um sistema de voto em que a classe com mais votos, ou seja, mais predita, é a classe escolhida como classificação final. Optou-se por um número ímpar de modelos para que não houvesse o problema do empate na classificação (número igual de votos para real e *fake*).

3.4. Resultados Obtidos

Os resultados obtidos pelo modelo de classificação são apresentados na Figura 33. Observa-se que o nível de acurácia atingido pelo modelo conjunto (“*ensemble*” em amarelo na Figura 33) não supera aquele do modelo com melhor desempenho entre os três. Desta forma, seria mais vantajoso utilizar apenas um dos modelos, aquele com os melhores resultados.

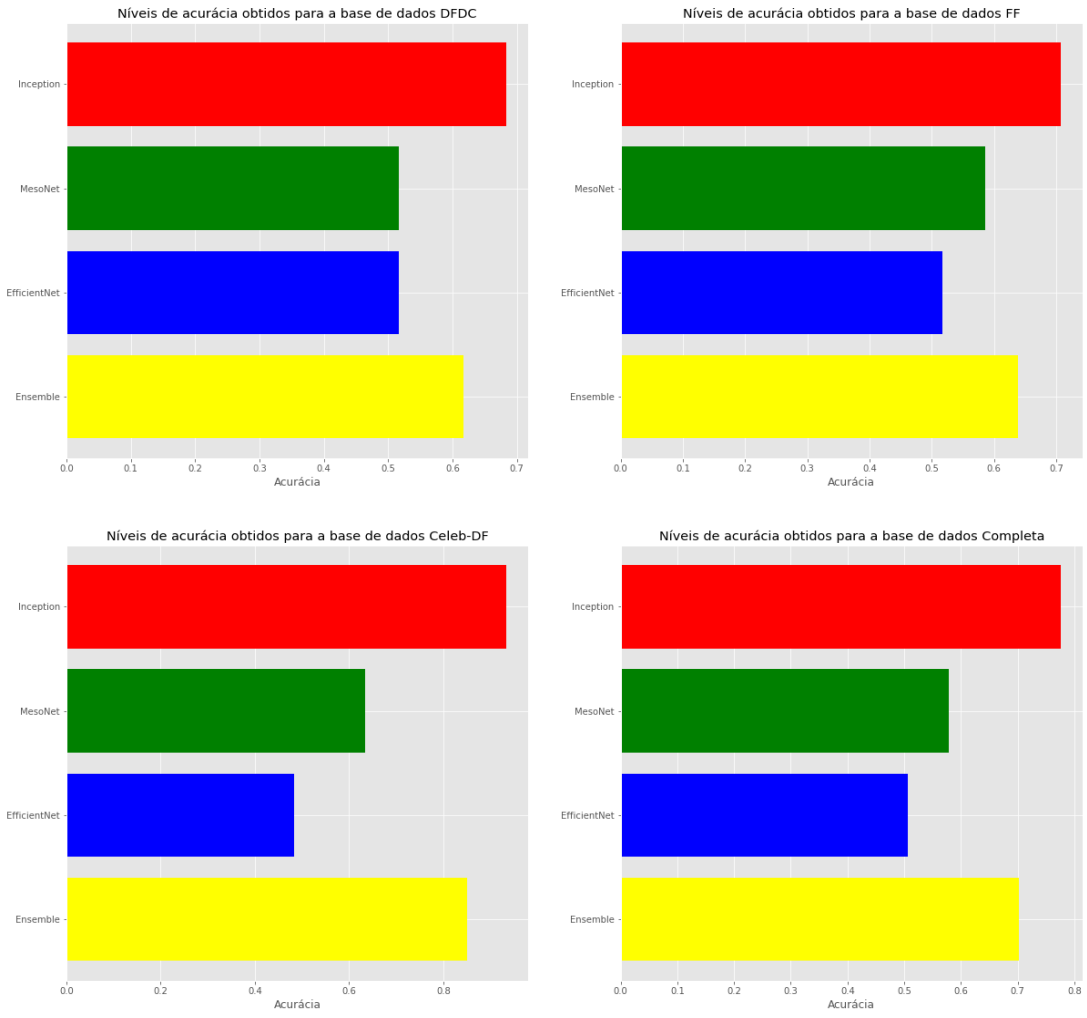

Figura 33 - Comparação dos resultados obtidos pelo sistema de classificação em diferentes bases de dados.

