UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

ALEXANDRE KOCHENBORGER DUARTE

Detecção e Identificação de Pessoas em Imagens com Borramento Causado por Movimento

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Cláudio Rosito Jung

Porto Alegre
2024

“The greatest triumphs of science are born out of the struggles and failures of countless experiments.”

— MARIE CURIE

AGRADECIMENTOS

Agradeço à minha família por estar ao meu lado durante toda esta longa jornada. Agradeço à equipe de professores e funcionários da UFRGS que permitem a minha formação e a de todos os colegas com os quais dividi inúmeros momentos e alegrias e tristezas. Agradeço ao profº Cláudio Jung, orientador deste trabalho, por me dar a oportunidade de me aprofundar um pouco mais na área de inteligência artificial, um assunto extremamente importante e relevante nos dias hoje.

RESUMO

Nos dias de hoje, é comum pessoas registrarem momentos curiosos enquanto se movimentam, como acidentes e assaltos, e às vezes podem capturar, mesmo que acidentalmente, pessoas que podem ser de interesse investigativo. Em vários casos, a captura fica borrada devido ao movimento relativo entre a câmera e a pessoa. O avanço da tecnologia, em especial na área de visão computacional, tem se mostrado promissor para tarefas como reconhecimento facial, usando técnicas tipicamente baseadas em redes neurais convolucionais por se demonstrarem mais eficientes que os métodos clássicos. Porém, um borramento causado por movimento (*motion blur*) se apresenta como um desafio, uma vez que um rosto pode não ser detectado devido ao efeito de deformação e embaçamento causado pelo fenômeno, fazendo a detecção desse rosto também um passo essencial para o reconhecimento. Este trabalho propõe um estudo sobre o impacto que técnicas de tratamento de imagem que amezinam o efeito do *motion blur* (*deblurring*) têm na eficácia tanto na detecção quanto no reconhecimento facial. Além do *deblurring*, também são testadas algumas modificações nos resultados obtidos pela rede detecção facial e também nos dados de entrada (*inputs*) da rede de reconhecimento, sendo que ambos se demonstraram eficientes na melhora do resultado final. A combinação de *deblurring* com essas modificações geraram os melhores resultados. O tratamento das imagens melhorou significativamente a detecção e o reconhecimento facial, e o aumento da área da detecção também apresentou melhora consideravel no reconhecimento.

Palavras-chave: Borramento por movimento. *motion blur*. tratamento de imagem. *deblurring*. redes neurais. CNN. inteligência artificial. *deep learning*. reconhecimento facial. detecção facial.

Detection and Identification of People in Images affected by Motion Blur

ABSTRACT

In this day and age, it's common people recording curious moments while they are moving, like accidents and robberies, and sometimes they can capture, even if by accident, people that might be of investigative interest. In several cases, the image frame can get blurred due to the relative movement between the camera and the person. The advancement of the technology, specially in the computer vision field, have been proving itself promising for tasks such as facial recognition, using techniques typically based on convolutional neural networks as they show themselves to be more efficient than classic methods. However, motion blur presents itself as a challenge, as a face may not be detected due to the deformation and blurring effect caused by the phenomenon, making detection also an essential step towards recognition. This work proposes a study on the impact that image processing techniques that reduce the motion blur effect (deblurring) have on the effectiveness of both facial detection and recognition. In addition to the deblurring, some modifications are also tested in the results obtained by the facial detection network and also in the input data of the recognition network, both of which improved the final result. The combination of deblurring with those modifications had the best results.The deblurring of the images improved significantly the detection and the facial recognition, and the increase of the face detection area also improved the recognition significantly.

Keywords: motion blur, image treatment, deblurring, neural networks, CNN, artificial intelligence, deep learning, facial recognition, facial detection.

LISTA DE ABREVIATURAS E SIGLAS

ML	*Machine Learning*
DL	*Deep Learning*
TL	*Transfer Learning*
GPU	*Graphic Processing Unit*
PSF	*Point Spread Function*
SAM	*SegmentAnything*
PSNR	*Peak Signal-to-Noise Ratio*
SSIM	*Structural Similarity Index Measure*
TF	*TensorFlow*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

Nos dias de hoje, vivemos cercados por tecnologia das mais variadas formas. Entre as mais comuns e acessíveis, estão as câmeras, comumente encontradas em celulares, capazes de registrar momentos tanto em fotografias como em vídeos. Por qualquer motivo, durante o registro do momento, é possível que os objetos ou pessoas de interesse estejam em movimento. Seja qual for a câmera usada, existe um tempo de exposição, isto é, um tempo que o sensor da câmera utiliza para capturar a luz e registrá-la de forma digital como um arquivo de vídeo ou de imagem (BEN-EZRA, 2015). Movimentos que ocorram durante esse tempo de exposição irão fazer com que a imagem seja registrada em pontos diferentes em relação ao ponto inicialmente detectado, criando partes enevoadas e esmaecidas ao longo de um trajeto. Esse efeito é conhecido como borramento por movimento, ou *motion blur*.

Cabe notar que existem diversos tipos de borramento, como borramento de foco, ou *out-of-focus blur*, que ocorre quando os objetos não estão na distância focal da câmera. Cada tipo de borramento é tratado de uma maneira diferente (por ter uma formulação matemática distinta), e este trabalho focará somente em borramentos causados por movimento. Mais especificamente, este trabalho focará no tratamento de imagens com rostos humanos. A remoção do borramento é conhecida como *deblurring* (ZHANG et al., 2022).

É sabido que métodos de *deblurring* de propósito geral não atingem o melhor resultado possível em rostos. De acordo com Wang, Xu and Zheng (2023), rostos possuem menos bordas que uma imagem com vários objetos ou um cenário no fundo, por exemplo. E também, rostos possuem algumas características específicas, como o formato da boca e dos olhos, que podem ser usadas como *prior* (ANWAR; HUYNH; PORIKLI, 2018) a fim de melhorar o tratamento da imagem.

Embora o problema de *deblurring* venha sendo estudado há vários anos (PAN; Z; Z, 2014; HACOHEN; E; D., 2013), técnicas recentes baseadas em *deep learning* têm atingido melhores resultados, como (KARAALI; JUNG, 2022) e (ZENG; DIAO, 2020) . Essa qualidade pode ser medida através de métricas de comparação entre imagens,

como *Peak Signal-to-Noise Ratio* (PSNR) e *Structural Similarity Index Measure* (SSIM) (HORé; ZIOU, 2010). O uso de *deep learning* facilita a estimação do kernel de borramento, assim como permite outras abordagens, os chamados métodos fim-a-fim, que não tentam estimar o *kernel* mesmo quando é desconhecido. Embora estimar o *kernel* seja factível sem o uso de *deep learning*, os resultados costumam ser mais deteriorados. O *deblurring* pode ser resolvido por dois tipos de técnicas, chamadas de *blind deblurring* (XUE, 2022), onde o kernel de borramento é desconhecido, e *non-blind deblurring* (DASGUPTA, 2022), quando é conhecido. Naturalmente, um kernel desconhecido é um problema a mais a ser resolvido, tornando o *blind deblurring* uma tarefa mais desafiadora. Neste trabalho, são testados resultados de tratamentos por ambos os métodos.

1.2 Objetivos

Este estudo focará no impacto que técnicas de *deblurring* de borramento causado por movimento possuem no reconhecimento facial baseado em *deep learning*, ou seja, *deblurring* de imagens rostos humanos e sua influência na precisão da análise de uma rede neural. Dada a ausência de datasets para reconhecimento facial com imagens borradas, este trabalho irá explorar o borramento sintético de faces nítidas através de operações de convolução. Em seguida, as imagens borradas serão tratadas com alguns modelos existentes de *deblurring*, e as imagens processadas serão analisadas por algoritmos de detecção e reconhecimento facial. Como objetivo secundário, será avaliado o impacto do borramento e *deblurring* na área do rosto detectado, e o efeito resultante no reconhecimento.

1.3 Estrutura

O restante do trabalho é dividido em mais 4 capítulos. O capítulo 2 apresenta todo o embasamento teórico e técnicas que foram utilizadas para realização dos experimentos. O capítulo 3 apresenta a metodologia utilizada para estes experimentos, bem como detalhes da geração e obtenção da base de dados utilizada. O capítulo 4 mostra os resultados dos diferentes experimentos e conclusões sobre cada um. O capítulo 5 resume o trabalho realizado, as conclusões principais obtidas e possibilidades de continuidade do estudo.

2 REFERENCIAL TEÓRICO E TRABALHOS RELACIONADOS

2.1 Aprendizado de Máquina

O termo “inteligência artificial” (IA) foi primeiramente definido pelo professor John McCarthy como a “ciência e engenharia que fazem máquinas inteligentes”, em 1997 (MCCARTH, 1997). O termo “aprendizado de máquina”, ou *machine learning* (ML), se refere a uma área do campo da inteligência artificial que atribui a programas a habilidade de analisar uma base de dados e usar essa experiência para tomar decisões quando fornecidos dados não ainda analisados, ou seja, executar ações sem receber instruções específicas e melhorar decisões futuras (KULIN TARIK KAZAZ; MOERMAN, 2021). Existem dois tipos principais de *machine learning*: supervisionado (LIU; WU, 2012) e não-supervisionado (USAMA et al., 2019).

O aprendizado supervisionado faz uso de entradas se sabendo quais serão as respectivas saídas, sendo ambos relacionados através de um rótulo de referência para ensinar ao algoritmo como prever futuras saídas para entradas que não fizeram parte da base de dados do treinamento. Por exemplo, este trabalho faz uso de uma rede neural treinada com imagens de rostos de uma pessoa, e imagens diferentes da mesma pessoa são utilizadas como novos *inputs* a fim de avaliar o desempenho da rede de determinar quem é a pessoa desses *inputs*. Detalhes serão vistos na seção 2.2.2

No treinamento do aprendizado não-supervisionado, o algoritmo recebe apenas as entradas sem saber quais serão as saídas, ou seja, não há rótulos nem referências para dizer ao algoritmo qual deve ser o resultado. O aprendizado é feito pela rede tentando encontrar similaridades nas estruturas dos dados de treino. É utilizado, por exemplo, em agregação de dados de redes sem fio (YOON; SHAHAB, 2007).

2.2 Aprendizado Profundo

Aprendizado profundo, ou *Deep Learning* (DL), é uma parte da área de *machine learning* na qual os dados passam por múltiplas transformações não-lineares para calcular um saída. O termo DL se refere à quantidade de passos que é necessária para se determinar uma saída.

Uma vantagem do DL sobre o ML clássico é conseguir extrair automaticamente *features* de alto nível de dados complexos, o que significa que processo de aprendizado

não precisa ser arquitetado previamente pelo desenvolvedor (LECUN; HINTON, 2015).