Como pudemos observar durante as análises anteriores, nenhum dos modelos implementados obteve resultados comparáveis aos da referência definida para a base *FaceForensics* (Figura 17), que são próximos aos 90%. Quando consideramos a referência de desempenho para a base *Celeb-DF* (Tabela 3, última coluna), observamos que obtivemos resultados muito superiores aos relatados, principalmente para os métodos baseados na análise de janelas de *frames*, que atingiram até 91.8 de valor AUC (%) (Figura 28), contra os 65.5 relatados na referência. Com relação à referência para o DFDC (Figura 18), obtivemos resultados próximos a 0.5 ou superiores para o valor de *loss* (“perda” ou “erro”) (Figura 26). Estes valores podem ser considerados distantes dos apresentados pela melhor solução, já que mesmo o modelo implementado que apresentou o melhor

desempenho não entraria no “top 5” da referência, que tem uma variação muito pequena entre os valores alcançados pelo primeiro e quinto colocados.

Para os casos em que os modelos não apresentaram resultados satisfatórios, há algumas possíveis explicações. Primeiramente, conforme mencionado por (LI, 2020), os dados presentes na maioria das bases referentes à detecção de *deepfakes* não condiz com um cenário real de aplicação. Os autores alegam que artefatos como os presentes em muitos dos vídeos fornecidos durante o DFDC são provenientes de manipulações mal produzidas, e que são facilmente identificáveis. Quando voltamos nosso olhar às manipulações que realmente circulam na internet e que podem oferecer algum tipo de perigo, nos deparamos com falsificações quase indistinguíveis de seus pares reais. Desta forma, a presença de artefatos facilmente distinguíveis pode ter afetado a capacidade de generalização para imagens com falsificações de melhor qualidade.

Um segundo fator que pode influenciar no nível de acerto dos modelos é a compressão dos dados, conforme discutido em RÖSSLER *et al.* (2019).

Um terceiro fator é o fato de que a base de dados fornecida pelo DFDC continha não só manipulações nos rostos, mas também na fala. Como este tipo de análise não foi abordado pelo presente projeto, pode ser que tenha havido perdas relacionadas a este fato.

Finalmente, um último fator que pode ter afetado o desempenho das análises realizadas é referente aos recursos de desenvolvimento limitados. As equipes com melhores desempenhos na competição do *Facebook* relataram até mesmo dias de processamento em múltiplas GPUs. E mesmo assim, conseguiram resultados próximos a 70% de precisão (VINCENT, 2020).

Desta forma, é importante destacar que, apesar das limitações, os resultados alcançados neste projeto também se situam próximos de 70% na base DFDC (Figura 33), e a disponibilidade de melhores recursos poderia ter contribuído para a sua melhoria.

Uma última análise a ser feita é que os melhores resultados, disparadamente (Figuras 27, 28 e 29), foram obtidos por métodos baseados na análise de um conjunto de *frames*, mesmo com um conjunto de treinamento consideravelmente menor. Sendo assim,

fica evidente a importância da análise da coerência e características temporais quando da detecção de *deepfakes*.

3.5. Dificuldades e Limitações

As principais dificuldades encontradas no decorrer do projeto estão relacionadas principalmente à implementação e treinamento dos modelos de detecção.

Um primeiro fator a se considerar aqui é que a maioria dos trabalhos da área apresentam apenas uma descrição concisa da estrutura do modelo proposto, sem o código fonte associado ao mesmo. Quando o fazem, as implementações raramente são realizadas com o uso do *Tensorflow/Keras*. Este processo de implementação a partir de uma descrição resumida ou mesmo a partir da tradução de uma biblioteca para outra representou uma grande dificuldade, já que muitos trabalhos tratavam de conceitos e arquiteturas que não eram familiares. Além disso, para realizar a tradução entre bibliotecas, foi necessário um estudo dos códigos para compreender as funcionalidades e então realizar sua implementação em *Keras*.

Um outro ponto a se destacar ainda dentro da implementação e execução dos modelos, é a ferramenta utilizada para tal. O *Google Colab* (ou *Colaboratory*) é uma ferramenta extremamente poderosa e que facilita a implementação e execução em *Python/Tensorflow/Keras*, fornecendo também o acesso à aceleração por GPUs. Contudo, há um limite máximo de uso (GPU, CPU, Memória). Quando tal limite é atingido, é necessário aguardar um período até que os recursos estejam novamente disponíveis. Isto representou também uma dificuldade no projeto, já que sem o acesso a GPUs o treinamento dos modelos leva um tempo extremamente elevado.

Um outro ponto é referente à manipulação das bases de dados. Por serem compostas por imagens e vídeos, seu tamanho em disco é elevado, atingindo facilmente a ordem de GigaBytes. O pré-processamento foi todo realizado em uma máquina própria e, devido a limitações da mesma, no caso um notebook, o processo de extração das faces das imagens, assim como organização dos arquivos levou um tempo considerável.

Com relação ao estudo e compilação inicial dos trabalhos na área, apesar de ser um tema atual, há muito material a respeito do mesmo. A compilação dos principais resultados e métodos mais promissores representou também uma dificuldade. Aqui, acredito que um melhor planejamento e uma seleção mais criteriosa dos trabalhos mais relevantes poderia ter poupado um tempo considerável. Soma-se a isto o fato de que, por este ser um tema novo também para o autor, houve um momento de embasamento nos principais conceitos da área, que consumiu um tempo acima do esperado. Por outro lado, obteve-se um grande aprendizado estudando os diversos modelos e técnicas abordadas neste trabalho.

Uma limitação do método proposto é seu alto tempo de execução. Uma vez selecionado um vídeo, é necessário realizar a extração de seus *frames* e então das faces presentes nos mesmos, para então realizar a classificação por meio do sistema múltiplo. Desta forma, pensando em um cenário real de aplicação, o método proposto teria de ser utilizado de forma *offline*, ou seja, em aplicações que não exigem processamento em tempo real.

3.6. Considerações Finais

Neste capítulo foram apresentados todos os passos realizados durante o desenvolvimento do projeto. Descreveu-se a etapa de pesquisa de modelos e bases relacionadas à detecção de *deepfakes*, o pipeline de pré-processamento dos dados, a estruturação do projeto e as estratégias de treinamento e avaliação de modelos empregadas. Foi proposto um novo modelo de classificação múltipla, composto pelos três modelos com melhores desempenhos individuais e foi avaliado o novo desempenho obtido. Finalmente, foram discutidas as dificuldades encontradas no decorrer do projeto, assim como as principais limitações do modelo proposto e do método adotado, apresentando também as lições aprendidas.

O capítulo seguinte, último deste trabalho, apresenta as conclusões a respeito do projeto, destacando suas contribuições e apresentando uma visão a respeito do futuro das pesquisas no tema abordado, assim como possibilidades de trabalhos futuros.

CAPÍTULO 4: CONCLUSÃO

4.1. Contribuições

O presente projeto deixa como principais contribuições um estudo introdutório de um tema extremamente recente e de extrema importância, com possíveis impactos diretos em nosso dia a dia. Além de serem apresentadas as técnicas de falsificação de faces em imagens e vídeos, são introduzidos e implementados alguns dos métodos mais recentes de detecção de tais manipulações. É realizada uma análise de tais métodos segundo critérios amplamente utilizados quando tratando de sistemas de classificação, sendo os com melhores desempenhos reunidos em um sistema de classificação múltipla. A análise dos resultados, que por um lado deixa evidente o quão desafiador o tema da detecção de falsificações ainda é, demonstra que a análise de características temporais é uma abordagem promissora para futuros estudos na área. Finalmente, discute-se a respeito dos principais fatores limitadores da pesquisa atual na área.

4.2. Trabalhos Futuros

Modelos de detecção de *deepfakes* baseados em *deep learning*, apesar de seu grande poder, apresentam como principal limitação o fato de ser um modelo do tipo "caixa preta", o que torna difícil sua explicabilidade. Ou seja, o conhecimento adquirido durante o treinamento do modelo fica armazenado na forma de pesos em suas conexões; desta forma, a interpretabilidade do conhecimento adquirido é difícil de ser realizada por seres humanos. Sendo assim, uma preocupação crescente no ramo de pesquisa em redes neurais é quanto à questão de sua interpretabilidade.

No caso de detecção de *deepfakes*, diversas pesquisas vêm sendo feitas neste sentido. AFCHAR *et al.* (2018) apresenta uma análise baseada nos mapas de características obtidos em determinadas camadas convolucionais. NGUYEN *et al.* (2019) aborda a questão implementando uma rede baseada em aprendizado multitarefas (do inglês, *multi-task learning*), em que a saída é composta tanto pela classificação da imagem quanto por sua segmentação, isto é, uma máscara com a localização dos possíveis artefatos

identificados. BONETTINI *et al.* (2020) segue a mesma linha, porém implementa um mecanismo de atenção para segmentar regiões distintivas das imagens manipuladas. Por fim, LI *et al.* (2019) propõe um modelo que busca determinar o contorno da região manipulada, assumindo como premissa o processo de sobreposição que diversos algoritmos de manipulação usam quando da falsificação de rostos.