2.2.1 Redes Neurais

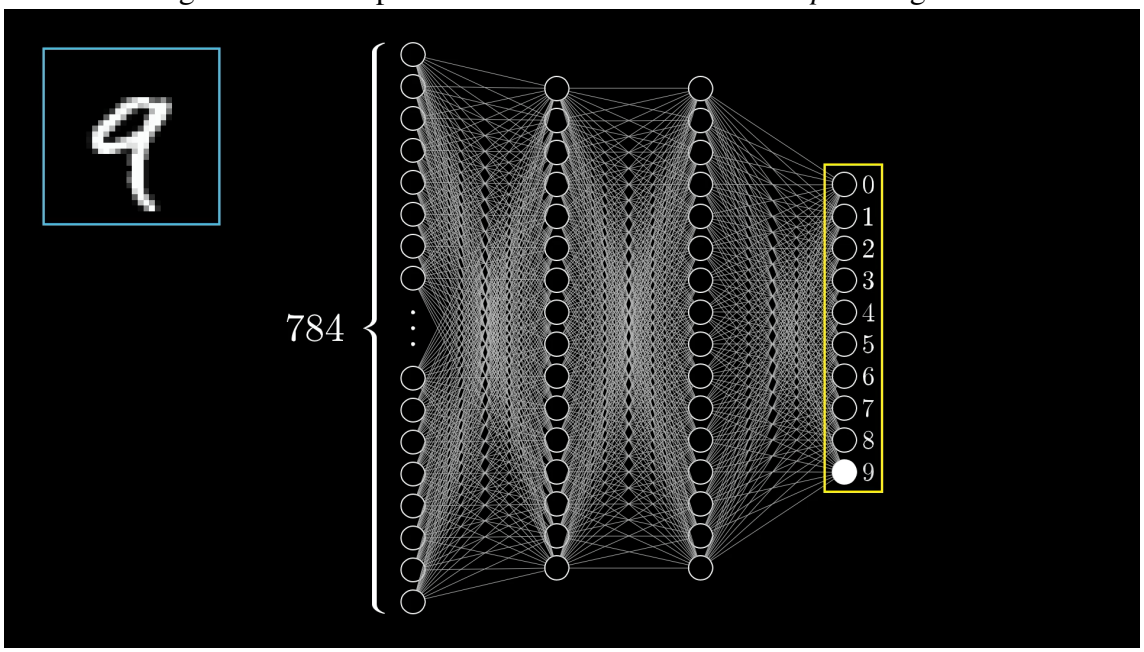
Tipicamente, DL faz uso de redes neurais (SCHMIDHUBER, 2015). Uma rede neural recebe esse nome por ser inspirada no funcionamento do cérebro humano. Os passos de determinação de um saída são dados por camadas de nodos chamados neurônios. Um neurônio recebe uma informação que é interpretada como um número. Um conjunto de neurônios vai possuir um conjunto de números, e a média desses números determina um valor que é considerado uma decisão. Cada neurônio de uma camada está ligado a todos os neurônios da camada seguinte, e essa decisão significa dar mais peso a um desses próximos neurônios. O processo se repete até a última camada, onde a última decisão é a saída. O objetivo da rede neural é, além de tomar uma decisão, verificar o quão errada ela está e executar ajustes para que a decisão correta/esperada seja tomada pela rede nas próximas análises.

Um exemplo pode ser visto na Figura 2.1, onde uma rede neural tenta determinar qual o dígito numérico está representado em uma imagem (*input*). Os dígitos ocupam um espaço de dimensões 28×28 pixels, cada um com um valor que vai de 0.0, preto, até 1.0, branco. O número da primeira camada de neurônios foi dado pelo total de pixels, ou seja, 784. Esses 784 pixels serão a camada de entrada, e os resultados, que são a decisão, serão dados pela camada de saída. Portanto, 10 neurônios, 1 para cada dígito possível. Vale ressaltar que não há uma regra fixa que determine a quantidade de neurônios nas camadas da rede que fornecerão o melhor resultado.

As camadas intermediárias servem para detectar partes de um dígito, um padrão específico, como um círculo, uma linha ou um quadrado, por exemplo. Uma Figura que possa fazer parte de um dígito. Cada neurônio fica responsável por detectar uma certa Figura, ou seja, um certo padrão de pixels que possam representar um número específico. Essas Figuras procuradas são as *features* de um dígito. Quando o atributo correspondente a esse neurônio for mais evidente para a camada em questão, maior será o valor que ele dará para o neurônio da camada seguinte que corresponde a um dígito que possui essa *feature*. Portanto, no caso, o número de neurônios intermediários depende do número de partes menores de um dígito diferentes que queremos analisar.

Cada neurônio de uma camada é ligado a neurônios da camada seguinte, possivelmente todos, pois um mesmo pixel de um padrão procurado pode também ser parte de

Figura 2.1: Exemplo de rede neural tendo como *output* o dígito 9.



Extraído de <https://www.3blue1brown.com/lessons/neural-networks>

um outro padrão. Então, por meio de pesos, o que está sendo analisado vai influenciar os próximos neurônios com intensidades diferentes. Esse processo se repete até uma saída ser determinado na última camada.

O treinamento de redes é normalmente parametrizado pelo número de "épocas", que determinam quantas iterações envolvendo os dados de entrada a rede fará ao analisar uma entrada. Ao final de cada iteração, os neurônios possuirão um valor que foi determinado pelos pesos dos neurônios das camadas anteriores. Porém, esses valores finais tipicamente possuem um erro que os distanciam do valor ideal, ou seja, do valor que mais fielmente representaria aquela entrada. Assim, a cada iteração, os pesos das camadas anteriores são reajustados para que o resultado final convirja para esse valor ideal. Esse ajuste é dado por um algoritmo, sendo um dos mais comuns o chamado algoritmo de *back-propagation* (WALDO, 2022). De forma resumida, a saída é comparada com o rótulo da entrada correspondente (caso supervisionado). A comparação ocorre através de uma função chamada função de custo, que calcula a diferença entre o resultado obtido e o esperado, e a convergência para o valor é baseada em algoritmos do vetor gradiente descendente, que serve para descobrir onde a função de custo cresce mais rapidamente (ANDRYCHOWICZ et al., 2016). Portanto, o negativo do vetor dará a direção onde o custo cai mais rapidamente. A idéia é dar um pequeno passo nessa direção e repetir o cálculo, considerando que podem haver muitas curvas e a curva mais próxima não necessariamente será a melhor opção. Um passo pequeno vai dar maior precisão na escolha,

porém fará com que a função demore mais para convergir para o resultado. Esse passo, na verdade, é uma proporção do vetor gradiente.

2.2.2 Transfer Learning

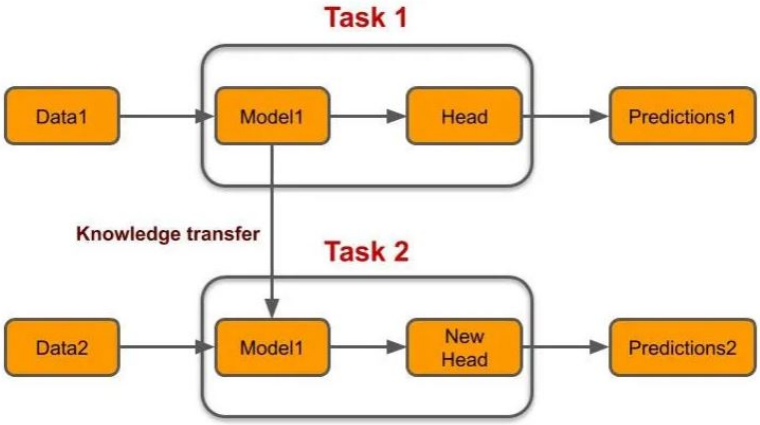
O termo *transfer learning* (TL), de forma objetiva, refere-se ao uso de *features* aprendidas para resolver um problema na resolução de um outro problema similar. (PAN; YANG, 2010) Em termos de algoritmo, uma rede que foi treinada anteriormente pode ser reaproveitada para determinar saídas similares aos que a rede foi treinada para avaliar originalmente. Isso pode ser feito se excluindo algumas camadas da rede original e as substituindo por novas camadas treinadas com os novos dados. O treinamento de uma rede fará com que as camadas iniciais, as mais complexas, retenham o conhecimento das *features* aprendidas que poderão ser reaproveitadas. Esse reaproveitamento também pode ser chamado de transferência de conhecimento.

O TL pode ser dividido em duas partes: pré-treinamento e o ajuste fino. O pré-treinamento se refere ao modelo a ser reaproveitado. A arquitetura deve ser definida, e pode ser treinada ou se pode carregar pesos com o resultado de um treinamento prévio. A etapa de ajuste fino se refere à substituição de camadas da rede.

Uma das grandes vantagens da técnica de TL é a economia de tempo e de recursos. Muitos modelos necessitam de grandes volumes de dados e podem demorar semanas para serem treinados, mesmo com a utilização de várias GPUs para acelerar o processamento. Por isso, geralmente, o modelo pré-treinado é um modelo que fez uso de grande volume de dados e seus pesos são carregados e reaproveitados para outra tarefa. Devido a esse volume de dados, o TL também ajuda a evitar o *overfitting*. Na Figura 2.2, pode ser visto um esquema simples de como o TL funciona.

No contexto deste trabalho, por exemplo, um modelo, mais especificamente, o *VGGFace* (PARKHI; VEDALDI; ZISSERMAN, 2015), uma arquitetura criada para reconhecimento facial, foi treinado com milhões de imagens de rostos, onde aprendeu *features* de rostos dos mais variados tipos com o objetivo. Porém, originalmente, este modelo só seria capaz de determinar saídas para os rostos das pessoas cujas imagens faziam parte dos dados de treinamento. Para evitar treinar a rede novamente e para reaproveitar as *features* aprendidas, uma pequena rede neural, isto é, com poucas camadas, é treinada apenas com os rostos das pessoas de interesse do estudo. As camadas de saída, as camadas mais de topo da rede original, são excluídas, e a nova rede substitui as saídas originais,

Figura 2.2: Esquema da técnica de *transfer learning*. Um modelo possui várias camadas, das quais algumas (*head*) são substituídas por novas camadas. As novas camadas possuem um treinamento diferente das camadas removidas para direcionar o resultado final para o novo objetivo da nova rede formada.



Extraído de <https://msalamiitd.medium.com/>

"forçando"a rede do *VGGFace* a escolher uma das pessoas da nova base de dados como resultado. Mais detalhes serão vistos na seção 3.3.

2.3 *Deblurring*

O objetivo geral de técnicas de *deblurring* é estimar uma imagem nítida a partir de sua versão borrada. De um modo geral, a formulação matemática é dada por

$$Y = K \otimes X + n, \tag{2.1}$$

onde Y é a imagem degradada (conhecida), K é o *kernel* (matriz) que descreve o borramento existente, X é a imagem nítida, $\otimes$ é uma operação de convolução, e n representa um ruído Gaussiano aditivo. Nesse caso, o objetivo é determinar a imagem X.

Existem duas grandes classes de modelos de *deblurring*: *blind* e *non-blind*. *Non-blind deblurring* significa resolver a Eq.(2.1) quando o *kernel* K é conhecido, enquanto que *blind deblurring* ataca o caso em que K é desconhecido (sendo tipicamente estimado no processo).

No mundo real, o borramento de uma imagem não será uniforme devido a fatores como movimento de objetos em uma trajetória curva e movimento da câmera que registra a imagem. Isso torna a Eq.(2.1) imprópria para caracterizar o borramento local. No caso de rostos, para este trabalho, assume-se que a pessoa normalmente estará virada de frente

para a câmera, de modo que mudanças na trajetória dentro do tempo de exposição da câmera serão casos incomuns para que um borramento não-uniforme ocorra. Isso torna plausível representar o borramento dos rostos pela eq.(2.1).

Como o foco do trabalho está em reconhecimento facial sem um conhecimento prévio do movimento do rosto, os testes de modelos com *blind deblurring* terão prioridade, ou seja, ficará a cargo da rede neural estimar o *kernel*. Métodos *blind* que não estimam o *kernel*, chamados de métodos fim-a-fim, como (KUPYN VOLODYMYR BUDZAN; MISHKIN; MATAS, 2018), ignoram informações de movimento que podem comprometer a qualidade do resultado.

Métodos baseados em *deep learning* têm se mostrado mais eficaz do que métodos mais antigos (WANG; XU; ZHENG, 2023), chamados de métodos baseados em modelos, sendo a abordagem mais utilizada atualmente. Portanto, os testes e estudos se basearão nessa mesma abordagem. Algumas técnicas serão revisadas a seguir.

2.3.1 *Deblurring* baseado em Imagens (componente espacial)

Métodos baseados em *deep learning*, têm obtido resultados promissores. Uma classe de métodos se preocupa em estimar o *kernel* de borramento, como (BAHAT; EFRAT; IRANI, 2017), (CHAKRABARTI, 2016), (SUIN; PUROHIT; RAJAGOPALAN, 2020) e (SUN; XU; PONCE, 2015), enquanto a outra tenta estimar a imagem tratada sem estimar o *kernel* antes, como (KUPYN VOLODYMYR BUDZAN; MISHKIN; MATAS, 2018), (CHO SEO-WON JI; JUNG; KO., 2021) e (LIANG JIEZHANG CAO; GOOL; TIMOFTE, 2021). Ambas as classes de técnicas possuem seus desafios: estimar o *kernel* de borramento não-uniforme não é trivial, enquanto que tentar tratar a imagem sem o *kernel* ignora um *prior* de movimento, afetando a qualidade do *deblurring*.

No caso de um borramento real, em geral, não haverá *ground truth*, além provavelmente não ser uniforme, o que torna estimar esse tipo de borramento uma tarefa complexa. Por exemplo, o modelo *Self-supervised Non-uniform Kernel Estimation with Flow-based Motion Prior for Blind Image Deblurring* (UFPDeblur), proposto por (FANG et al., 2023), simula o borramento não-uniforme através de normalização dos *kernels* de movimento no espaço latente de uma maneira auto-supervisionada, sendo o espaço estimado com a introdução de incerteza para melhorar a eficiência da estimação. O *kernel* de borramento complexo é representado em uma distribuição Gaussiana simples através de uma técnica chamada de normalização do fluxo, primeiramente proposta por (DINH;

KRUEGER; BENGIO, 2014).

A tentativa de se resolver um borramento real por *blind deblurring* se aproxima do problema de reconhecimento facial, uma vez que um rosto registrado em uma imagem virtualmente nunca possuirá uma imagem nítida que possa servir como *ground truth*. Porém, para que os resultados sejam melhores, é importante que o método tenha um foco em rosto, ou seja, que utilize *features* de rosto como *prior* para evitar perda de alguns detalhes mais finos, como contorno da boca e separação entre dentes, por exemplo. Para fins de teste e comparação, o UFPDeblur foi um dos modelos escolhidos para realizar os experimentos.

2.3.2 *Deblurring* Facial

Deblurring facial é um problema de domínio específico. Embora métodos genéricos de *deblurring* também podem ser utilizados, faces possuem alguns componentes fixos que podem ser usados como *prior* para melhor performance, como a pele, cabelo, olhos, nariz e boca. Todos esses são elementos comuns às faces, mas cada rosto possui pequenas diferenças, como formas e textura. Esses elementos são cruciais para caracterizar uma face específica, de modo que, para melhor identificação, o foco em faces se torna uma necessidade.

Entre os vários tipos de métodos que utilizam informação como *prior*, alguns se baseiam em uma face como referência, como os propostos por (HACOHEN; E; D., 2013) e (PAN; Z; Z, 2014). Outros utilizam faces 3D, como (REN J. YANG; CAO; TONG, 2019). Entretanto, os que apresentaram melhores resultados foram os baseados em segmentação semântica, como (SHEN W.-S. LAI; KAUTZ; YANG, 2018) e (YASARLA; PERAZZI; PATEL, 2020), atingindo o estado-da-arte na época de suas publicações. Porém, estimar esse mapa de segmentação não é trivial, e pode resultar em artefatos se feito incorretamente, principalmente em componentes pequenas do rosto, como olhos e lábios. Esse problema foi notável nas imagens tratadas por Shen W.-S. Lai, Kautz and Yang (2018). Yasarla, Perazzi and Patel (2020) conseguiram reduzir o impacto de estimações incorretas aplicando um *score* de confiança para ajudar a determinar o impacto do mapa de segmentação no *deblurring* da imagem, mas o processo ainda era subótimo por não utilizar *priors* de rosto.

Para resolver esse problema, Lee, Jung and Heo (2020) propuseram uma abordagem baseada em redes adversariais generativas (MSPL-GAN). O método aproveita infor-

mações semânticas do *prior* de um rosto sem fazer segmentação para prevenir um mapa de segmentação falho. Além disso, o método progressivamente restaura o rosto em quatro passos, inspirado por técnicas de aprendizado progressivo (KARRAS; LAINE; LEHTINEN, 2017; KARNEWAR; WANG, 2020). O MSPL-GAN possui múltiplas sub-redes, e cada sub-rede é treinada para focar em restaurar uma componente do rosto. Esse método simples permite o *deblurring* de rosto mais eficientemente. Para gerar imagens de rosto mais fotorrealistas, o método propõe um discriminador multi-semântico, desenvolvido para cuidar de todos os *outputs* do gerador usando uma única rede classificadora. A estrutura geral do modelo pode ser vista na Figura 2.3.

Figura 2.3: *Overview* do *framework* proposto por (Lee; Jung; Heo, 2020) , compreendendo um gerador e um discriminador. O gerador reconstrói a imagem em quatro passos, e o discriminador observa as imagens geradas e tenta determinar se são reais ou falsas.

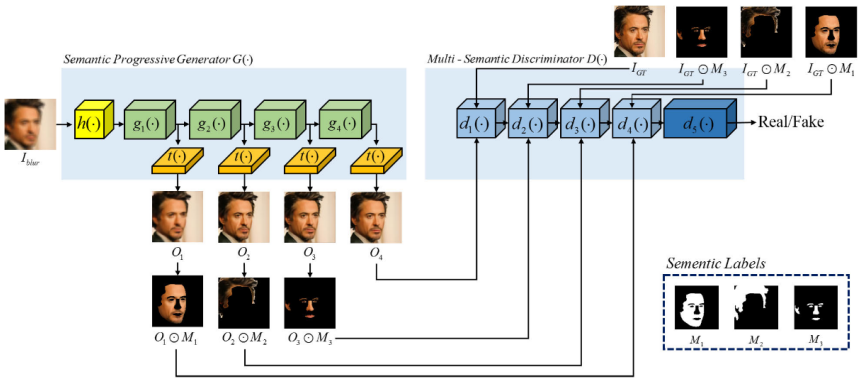


FIGURE 2. Overview of the proposed face deblurring framework comprising of a generator and a discriminator. The generator reconstructs the image in four steps, and the discriminator observes all the output images from the generator.

Extraído de (Lee; Jung; Heo, 2020)

O MSPL-GAN também foi escolhido como um dos modelos para realizar os experimentos de *deblurring* neste trabalho.

2.4 Detecção e Reconhecimento Facial

A detecção da pessoa se refere à detecção de um rosto. O objetivo da detecção facial é encontrar *features* que representem um rosto humano sem se importar em determinar exatamente quem é aquela pessoa (MINAEE et al., 2021). Ou seja, queremos reconhecer um padrão comum a todas pessoas mas sem analisar precisamente a diferença que essas *features* possam apresentar de uma pessoa para outra, como por exemplo, pequenas diferenças de tamanho, cor ou marcas que possam existir na pele.

O reconhecimento facial se refere à confirmação da identidade de uma pessoa pelo seu rosto. Como mencionado anteriormente, os elementos do rosto (boca, nariz, olhos,

etc.) são cruciais para diferenciar uma pessoa de outra. Antes que essa confirmação possa ser executada, deve-se primeiramente detectar um rosto de uma pessoa em uma imagem. Se o rosto puder ser extraído do resto da imagem, estaremos isolando as *features* desse rosto, que naturalmente possuirá menos elementos do que uma imagem inteira com um fundo presente. Isso poderá garantir maior precisão na hora da identificação da pessoa por parte da rede neural.

Para que uma rede possa identificar a pessoa, é necessário que ela seja treinada com imagens desta pessoa a ser identificada, e que haja uma associação entre as imagens e o nome da pessoa, por exemplo. Em técnicas de reconhecimento facial, é comum que imagem seja convertida para uma cor monocromática (*grayscale*) para reduzir a complexidade da imagem e também acelerar o processamento (BUI et al., 2016).

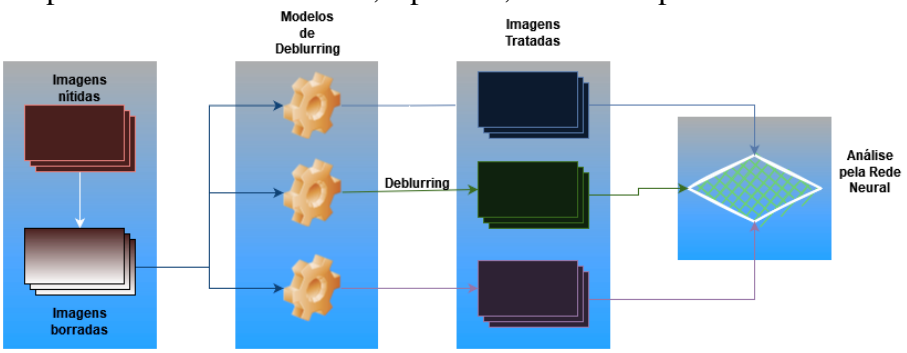
Já a detecção pode ser feita por, por exemplo, cascatas de Haar (BRADSKI, 2000) ou por detecção de pontos comuns a faces (Dlib, (KING, 2009)). Testes realizados com a biblioteca Dlib se mostraram eficazes na detecção de faces. A parte do reconhecimento propriamente dito pode ser realizado usando várias técnicas (WANG et al., 2022). Uma técnica popular é o modelo de reconhecimento facial *VGGFace* (PARKHI; VEDALDI; ZISSERMAN, 2015). O *VGGFace* propriamente dito é uma rede convolucional treinada com um *dataset* de rostos. Os pesos desse modelo estão disponíveis e podem ser utilizados junto com um classificador, outra rede, para conseguir determinar a identidade de uma pessoa sem precisar treinar a rede novamente com o uso de TL.