Sendo assim, uma possibilidade de extensão do presente projeto é no sentido de facilitar a interpretabilidade de seus resultados.

Uma outra possibilidade de trabalho futuro seria o estudo da influência da forma de representação dos dados no desempenho da rede. AMERINI *et al.* (2019) apresenta uma aplicação de fluxos ópticos no auxílio à detecção de falsificações. De forma semelhante, ZHANG *et al.* (2019) discute a possibilidade do uso das representações das imagens no domínio da frequência como forma de revelar artefatos introduzidos no processo de sintetização de *deepfakes* por meio de GANs.

Por fim, como descrito na seção 3.5, uma das principais limitações do projeto diz respeito ao tempo elevado de execução para cada detecção realizada. Assim, uma possibilidade de extensão do presente projeto é o estudo de novos métodos e novas arquiteturas de rede na busca de melhoria da velocidade de processamento, como, por exemplo, as redes YOLO (REDMON *et al.*, 2016) e SSD (LIU *et al.*, 2016), redes que apresentam processamento extremamente rápido.

REFERÊNCIAS BIBLIOGRÁFICAS

AFCHAR, D. MesoNet. **GitHub**, 2020. Disponivel em: <https://github.com/DariusAf/MesoNet>. Acesso em: 15 Março 2020.

AFCHAR, D. et al. MesoNet: a Compact Facial Video Forgery Detection Network. **2018 IEEE International Workshop on Information Forensics and Security (WIFS)**, Hong Kong, 11-13 Dezembro 2018. 1.

AMERINI, I. et al. Deepfake Video Detection through Optical Flow Based CNN. **2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)**, Seul, 27-28 Outubro 2019.

AVERBUCH-ELOR, H. et al. Bringing portraits to life. **ACM Transactions on Graphics**, Novembro 2017.

BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA., R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 39, n. 12, p. 2481-2495, Dezembro 2017.

BARNI, M. et al. Aligned and Non-Aligned Double JPEG Detection Using Convolutional Neural Networks. **Journal of Visual Communication and Image Representation** , Agosto 2017.

BONDI, L. et al. Tampering Detection and Localization Through Clustering of Camera-Based CNN Features. **2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)**, Honolulu, 21-26 Julho 2017. 185-1864.

BONETTINI, N. et al. Video Face Manipulation Detection Through Ensemble of CNNs, Abril 2020.

CHO, K. et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, Junho 2014.

CHOI, Y. et al. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**, Salt Lake City, 18-23 Junho 2018.

CHOLLET, F. Xception: Deep Learning with Depthwise Separable Convolutions. **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, Honolulu, 21-26 Juho 2017.

DANG, H. et al. On the Detection of Digital Face Manipulation, Outubro 2019.

DAVLETSHIN, A. Deepfake Detection Challenge - Discussion: 3rd place solution and code. **Kaggle**, 2020. Disponivel em: <https://www.kaggle.com/c/deepfake-detection-challenge/discussion/158158>. Acesso em: 12 Julho 2020.

DENG, J. et al. ImageNet: A large-scale hierarchical image database. **2009 IEEE Conference on Computer Vision and Pattern Recognition**, Miami, 20-25 Junho 2009.

DOGONADZE, N.; OBERNOSTERER, J.; HOU., J. Deep Face Forgery Detection, Abril 2020.

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

FACEAPP. **FaceApp**, 2017. Disponivel em: <https://faceapp.com/app>. Acesso em: 10 Julho 2020.

FACEBOOK. Home. **Deepfake Detection Challenge (DFDC)**, 2019. Disponivel em: <https://deepfakedetectionchallenge.ai/>. Acesso em: 10 Julho 2020.

FACEFORENSICS Benchmark. **FaceForensics Benchmark**, 2020. Disponivel em: <http://kaldir.vc.in.tum.de/faceforensics_benchmark/index.php>. Acesso em: 10 Julho 2020.

FARID, H. Image Forgery Detection: A survey. **IEEE SIGNAL PROCESSING MAGAZINE**, p. 1-25, Março 2019.

FERRARA, P. et al. Image Forgery Localization via Fine-Grained Analysis of CFA Artifacts. **IEEE Transactions on Information Forensics and Security**, v. 7, n. 5, p. 1566-1577, Outubro 2012. ISSN 1556-6021.