3 METODOLOGIA

Este trabalho propõe um estudo relacionado ao tratamento de imagens corrompidas por borramento causado por movimento no contexto de reconhecimento facial. Mais precisamente, foi avaliado o impacto conjunto da detecção de faces e o posterior reconhecimento antes e após o uso de técnicas de correção de borramento. O trabalho pode ser dividido em três etapas principais: geração de dados para experimentação, tratamento das imagens e análise das imagens pela rede.

A geração de dados consiste em aplicar *motion blur* a imagens nítidas, de forma artificial. Através de um algoritmo simples, operações de convolução são realizadas com tamanhos e ângulos de kernel diferentes para gerar várias versões borradas da mesma imagem nítida com diferentes magnitudes e orientações de borramento. Para o tratamento das imagens borradas, alguns modelos desenvolvidos para executar deblurring de motion blur foram escolhidos. Todos os modelos foram treinados pelos seus respectivOs autores, e os testes realizados neste trabalho utilizaram os pesos pré-treinados. Cópias de cada imagem borrada foram submetidas a cada modelo, ou seja, uma imagem já tratada por um desses modelo não foi submetida a um novo tratamento por outro. Por fim, todas as imagens são submetidas a uma rede neural para análise e classificação. As imagens borradas não tratadas também foram analisadas para servirem de referência à analise das imagens tratadas, sendo assim possível comparar os resultados. A Figura 3.1 ilustra o processo adotado neste trabalho, e cada etapa é detalhada a seguir.

Figura 3.1: Esquema da metodologia do trabalho. Imagens nítidas são borradas, depois são tratadas por cada um dos modelos, e por fim, analisadas pela rede neural.



Fonte: Os autores

Figura 3.2: Um kernel de *motion blur* de tamanho 5×5 que representa um vetor bidimensional de ângulo zero.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fonte: Os autores

3.1 Geração da Base de Dados

Foram escolhidas três pessoas diferentes, e três imagens diferentes de cada, sendo um total de nove imagens nítidas a partir das quais foram criadas diversas versões borradas. As imagens nítidas possuem resoluções variadas entre si, e foram coletadas manualmente através de pesquisa no Google. Cada imagem possui apenas um único rosto, e o borramento gerado foi feito de forma artificial através de um *script* escrito em linguagem Python, utilizando-se funções da biblioteca OpenCV para aplicação de operações de convolução. Para gerar um kernel de *motion blur* "canônico" na direção horizontal, é crida uma matriz nula (com dimensões especificadas) e a linha central é preenchida com valores um. Kernels em direções arbirtárias podem ser gerados rotacionando o kernel canônico, e neste trabalho foram usados os métodos *cv2.getRotationMatrix2D()* e *cv2.warpAffine*. Um exemplo de kernel canônico horizontal é ilustrado na Figura 3.2, e uma versão rotcionada na Figura 3.3. O kernel de borramento se aproxima do conceito de *Point Spread Function* (CHAOYANG; GUANGJUN; QIFENG, 2008) (PSF), que descreve como os pontos de uma imagem são borrados, mas este também podendo incluir outros tipos de borramento.

Figura 3.3: Um kernel de *motion blur* de tamanho 5×5 que representa um vetor bidimensional de ângulo 25. Notar que mais linhas são preenchidas para tentar compensar a inexatidão no ângulo devido ao tamanho do kernel.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0.010 \\ 0 & 0 & 0.093 & 0.531 & 0.703 \\ 0.156 & 0.562 & 1 & 0.562 & 0.156 \\ 0.703 & 0.5 & 0.093 & 0 & 0 \\ 0.010 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fonte: Os autores

O tamanho do kernel está relacionado com a magnitude do borramento. Assim, espera-se que os resultados de reconhecimento tendam a piorar conforme o aumento das dimensões do kernel.

Cada imagem nítida foi borrada com kernels de tamanhos 10×10, 20×20, 30×30 e 40×40. Cada um desses kernels foi aplicado em 14 direções diferentes angularmente equidistantes com espaçamento de $25°$. Mais precisamente, foram usados os ângulos no conjunto $\{0°, 25°, 50°, ..., 325°\}$. No total, temos 504 imagens borradas para os experimentos.

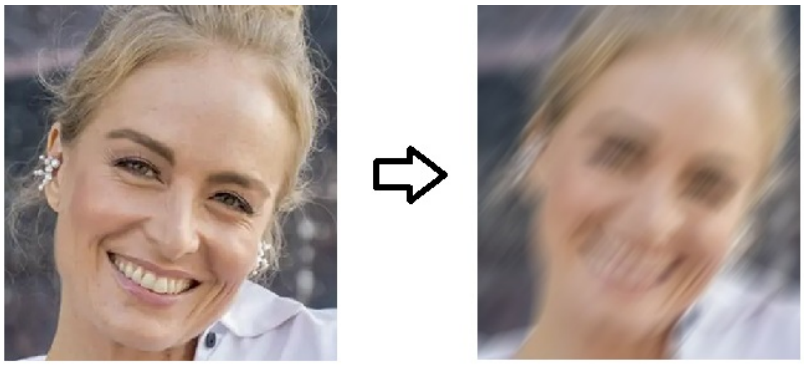
Além das imagens das 3 pessoas utilizadas para os experimentos,também foram coletadas imagens de outras 7 pessoas, estas mantidas nítidas para o treinamento do classificador. O classificador, portanto, foi treinado com imagens de um total de 10 pessoas diferentes. As imagens que foram borradas para realização dos experimentos são diferentes das imagens utilizadas para o treino e diferentes das imagens utilizadas para validação do classificador. Mais detalhes na subseção 3.3.2.

3.1.1 Preparação dos Dados e Detecção de Rosto

Para o *deblurring*, foram utilizados três modelos diferentes: UFPDeblur (FANG et al., 2023), MSPL-Gan (Lee; Jung; Heo, 2020) e Unsup-IMG(XIA; CHAKRABARTI, 2019). Pode ser visto um exemplo de imagem borrada na Figura 3.4 antes do tratamento por um dos modelos. Porém, destes, apenas o UFPDeblur foi treinado para receber imagens de qualquer dimensão, podemos assim receber as imagens para tratamente diretamente após a aplicação do blur. Os outros foram treinados com imagens de dimensões 128x128, obrigando que uma versão redimensionada do conjunto de imagens seja confeccionada. Um exemplo pode ser visto na Figura 3.5. A imagem original é borrada, que é submetida à análise pela biblioteca Dlib. O rosto detectado é recortado e redimensionado antes de ser salvo em um novo arquivo.

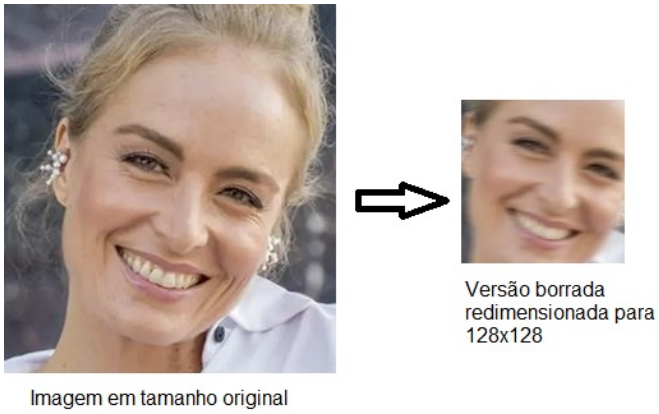
Para gerar as imagens borradas redimensionadas, existe o desafio da detecção. Se uma imagem for redimensionada diretamente, pode haver grandes perdas de informação útil para a rede, pois o rosto quase sempre ocupará apenas uma porção da imagem. Então é conveniente detectar a área do rosto na imagem, recortar esta área e redimensionar apenas ela. A detecção foi feita com o uso da biblioteca Dlib, que usa uma rede neural para procurar 68 pontos característicos em um rosto humano. Utilizando um classificador de cascatas de Haar (SWE, 2020), a biblioteca retorna um *array numpy* multidimensional,

Figura 3.4: Exemplo de uma imagem nítida e o resultado após o borramento artificial aplicado.



Fonte: Os autores

Figura 3.5: Exemplo do resultado final do borramento de uma imagem redimensionada para 128×128. Notar que as imagens nítidas não precisaram ser redimensionadas.

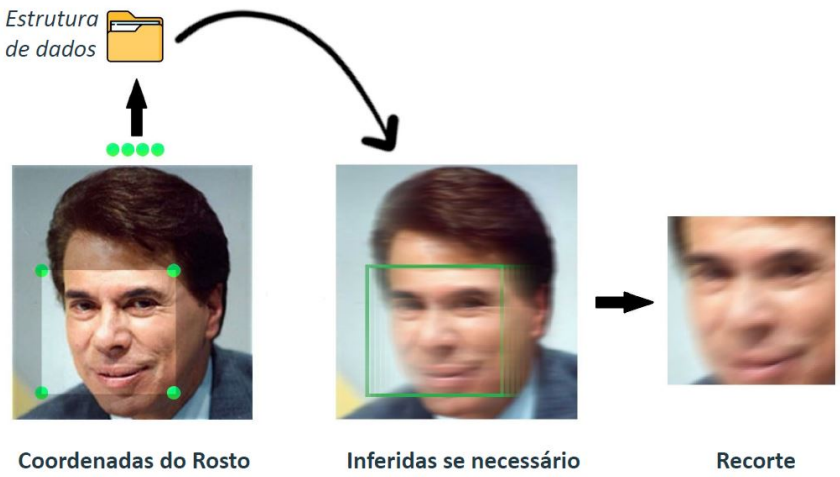


Fonte: Os autores

onde cada elemento é um rosto detectado na imagem. Também estão disponíveis funções que retornam as coordenadas dos rostos. A partir dessas coordenadas, é possível recortar o rosto e redimensioná-lo em uma nova imagem, minimizando a perda de informação.

Nos testes realizados, a detecção de rosto foi bem sucedida com todas as nove imagens nítidas. Ao se aplicar o *motion blur*, nem todas as imagens passaram a ter o rosto detectado. As imagens borradas que tiveram rosto detectado foram redimensionadas a partir das coordenadas dadas pela Dlib. Para os casos em que o rosto não foi encontrado, por ser de interesse do estudo também avaliar a melhora na detecção, foi feita uma aproximação da área que seria detectada pela Dlib. As coordenadas dos rostos das imagens nítidas foram guardadas em uma pequena estrutura de dados e estas foram aplicadas às imagens borradas dadas como sem rosto. Dessa forma, é garantido que todo o conjunto de imagens será redimensionado independentemente da intensidade ou da direção do borramento. O processo pode ser visto na Figura 3.6.