FRITH, C. D. Role of facial expressions in social interactions. **Philosophical Transactions of the Royal Society B: Biological Sciences**, Dezembro 2009. 3453-3458.

GOODFELLOW, I. J. et al. Generative Adversarial Nets. **NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems**, Montreal, v. II, p. 2672–2680, 10 Dezembro 2014.

GOOGLE. Welcome To Colaboratory. **Colab Research**, 2020. Disponivel em: <https://colab.research.google.com/notebooks/intro.ipynb>. Acesso em: 12 Julho 2020.

GÜERA, D.; DELP., E. J. Deepfake video detection using recurrent neural networks. **2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)**, Auckland, 27-30 Novembro 2018.

HE, K. et al. Deep Residual Learning for Image Recognition. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, Las Vegas, 23-30 Junho 2016.

HOCHREITER, S. Long Short-term Memory. **Neural Computation**, Dezembro 1997.

ISOLA, P. et al. Image-to-Image Translation with Conditional Adversarial Networks. **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, Honolulu, 21-26 Julho 2017. 5967-5976.

JULLIAND, T.; NOZICK, V.; TALBOT., H. Image Noise and Digital Image Forensics. In: SHI, Y.-Q., et al. **Digital-Forensics and Watermarking:** 14th International Workshop. Tóquio: [s.n.], 2016. p. 3-17.

KAGGLE. Deepfake Detection Challenge. **Kaggle**, 2019. Disponivel em: <https://www.kaggle.com/c/deepfake-detection-challenge/overview>. Acesso em: 10 Julho 2020.

KARRAS, T.; LAINE, S.; ALLA., T. A Style-Based Generator Architecture for Generative Adversarial Networks. **2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, Long Beach, 15-20 Junho 2019.

KINGMA, D. P.; BA., J. Adam: A Method for Stochastic Optimization. **3rd International Conference for Learning Representations**, San Diego, 7-9 Maio 2015.

KOWALSKI, M. FaceSwap. **GitHub**, 2018. Disponivel em: <https://github.com/MarekKowalski/FaceSwap/>. Acesso em: 10 Julho 2020.

LI, L. et al. Face X-ray for More General Face Forgery Detection, Dezembro 2019.

LI, Y. Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics. **GitHub**, 2020. Disponivel em: <https://github.com/danmohaha/celeb-deepfakeforensics>. Acesso em: 12 jul. 2020.

LI, Y.; CHANG, M.; LYU., S. In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking. **2018 IEEE International Workshop on Information Forensics and Security (WIFS)**, Hong Kong, 11-13 Dezembro 2018.

LI, Y.; YANG, X.; SUN, P. Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics; Honggang Qi; Siwei Lyu., 2019.

LIMA, V. 7 Razões para aprender Python. **School of Net**, 2018. Disponivel em: <https://blog.schoolofnet.com/7-razoes-para-aprender-python/#:~:text=Al%C3%A9m%20da%20Google%2C%20outras%20grandes,e%20at%C3%A9%20mesmo%20a%20NASA.>. Acesso em: 12 Julho 2020.

LIU, W. et al. SSD: Single Shot MultiBox Detector. **Computer Vision – ECCV 2016: 14th European Conference**, Amsterdam, 11-14 Outubro 2016. 21-37.

LOUBAK, A. L. Aplicativo Zao usa deepfake para criar vídeos e viraliza na China. **TechTudo**, 2019. Disponivel em: <https://www.techtudo.com.br/noticias/2019/09/aplicativo-zao-usa-deepfake-para-criar-videos-e-viraliza-na-china.ghtml>. Acesso em: 10 Julho 2020.

MARTIN, S. What Is Transfer Learning? **Nvidia**, 2019. Disponivel em: <https://blogs.nvidia.com/blog/2019/02/07/what-is-transfer-learning/>. Acesso em: 11 Julho 2020.

MCCLOSKEY, S.; ALBRIGHT., M. Detecting GAN-generated Imagery using Color Cues, 19 Dezembro 2018.

MINGXING, T.; QUOC, L. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, 2019.

MONARD, M. C.; BARANAUSKAS., J. A. Conceitos sobre Aprendizado de Máquina. In: REZENDE, S. O. . M. M. C. C. A. C. P. L. **Sistemas Inteligentes para Engenharia**. Belo Horizonte: Editora UFMG, 1999. Cap. 4, p. 39-56.

NGUYEN, H. H. et al. Multi-task Learning For Detecting and Segmenting Manipulated Facial Images and Videos, Junho 2019.