Figura 3.6: Ilustração do processo de recorte de rosto quando a imagem está borrada o suficiente para a detecção facial falhar. As coordenadas da imagem limpa são guardadas e inferidas na imagem borrada quando o rosto não é detectado.



Fonte: Os autores

3.2 Deblurring das Imagens

Como dito, uma cópia do *batch* de imagens é utilizado como *input* de cada modelo de *deblurring*, sendo escolhido o *dataset* redimensionado quando necessário. O objetivo é analisar as imagens tratadas e avaliar o reconhecimento de indíviduos em cada uma.

3.2.1 Modelos de *Deblurring* Utilizados

Como explicado na seção 2.3, um dos modelos utilizados foi o MSPL-GAN, um modelo baseado em redes adversariais generativas e treinado com imagens de rosto. Este modelo, embora aceite imagens de qualquer tamanho, foi treinado com imagens de tamanho 128x128, e portanto, foi utilizado o *dataset* redimensionado.

O segundo modelo utilizado é o UFPDeblur, um modelo desenvolvido para tratamento de borramento real não uniforme, não sendo treinado especificamente para aplicar *deblurring* de rostos. As imagens tratadas foram as de tamanho original, sendo estes tamanhos variados entre si.

O terceiro modelo escolhido é UnsupIMG (XIA; CHAKRABARTI, 2019), que introduz um *framework* não-supervisionado para treinar redes de estimação de imagens, e aprender a explorar características de domínio especifico, no caso, rostos. Não é necessário *ground truth* durante o treinamento, embora o modelo também esteja preparado

para treinamento supervisionado. Para o *blind deblurring* não-supervisionado, o modelo assume que o kernel de borramento é desconhecido. Para o *non-blind deblurring*, as imagens de *ground truth* não são usadas, mas o modelo assume que os kernels são conhecidos. Os tamanhos kernels de borramento do treinamento variam de 13×13 a 27×27, aumentando de 2 em 2. No caso do UnsupIMG, o *dataset* redimensionado foi utilizado e processado das duas formas; os pesos do modelo treinado para ambos os modos foram disponibilizados. É importante notar que os tamanhos de kernel 10×10, 30×30 e 40×40 estão além dos tamanhos de kernel utilizados para o treinamento supervisionado do modelo.

De forma objetiva, a execução do *deblurring* por parte de cada modelo se dá por uma linha de comando única. As imagens são postas em um diretório específico e o outro diretório serve para guardar o resultado, e cada imagem foi tratada por cada uma das técnicas *deblurring*. Devido à falta de uma GPU compatível com os códigos originais dos modelos de *deblurring*, todos os modelos foram executados apenas com uso de CPU. Foram necessárias alterar linhas de vários *scripts* para garantir que apenas a CPU processasse as imagens.

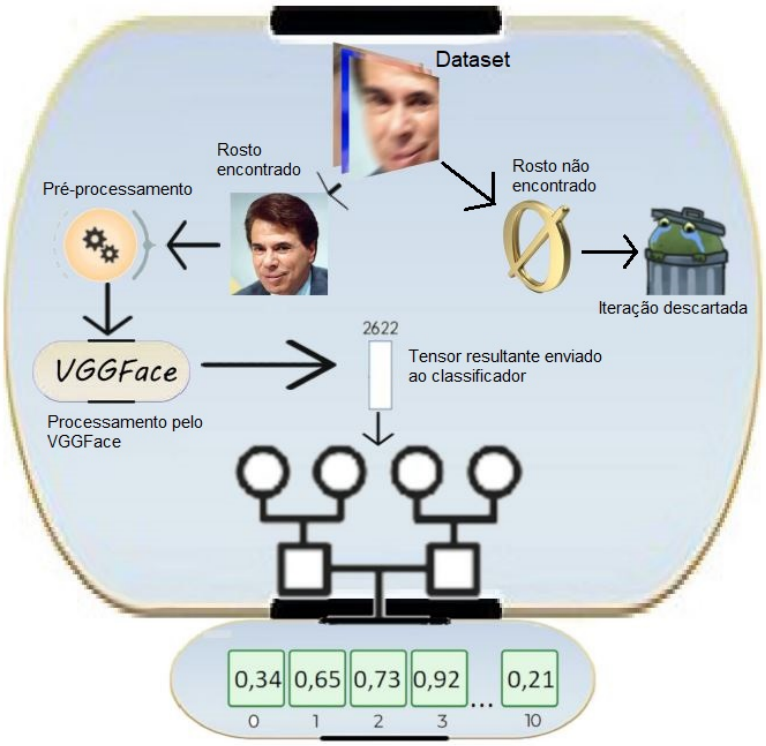
3.3 Análise e Reconhecimento de Indivíduos

A análise das imagens é composta de 3 módulos principais. O primeiro é a rede *VGGFace* (PARKHI; VEDALDI; ZISSERMAN, 2015), modelo pré-treinado que possui conhecimento de *features* de rostos. O segundo é o classificador, pequena rede treinada apenas com os rostos das pessoas que se deseja analisar e que serve de saída para a rede do *VGGFace*. O terceiro é o algoritmo que envia as imagens para a rede, analisa e guarda os resultados. Um resumo da análise pode ser visto na Figura 3.7.

3.3.1 Recursos Utilizados

Com exceção dos modelos de *deblurring*, todos os testes foram realizados no ambiente do Google Colab, que faz uso de uma GPU *NVIDIA* Tesla T4. As bibliotecas principais utilizadas foram o TensorFlow (ABADI et al., 2016) (TF) e o Keras (CHOLLET, 2015). TensorFlow é uma plataforma de código aberto que oferece inúmeras funções para desenvolvimento de aplicações que utilizam inteligência artificial. O Keras é uma

Figura 3.7: Resumo da análise de um *dataset*. Se não for encontrado rosto, a iteração é descartada e a próxima é iniciada. Se for encontrado o rosto, a imagem será pré-processada e enviada à rede do *VGGFace*, que devolverá um tensor de 2622 dimensões. O tensor será enviado a um classificador que irá determinar um vetor de probabilidades, cujo elemento de maior valor representará a pessoa identificada pela rede como a mais provável na imagem.



Fonte: Os autores

API de alto nível que serve como uma interface entre o usuário e a biblioteca que serve de backend para o processamento. No caso deste trabalho, o TF é o backend, mas o Keras também pode ser usado em conjunto com Pytorch (PASZKE et al., 2019). Dentre os recursos oferecidos pelo TF, o Keras foca na instanciação de modelos e de camadas de redes neurais.

3.3.1.1 VGGFace

O *VGGFace* é uma rede neural treinada com 2622 pessoas diferentes e um total de 2.6 milhões de imagens. Seu treinamento foi feito com dois *datasets*, o *Labeled Faces in the Wild* (LFW) (HUANG et al., 2008) e o *YouTube Faces* (YTF) (WOLF; MAOZ, 2011), A rede e a arquitetura são descritos em (PARKHI; VEDALDI; ZISSERMAN, 2015). A rede aceita imagens exclusivamente de dimensões 224×224, significando que todas as imagens precisam ser redimensionadas antes de serem analisadas. Em maior parte, as imagens sofrem redução de tamanho, porém, não é o caso quando estas são as imagens

do *dataset* de dimensões 128×128, causando uma inevitável perda de informação do ponto de vista do *VGGFace*.

Após a instanciação de sua arquitetura, os pesos do modelo treinado do *VGGFace* são carregados e as duas últimas camadas (*flattening* e ativação) da rede são excluídas. A exclusão das camadas deve ser feita após o carregamento dos pesos para não haver incompatibilidade com a arquitetura. Não existe uma regra que determine a escolha que leve ao melhor resultado, mas foram excluídas as últimas duas pois são as camadas que vem após a última camada de convolução, uma camada que gera um tensor de 2622 dimensões, ou seja, o mesmo número de identidades usadas no treinamento. As camadas excluídas são substituídas pelas camadas do classificador, como dita a técnica de TL.

3.3.2 Classificador

O classificador é uma pequena rede que recebe apenas as imagens das pessoas que se quer classificar, sendo esta uma rede individual e independente do *VGGFace*. Como mostra a figura 3.8, ele é usado como saída da rede principal, aproveitando-se das características aprendidas pelo *VGGFace* para atingir uma maior certeza do resultado com menos dados, ou seja, o conhecimento aprendido pelo *VGGFace* faz com que se precise de menos dados de treinamento para se alcançar um mesmo valor de certeza. A arquitetura do classificador segue o modelo proposto por (ASTAWA et al., 2021), que utiliza três camadas densas, cada uma seguida de uma camada de ativação.

A última camada foi alterada para representar o mesmo número de pessoas diferentes que o classificador pode reconhecer (dez).

As imagens que serão usadas pelo classificador são recortes dos rostos de cada imagem original. Esses recortes são redimensionados para 224x224 antes de se fazer um pré-processamento de cada um.

3.3.2.1 Pré-processamento e Treinamento

A imagem é transformada em um *array* unidimensional. Depois, esse array é passado ao método *preprocess_input()*, que serve para adequar a escala de valores dos pixels imagem ao modelo em questão. Por exemplo, é normal imagens apresentarem um canal de cor de pixel com um valor entre [0-255], mas o método *preprocess_input()* pode normalizar este valor para um outro valor entre [0-1], ou [-1,1], a fim de acelerar o

Figura 3.8: Ilustração do objetivo do classificador. As características aprendidas pelo *VGGFace* são aproveitadas para se determinar *features* de rostos com maior precisão. O tensor recebido do *VGGFace* será especializado em um vetor de probabilidades que representa as pessoas com o qual o classificador foi treinado.

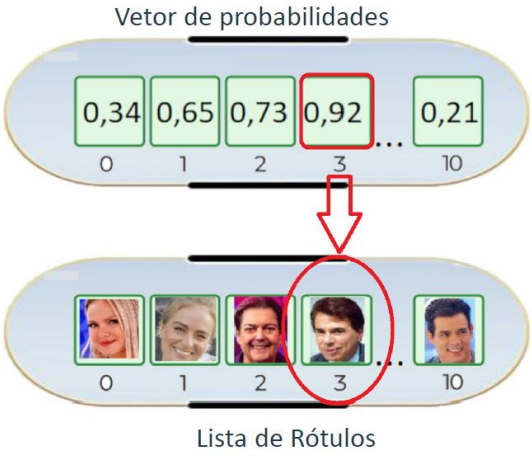
Fonte: Os autores

treinamento da rede mais tarde. No caso do *VGGFace*, o valor é normalizado entre [-1,1]. A seguir, o resultado desse pré-processamento é enviado à rede do *VGGFace* (já sem as camadas finais), e o tensor de 2262 dimensões resultante é guardado em uma lista.

Cada pessoa possui uma pasta onde suas imagens estão localizadas. O pré-processamento descrito ocorre por pasta, tornando simples criar rótulos para cada tensor. Os rótulos vão sendo atribuídos ao mesmo tempo que as imagens são pré-processadas e são guardados em uma outra lista, de modo que cada tensor ocupará uma posição p em uma lista X, e o seu respectivo rótulo ocupará a mesma posição p em uma lista Y. Os rótulos são números, mas facilmente identificáveis se contando o número de pastas que representam cada pessoa, como visto na Figura 3.9. A lista X é usada como parâmetro de treinamento da primeira camada do classificador.

Esse mesmo pré-processamento e atribuição de rótulos é repetido para as imagens de teste, que são as imagens que serão fornecidas ao classificador como dados de validação, ou seja, as saídas desejados. A rede é treinada com 100 épocas. Seguindo o estudo realizado por (ASTAWA et al., 2021), que utilizou 22% de seus dados disponíveis como imagens de teste e o restante como imagens de treinamento, foram escolhidas 5 imagens de teste para cada pessoa. No total, o classificador foi treinado para reconhecer 10 pessoas diferentes, cada uma possuindo 20 imagens de treinamento e mais 5 de validação, sendo um total de 250 imagens utilizadas. As imagens utilizadas no classificador não tem

Figura 3.9: Ilustração de como uma probabilidade está relacionada como uma identidade. O ordenamento das pessoas é determinado pela ordem em que suas respectivas pastas de dados são acessadas durante uma busca por diretórios com dados.



Fonte: Os autores

relação com as imagens utilizadas nos experimentos de *deblurring*.

Com o classificador instanciado, as duas listas são convertidas para *numpy arrays* antes de serem passadas ao classificador como *dataset* de treino.

3.3.3 Análise

3.3.3.1 Organização dos Dados de Teste

Todas as imagens que serão submetidas à análise estarão em um único diretório, sem separação em pastas por pessoa. É analisado um grupo de 504 imagens por vez, sempre referente a um único modelo de *deblurring* (o que tratou esse grupo) ou a imagens borradas ainda não tratadas. As imagens são nomeadas conforme o seguinte padrão: <*nome*>_0n_kA_B**,** onde <*nome*> é o nome da pessoa representada na imagem, n é um número que representa a imagem nítida que originou a imagem borrada em questão, A representa as dimensões do *kernel* utilizado para borrar a imagem (sempre quadrado, $A \times A$) e B representa o ângulo do *kernel* de borramento. Portanto, a rede já recebe todas essas informações juntamente à imagem; nenhuma delas precisa ser inferida, deduzida ou estimada.

A análise é composta em parte de passos já vistos nas sessões anteriores. Cada imagem de teste é carregada, uma por vez. A Dlib é utilizada para detecção de rosto. Se não houver ou não for encontrado um rosto na imagem, a iteração da imagem atual é descartada e a próxima imagem carregada.

A imagem é transformada em um *numpy array* unidimensional. Esse *array* é usado pelo método *predict()* para gerar um vetor de probabilidades, que é então utilizado pelo método *np.argmax()* para determinar o item mais provável, ou seja, a pessoa que mais provavelmente está representada na imagem. O retorno é um número que definirá o rótulo correspondente. Caso o rótulo seja diferente do <*nome*> no arquivo na imagem, significa que a predição está incorreta.

Em resumo, durante a análise, cada imagem recebe o mesmo tipo de pré-processamento que as imagens de treino e de validação. O tensor resultante do envio da imagem a ser analisada pela rede do *VGGFace* é guardado em uma variável e enviado ao classificador através do método *predict()*. A conexão entre as duas redes ocorre de forma virtual; as camadas do classificador não são ligadas diretamente à rede do *VGGFace* para formar uma única rede.

3.3.3.2 Dados Gerados pela Análise

Independentemente da predição estar correta ou não, dados são coletados e guardados separadamente em cada caso. Há uma contagem de quantas imagens tiveram o reconhecimento correto e quantas tiveram resultado incorreto. As probabilidades retornadas por cada predição também são guardadas, sendo possível calcular a média geral de ambos os casos.

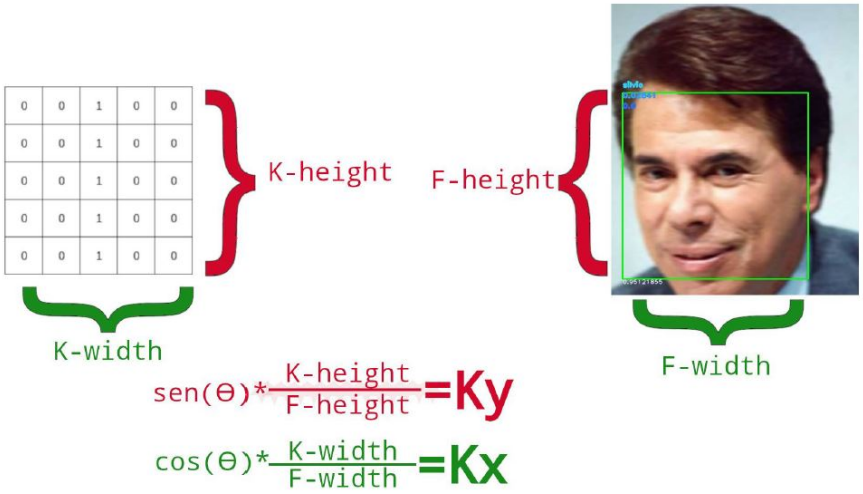
Utilizando o nome do arquivo, sabe-se qual a direção do *kernel* de borramento e seu tamanho. Referente a cada tamanho de *kerne*l e a cada ângulo, são guardas as médias totais e quantidade das imagens, assim como número de imagens sem rosto. Como dito, cada grupo de 504 imagens é referente a um modelo de *deblurring* em específico ou às imagens borradas antes do tratamento, ficando então, a cargo do usuário saber a qual modelo pertence essas imagens.

Por fim, é também gerado um número que relaciona o tamanho da área do rosto detectado pela Dlib, o tamanho e o ângulo do *kernel* de borramento. Quando uma imagem sofre borramento, um rosto que ocupa uma área pequena na imagem (em *pixels*) sofrerá uma deformação proporcionalmente maior do que um rosto que ocupe uma área grande. Este valor ajuda a determinar qual a intensidade máxima que um kernel pode ter antes que a quantidade de identificações erradas comece a se sobressair, e resume a média de sucesso ângulos mais horizontais (próximos de 0º ou 180º) e de ângulos verticais (próximos de 90º ou 270º).

O cálculo desse número é feito pela decomposição das componentes do vetor que

gera o borramento e pela divisão dessas componentes pelas dimensões do rosto encontrado na imagem. Os resultados da componente X de todas as imagens são guardados em uma variável e os da componente Y em outra. As variáveis são distintas entre os casos incorretos e os corretos. É guardado apenas a média geral de cada componente todas as imagens, não sendo separados por ângulos nem por tamanho de *kernel*. Um resumo da operação pode ser visto na Figura 3.10.

Figura 3.10: Cálculo da intensidade do kernel por área de rosto. A altura do kernel é dividida pela altura do rosto e multiplicada pelo seno do ângulo do *kernel* de borramento, obtendo-se a componente Y desse vetor. É calculada a média de todas as componentes Y de todas as imagens. O cálculo é análogo para a componente X, utilizando as larguras e o cosseno do mesmo ângulo.



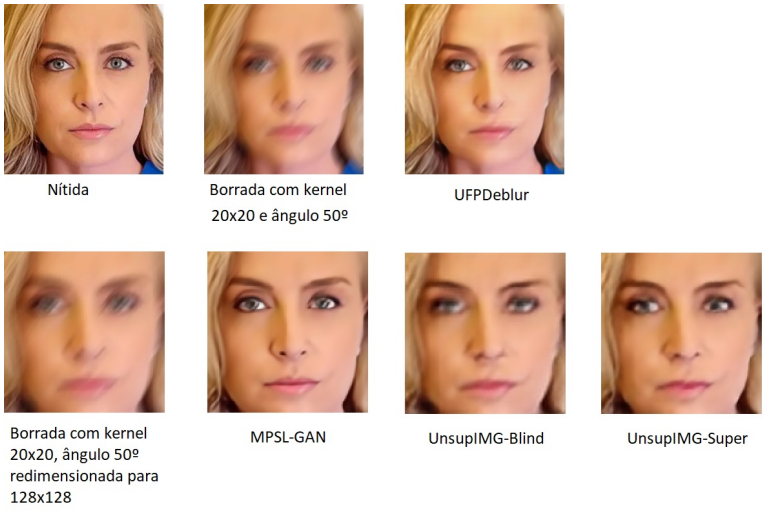
Fonte: Os autores

Para os casos das imagens redimensionadas (128×128), entretanto, as dimensões originais do rosto são perdidas, e o tamanho do *kernel* descrito no nome do arquivo está agora deformado na imagem, e não pode ser calculado pela rede de análise. Assim, o mesmo processo utilizado para realizar o recorte das imagens borradas é realizado. Todas as imagens borradas (sem tratamento) tiveram as dimensões do rosto e tamanho de *kernel* registrados em uma estrutura de dados previamente, e imagens cujos rostos não foram detectados tiveram o tamanho do rosto aproximado pela inferência das coordenadas das imagens nítidas. Para imagens redimensionadas, essas coordenadas são usadas para calcular as componentes do vetor original. No caso das imagens não redimensionadas, as componentes são determinadas em tempo real durante a análise.

4 EXPERIMENTOS E RESULTADOS

A partir da primeira análise da base da dados gerada, os resultados coletados foram analisados manualmente. E a partir dessa análise manual, alguns testes adicionais foram aplicados na tentativa de melhorar o resultado e testar algumas hipóteses que podem ter levado a esses resultados. Lembrando que o resultado da análise da rede é um vetor de probabilidades. Na Figura 4.1, é apresentado um exemplo que ilustra as diferentes transformações que ocorrem em uma imagem com rosto durante os testes.

Figura 4.1: Exemplo das imagens envolvidas para o tratamento de uma imagem borrada. A imagem nítida é a que origina todas as outras. Ela terá duas versões borradas, uma de mesmas dimensões e outra redimensionada para 128×128. O borramento é aplicado antes do redimensionamento. As imagens com os nomes dos modelos de *deblurring* abaixo são os resultados dos tratamentos pelos mesmos.



Fonte: Os autores

4.1 Análise das Imagens Tratadas

As análises são separadas principalmente entre os resultados referentes ao tamanho de kernel utilizado para o borramento do *dataset* tratado e referente aos ângulos de cada borramento. As tabelas referentes a ambos os casos não incluem os resultados de imagens de rostos não detectados. Na Tabela 4.1, pode-se ver o resultado que mostra a quantidade de rostos reconhecidos correta e incorretamente para cada *dataset*, assim como a quantidade de imagens sem rosto detectado. No caso do modelo UnsupIMG, as células referentes à Unsup-B. representam os resultados das imagens tratadas com *blind deblurring*, enquanto Unsup S. representa os resultados obtidos por *deblurring* supervisionado.