NGUYEN, T. et al. Deep Learning for Deepfakes Creation and Detection, 5 Setembro 2019. 5.

OLAH, C. Understanding LSTM Networks. **colah's blog**, 2015. Disponivel em: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Acesso em: 11 Julho 2020.

PEROV, I. DeepFaceLab. **GitHub**, 2018. Disponivel em: <https://github.com/iperov/DeepFaceLab>. Acesso em: 07 dez. 2020.

PRATI, R. C.; BATISTA, G. E. D. A. P. A.; MONARD., M. C. Curvas ROC para avaliação de classificadores. **IEEE Latin America Transactions**, 2008. 215-222.

QUACH, K. Facebook's $500k deepfake-detector AI contest drama: Winning team disqualified on burried consent technically. **The Register**, 2020. Disponivel em: <https://www.theregister.com/2020/06/18/facebook_deepfake_kaggle_contest/>. Acesso em: 23 Junho 2020.

REDMON, J. et al. You Only Look Once: Unified, Real-Time Object Detection. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, Las Vegas, NV, 27-30 Junho 2016. 779-788.

RÖSSLER, A. et al. FaceForensics: A Large-scale Video Dataset for Forgery Detection in Human Faces, Março 2018.

RÖSSLER, A. et al. FaceForensics++: Learning to Detect Manipulated Facial Images, 26 Agosto 2019.

SABIR, E. et al. Recurrent Convolutional Strategies for Face Manipulation Detection in Videos, 16 Maio 2019.

SARTORI, B. Bruno Sartori. **Youtube**, 2012. Disponivel em: <https://www.youtube.com/channel/UCaiGLmKrcve_cipRElqSnqA>. Acesso em: 11 jul. 2020.

SEFERBEKOV, S. Deepfake Detection Challenge - Discussion: 1st place solution. **Kaggle**, 2020. Disponivel em: <https://www.kaggle.com/c/deepfake-detection-challenge/discussion/145721>. Acesso em: 12 Julho 2020.

STAMM, M. C.; WU, M. Information Forensics: An Overview of the First Decade. **IEEE Access**, v. 1, p. 167-200, Maio 2013. ISSN 2169-3536.

TENSORFLOW. Home. **Tensorflow**, 2020. Disponivel em: <https://www.tensorflow.org/>. Acesso em: 12 Julho 2020.

THIES, J. et al. Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, Las Vegas, 27-30 Junho 2016.

THIES, J.; ZOLLHÖFER, M.; NIEßNER., M. Deferred Neural Rendering: Image Synthesis using Neural Textures. **SIGGRAPH 2019**, Los Angeles, 28-1 Julho 2019.

TOLOSANA, R. et al. DeepFakes and Beyond: A Survey of Face Manipulation and Fake Detection. **Journal of Latex Class Files**, Madri, v. XIII, n. 9, 1 Março 2016.

TRAN, D. et al. Learning Spatiotemporal Features with 3D Convolutional Networks. **2015 IEEE International Conference on Computer Vision (ICCV)**, Santiago, 7-13 Dezembro 2015.

VENKATACHALAM, M. Recurrent Neural Networks. **Towards Data Science**, 2019. Disponivel em: <https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce>. Acesso em: 11 Julho 2020.

VINCENT, J. Facebook contest reveals deepfake detection is still and 'unsolved problem'. **The Verge**, 2020. Disponivel em: <https://www.theverge.com/21289164/facebook-deepfake-detection-challenge-unsolved-problem-ai>. Acesso em: 23 Junho 2020.

WANG, P. This Person does not Exist. **This Person does not Exist**, 2019. Disponivel em: <https://thispersondoesnotexist.com/>. Acesso em: 11 jul. 2020.

WEST, J.; BERGSTROM, C. Home. **Which Face is Real?**, 2019. Disponivel em: <http://www.whichfaceisreal.com/index.php>. Acesso em: 11 jul. 2020.

YANG, X.; LI, Y.; LYU., S. Exposing Deep Fakes Using Inconsistent Head Poses. **ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**, Brighton, 12-17 Maio 2019. 8261-8265.

ZHANG, X.; KARAMAN, S.; CHANG., S.-F. Detecting and Simulating Artifacts in GAN Fake Images. **2019 IEEE International Workshop on Information Forensics and Security (WIFS)**, Delft, 9-12 Dezembro 2019.

ZHU, J.-Y. et al. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. **2017 IEEE International Conference on Computer Vision (ICCV)**, Veneza, 22-29 Outubro 2017.