Tabela 4.1: Quantidade de rostos reconhecidos correta e incorretamente, e quantidade de imagens sem rosto para imagens borradas não tratadas e depois de tratadas por cada modelo de *deblurring*.

Resultado	Ñ-Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
Bem-sucedido	325	445	475	375	431
Malsucedido	164	48	25	123	68
Não detectado	15	11	4	6	5

Fonte: Os autores

Tabela 4.2: Média de reconhecimentos bem-sucedidos para cada tamanho de kernel

Kernel	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
10×10	0,8749	0,9231	0,8753	0,8205	0,8429
20×20	0,7839	0,8792	0,8663	0,786	0,7834
30×30	0,5683	0,7833	0,8242	0,7155	0,7516
40×40	0,4717	0,644	0,8126	0,7003	0,711

Fonte: Os autores

4.1.1 Análise da Intensidade do Kernel

Conforme o kernel de borramento aumenta, a qualidade do reconhecimento piora em todos os casos. Isso devido ao aumento da intensidade do borramento, o que também dificulta o tratamento da imagem pelos modelos de *deblurring*. Sendo um total de 504 imagens por análise, cada kernel possui exatamente 126 imagens. Os resultados para casos bem-sucedidos, onde a pessoa foi identificada corretamente, podem ser vistos na Tabela 4.2.

Para os casos onde o reconhecimento foi falho, os resultados são mais inconsistentes. No caso de imagens não tratadas, a média de reconhecimento também diminuiu conforme o aumento do kernel. Para as imagens tratadas pelo UFPDeblur, a média aumentou junto com o tamanho do kernel. No caso do MPSL-GAN, a média também caiu, porém, especificamente no caso do kernel de tamanho 20, todas as imagens tiveram reconhecimento bem-sucedido. Já para as duas análises do modelo UnsupIMG, só houve melhora no aumento de 10×10 para 20×20, e depois, piorou também junto com o aumento do kernel. Os resultados podem ser vistos na Tabela 4.3.

Como os casos de reconhecimentos falhos são consideravelmente mais baixos para as imagens tratadas, alguns deles possuem poucos dados para se chegar a conclusões. Os resultados podem ser altamente influenciados por uma quantidade muito pequena de imagens, como pode ser visto na Tabela 4.4.

Tabela 4.3: Média de reconhecimentos malsucedidos para cada tamanho de kernel. As células com traços significam que o caso em questão não ocorreu.

Kernel	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
10×10	0,5193	—	0,4563	0,3422	0,3811
20×20	0,4984	0,2934	—	0,4336	0,466
30×30	0,4679	0,3656	0,4452	0,4094	0,4335
40×40	0,4225	0,4524	0,4221	0,4283	0,4005

Fonte: Os autores

Tabela 4.4: Quantidade de imagens com reconhecimentos falhos para cada tamanho de kernel.

Kernel	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
10×10	2	0	1	5	3
20×20	25	1	0	19	9
30×30	61	12	9	44	17
40×40	76	35	15	55	39

Fonte: Os autores

Os casos do modelo UnsupIMG para kernel 10×10 pioraram tanto os resultados de detecção quanto os de reconhecimento. Isso possivelmente porque este tamanho de kernel está fora do intervalo de tamanhos para os quais o modelo foi treinado para tratar, evidenciando a importância de vários valores distintos na hora do treinamento.

De forma geral, quanto maior a diferença entre as médias dos resultados bem-sucedidos e malsucedidos, mais podemos ter garantia que uma identificação com uma probabilidade próxima a dos resultados bem-sucedidos está correta. Por exemplo, a diferença entre ambos os valores para as imagens não tratadas no caso do tamanho de kernel 40×40 é próximo de 0,05, o que demonstra que esse borramento já é muito intenso para produzir um resultado confiável, e é possível ver que 76 das 126 imagens, ou seja, mais da metade, tiveram um reconhecimento falho, embora a média do reconhecimento bem-sucedido seja superior. Ou seja, a taxa de reconhecimentos bem-sucedidos e falhos não é proporcional à quantidade de imagens de ambos os casos, mesmo considerando as imagens onde não foram detectados rostos (4.5), e o mesmo vale também para os casos das imagens tratadas. Também é notável que a diferença entre as médias de reconhecimentos correto e falho tende a aumentar consideravelmente com o tratamento das imagens.

A Tabela 4.5 mostra a quantidade de rostos não detectados para cada valor de kernel. Os valores não costumam ser altos, mesmo para o caso das imagens não tratadas. Para o valor de kernel mais alto, 40×40, o pior caso foi o tratamento pelo modelo UFPDeblur, 10 imagens, que representa cerca de 8% do total de imagens desse tamanho de kernel, mas apenas 1,5% do total de imagens do *dataset* testado. Vale notar que a média do tamanho dos rostos nas imagens é um fator determinante nesses valores para o

Tabela 4.5: Quantidade de imagens onde não foi detectado um rosto.

Kernel	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
10×10	0	0	0	0	0
20×20	0	0	0	1	1
30×30	6	1	0	0	0
40×40	9	10	4	5	4

Fonte: Os autores

Tabela 4.6: Média de reconhecimentos bem-sucedidos para cada ângulo de kernel. Em azul, estão os resultados relativos a alguns dos ângulos com maior porcentagem de reconhecimento na coluna correspondente, enquanto em vermelho, estão alguns dos resultados com menor porcentagem para enfatizar o prejuízo ocorrido no reconhecimento durante o borramento com ângulos mais verticais.

Ângulo	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
0	0,7753	0,8446	0,8537	0,7458	0,7891
25	0,7462	0,8225	0,8632	0,7296	0,8087
50	0,7232	0,8308	0,8517	0,7647	0,7594
75	0,7099	0,8076	0,7934	0,6955	0,7349
100	0,6767	0,7949	0,873	0,7351	0,7661
125	0,7387	0,8026	0,8473	0,7954	0,7756
150	0,8377	0,8354	0,846	0,8112	0,7779
175	0,778	0,8167	0,8376	0,7981	0,774
200	0,7953	0,8468	0,869	0,7835	0,8084
225	0,7128	0,8194	0,8487	0,8218	0,7462
250	0,7105	0,8611	0,8031	0,7425	0,7378
275	0,7126	0,7883	0,8636	0,7579	0,7915
300	0,6701	0,8153	0,8369	0,8103	0,8147
325	0,773	0,8589	0,8577	0,7497	0,8087
Média	0,7401	0,8245	0,8462	0,7666	0,7784
Mediana	0,7309	0,8225	0,8502	0,7613	0,7767

Fonte: Os autores

caso do experimento.

4.1.2 Análise do Ângulo do Kernel

Cada ângulo é representado por uma média de 36 imagens. A média geral da análise, que inclui todas as 504 imagens, também está representada ao final da Tabela 4.6.

Primeiramente, é possível notar que as médias dos reconhecimentos melhoraram em todos os casos. No caso do UnsupIMG tratando as imagens por *blind deblurring*, o reconhecimento, novamente, pode ter sido prejudicado pelo tamanhos dos kernels escolhidos, porém, como pode ser visto na Tabela 4.4, a detecção sofreu uma melhora considerável em relação às imagens não tratadas, ainda que os resultados do UnsupIMG for *blind*

Tabela 4.7: Média da intensidade de kernel por pixel da área ocupada pelos rostos, decomposto em componentes x e y. K.x é a componente x do vetor que representa a direção do borramento, K.y é a componente y, BS para bem-sucedido e MS para malsucedido.

Kernel	**Não Tratadas**	**UFPDeblur**	**MSPL-GAN**	**Unsup-B.**	**Unsup-S.**
K.x BS	0,0622	0,0675	0,1003	0,0908	0,0968
K.y BS	0,0612	0,0689	0,0554	0,0466	0,0518
K.x MS	0,0962	0,1361	0,1615	0,1402	0,1440
K.y MS	0,0954	0,1192	0,1170	0,0938	0,1011
K-BS	0,0872	0,0964	0,1145	0,1020	0,1097
K-MS	0,1354	0,1809	0,1994	0,1686	0,1759

Fonte: Os autores

deblurring tenham sido os piores entre os casos de imagens tratadas. Pode-se também notar que modelos que tiveram os piores resultados no reconhecimento também foram os que apresentaram maior número de casos falhos, mas não necessariamente os que geraram mais imagens com rostos que não foram detectados. Portanto, não é possível alegar que um mesmo modelo de *deblurring* irá melhorar detecção e reconhecimento com a mesma eficiência.

Observando-se a Tabela 4.6, é possível notar que alguns parecem ser priorizados pela rede, enquanto alguns parecem ser prejudicados durante a análise. Os resultados em vermelho, alguns dos piores resultados do *dataset* analisado e representado na coluna, possuem uma tendência a estarem relacionados aos ângulos mais verticais (próximos de 90º ou 270º), enquanto os melhores resultados parecem estar mais relacionados a ângulos defasados de 25º das diagonais, embora vários dos ângulos mais diagonais (50º, 125º, 225º e 325º) também apresentem resultados melhores que a média dos outros ângulos. Durante a detecção de rostos, as coordenadas geradas pela Dlib sempre formam um quadrado exato, descartando-se, então, a hipótese de que haveria alguma deformação na imagem durante o redimensionamento para o tamanho esperado pelo *VGGFace* (224×224) que pudesse estar interferindo com alguns ângulos. Para tentar determinar o motivo desses resultados, foi-se feito uma nova bateria de testes descritos na seção próxima.

4.1.2.1 Intensidade do Kernel pela Área do Rosto

A intensidade de kernel por pixel foi analisada para cada imagem, seguindo a metodologia descrita em 3.3.3.2.

Analisando-se a Tabela 4.7, pode-se ver a média da intensidade do kernel para casos bem e malsucedidos. A média dos casos bem-sucedidos não pode ser considerada como um valor seguro, isto é, um valor que garantirá uma chance alta de um reconhe-

cimento correto, pois esse valor depende em parte das intensidades de kernel utilizadas durante o borramento. Se fossem utilizados apenas kernels de dimensões pequenas para borramento, esses valores seriam pequenos também. Porém, os valores de reconhecimento falho podem ajudar a determinar o quão intenso o kernel de borramento pode ser antes de começar a tornar os resultados inconfiáveis, ou seja, com alta probabilidade de estarem incorretos. A definição de inconfiável é subjetiva, e pode depender da aplicação e objetivos da análise.

Os resultados da intensidade de kernel mostram que as componentes y do vetor de borramento sempre possuem médias mais baixas que as componentes x, resumindo os resultados de análise por ângulo e novamente mostrando que ângulos verticais apresentam resultados mais deteriorados. É possível notar também que uma variação baixa entre as componentes não significa uma variação baixa entre os resultados na análise por ângulos.

Pode ser visto também que a intensidade de um kernel para identificações corretas (K-BS) e também incorretas (K-MS) aumenta como o tratamento das imagens para todos os modelos. O aumento de K-BS significa que todos os modelos tornaram reconhecíveis rostos com borramentos antes muito intensos para serem identificáveis pela rede, enquanto o aumento de K-MS significa que a intensidade do borramento precisa ser maior para resultar em reconhecimentos falhos.

Embora o valor de K-BS sozinho não seja necessariamente um indicativo de eficiência de um modelo, após o tratamento, o aumento da diferença entre K-BS e K-MS, assim como o aumento do próprio K-BS podem ser considerados uma evidência de melhora no reconhecimento, mas não necessariamente na detecção. A diferença entre K-BS e K-MS serve como um indicativo que existem menos chances de ocorrerem reconhecimentos incorretos com valores de probabilidades de reconhecimento semelhantes, ou seja, permite determinar melhor uma faixa de valores de probabilidade que podemos dar mais confiança de que os resultados estão corretos ou incorretos. Essa noção é menos evidente na média dos reconhecimentos incorretos por ângulos, como pode ser visto na Tabela 4.8., uma vez que a diferença entre os valores das médias é proporcionalmente menor quando comparada a diferença entre K-BS e K-MS.

Por fim, pode-se verificar na Tabela 4.9 a quantidade de imagens que obtiveram um reconhecimento incorreto, e verificar que os ângulos próximos aos mais verticais possuem uma tendência a ter mais casos falhos, à exceção do UFPDeblur, que apresentou os resultados mais estáveis.

Tabela 4.8: Média de reconhecimentos malsucedidos para cada ângulo de kernel. Células com traços indicam que o caso em questão não ocorreu.

Ângulo	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
0	0,4034	0,4412	0.3234	0,4825	0,4177
25	0,4928	0,2754	—	0,4378	0,3769
50	0,4059	0,4084	0,8033	0,4545	0,4003
75	0,4721	0,3226	0,3696	0,4244	0,426
100	0,4643	0,3162	0,5426	0,3151	0,5107
125	0,3425	0,6158	0,3317	0,4255	0,4059
150	0,5606	0,6692	—	0,4433	0,4014
175	0,4302	0,4888	0,3781	0,4384	0,4438
200	0,4746	0,3093	0,4563	0,3646	0,4565
225	0,3728	0,3712	0,3499	0,4777	0,3444
250	0,4735	0,3786	0,2697	0,3984	0,4359
275	0,4256	0,4562	0,4133	0,298	0,4312
300	0,407	0,4683	0,4901	0,4167	0,3716
325	0,5459	0,5065	0,6226	0,6006	0,4314
Média	0.4521	0.4274	0.4318	0.4189	0.4166
Mediana	0,44725	0,4248	0,4133	0,4255	0,4177

Fonte: Os autores

Tabela 4.9: Quantidade de reconhecimentos malsucedidos para cada ângulo de kernel.

Ângulo	Não Tratadas	UFPDeblur	MSPL-GAN	Unsup-B.	Unsup-S.
0	12	6	1	7	7
25	11	2	0	7	3
50	12	5	1	12	5
75	15	2	3	8	7
100	12	2	4	8	3
125	8	2	3	6	5
150	12	2	0	7	2
175	13	5	1	9	5
200	14	4	1	10	4
225	11	4	1	15	5
250	15	4	2	13	6
275	14	4	5	10	7
300	5	3	2	8	6
325	10	3	1	3	3

Fonte: Os autores

4.2 Análise das Imagens Tratadas com Aumento de Área da Detecção

Para tentar determinar a razão de alguns ângulos serem priorizados e outros prejudicados, foram feitos testes onde a área ocupada por rostos estabelecida pela Dlib foi forçadamente ampliada antes do redimensionamento esperado pelo *VGGFace*. O objetivo desse teste era causar uma deformação na imagem do rosto e avaliar o impacto nos resultados por ângulo. Para o caso das imagens não tratadas e das tratadas pelo UFPDeblur, que aceita imagens de quaisquer dimensões, o aumento ocorreu em tempo real na hora da análise pela rede. Para os outros casos, como a imagem precisa ser redimensionada para 128×128, é necessário que esse aumento de área seja feito antes do *deblurring*. Portanto, várias versões de *datasets* redimensionados foram gerados.

Os testes foram feitos com aumento de área sempre proporcional à area estabelecida pela Dlib. Foram feitos três tipos de testes: foi aumentada a área horizontalmente se alterando a largura, verticalmente se aumentando a altura e ambos ao mesmo tempo para cada proporção, sempre nos dois sentidos de cada direção. Por exemplo, assumindo que *xmin* e *xmax* são as duas coordenadas que delimitam a área do rosto na imagem no eixo x, um aumento horizontal de 10% significa calcular 10% do valor absoluto da distância (em *pixels*) entre *xmin* e *xmax* e adicionar metade desse valor a *xmax* e subtrair metade desse valor de *xmin* . O cálculo é análogo para o aumento vertical.

Não foi permitido que os aumentos de área fosse além dos limites do tamanho da imagem, o que pode contribuir para delimitar o aumento da média de reconhecimento com o aumento da área de detecção.

Vale ressaltar que, devido às dimensões esperadas pelo *VGGFace*, os aumentos de área inevitavelmente vão causar perda de informação da imagem, ao mesmo tempo que incluem novas informações devido a inclusão de partes da cabeça da pessoa.

Os testes com aumentos exclusivamente horizontais ou verticais foram feitos apenas com as imagens não tratadas, pois o objetivo do teste era apenas verificar se ocorreria uma melhora nos resultados pela inclusão de *features* e verificar se os mesmos ângulos ainda seriam priorizados ou prejudicados. Os testes com aumentos para ambas as dimensões foram também realizados com imagens tratadas, este com o objetivo de melhorar o resultado já obtido anteriormente com *deblurring*.

Tabela 4.10: Média de reconhecimentos bem-sucedidos para cada ângulo de kernel com aumento horizontal de área da detecção.

Ângulo	**+0%**	**+10%**	**+30%**	**+50%**	**+70%**
0	0,7753	0,7585	0,7602	0,7866	0,8096
25	0,7462	0,7322	0,7754	0,7662	0,7874
50	0,7232	0,7081	0,7477	0,7603	0,7552
75	0,7099	0,733	0,7602	0,7677	0,7161
100	0,6767	0,7137	0,7509	0,7626	0,7331
125	0,7387	0,7135	0,7699	0,7871	0,8117
150	0,8377	0,8264	0,795	0,7817	0,7916
175	0,778	0,7311	0,7393	0,7737	0,7961
200	0,7953	0,7376	0,7711	0,806	0,794
225	0,7128	0,6918	0,7331	0,7421	0,7289
250	0,7105	0,7186	0,7444	0,7777	0,7551
275	0,7126	0,7318	0,7463	0,7545	0,7146
300	0,6701	0,7293	0,7738	0,7891	0,7703
325	0,773	0,767	0,7509	0,742	0,7322
Média	0,7401	0.7359	0.7592	0.7715	0.7659
Mediana	0,7309	0,73145	0,7555	0,7707	0,7627

Fonte: Os autores

4.2.1 Aumento Horizontal da Área de Detecção

Para cada imagem, foram testados aumentos horizontais de 10%, 30%, 50% e 70%. Os resultados estão na Tabela 4.10. Os resultados indicam que houve uma melhora no reconhecimento até o aumento de 50%. Com 70%, a perda de informação, a deformação causada pelo redimensionamento e possíveis limitações pelo tamanho da imagem foram mais nocivos que os ganhos pela adição de partes do rosto. Essa melhora possivelmente ocorre pelo fato de que, embora o foco seja o rosto, tanto os modelos de *deblurring* quanto o *VGGFace* e o classificador são treinados com cabeças inteiras, e aqui está um indicativo de que estas *features* cortadas podem prejudicar o reconhecimento em algum nível.

Conforme pode ser visto na Tabela 4.11, até o aumento de 30%, a quantidade de rostos com reconhecimento bem-sucedido aumentou em cerca de 10%. O aumento de 50% teve um impacto negativo em relação ao aumento de 30%, porém por uma única imagem de diferença, mas é um indicativo que, a partir valor de aumento, os ganhos podem não compensar as perdas.

Houve alguma mudança nos ângulos com melhores resultados, porém as alterações parecem inconsistentes, oscilando entre valores mais altos e mais baixos do que os valores dos resultados das imagens não tratadas conforme a expansão horizontal aumenta,

Tabela 4.11: Quantidade de rostos com reconhecimentos corretos e falhos para aumento de área horizontal

Tipo de resultado	**+0%**	**+10%**	**+30%**	**+50%**	**+70%**
Bem-sucedido	325	342	358	357	360
Malsucedido	164	147	131	132	129
Rosto não detectado	15	15	15	15	15

Fonte: Os autores

Tabela 4.12: Média de reconhecimentos bem-sucedidos para cada ângulo de kernel com aumento vertical de área da detecção.

Ângulo	**+0%**	**+10%**	**+30%**	**+50%**	**+70%**
0	0,7753	0,7184	0,7716	0,727	0,6975
25	0,7462	0,7317	0,7532	0,7317	0,7228
50	0,7232	0,7802	0,7786	0,7395	0,7332
75	0,7099	0,7393	0,7477	0,7584	0,7418
100	0,6767	0,704	0,7504	0,7525	0,7699
125	0,7387	0,7368	0,7795	0,8096	0,8045
150	0,8377	0,8284	0,8783	0,8225	0,8176
175	0,778	0,743	0,7908	0,7368	0,6797
200	0,7953	0,732	0,7373	0,7361	0,7152
225	0,7128	0,7449	0,7495	0,7247	0,7019
250	0,7105	0,7588	0,7304	0,7504	0,7349
275	0,7126	0,7084	0,7578	0,7797	0,7207
300	0,6701	0,7044	0,7339	0,7857	0,7781
325	0,773	0,7724	0,8301	0,7969	0,7962
Média	0,7401	0,7420	0,7702	0,7608	0,7436
Mediana	0,7309	0,7380	0,7555	0,7514	0,7340

Fonte: Os autores

embora os resultados dos ângulos para um mesmo aumento apresentem uma variação menor entre si.

Quanto à detecção, não houve nenhuma mudança nos resultados. Não era esperado que ocorresse melhora, uma vez que a área aumentada é baseada na área dada pelo próprio detector, e do ponto de vista da detecção, a deformação da imagem só causa perda de informação, sem nenhum ganho.

4.2.2 Aumento Vertical da Área de Detecção

Os mesmos valores testados com o aumento horizontal foram testados para o aumento vertical. Os resultados podem ser vistos na Tabela 4.12.

O valor da mediana para o aumento de 30% foi idêntico para ambos os tipos de aumento. Ambos os tipos de aumento tiveram uma média máxima de probabilidade de

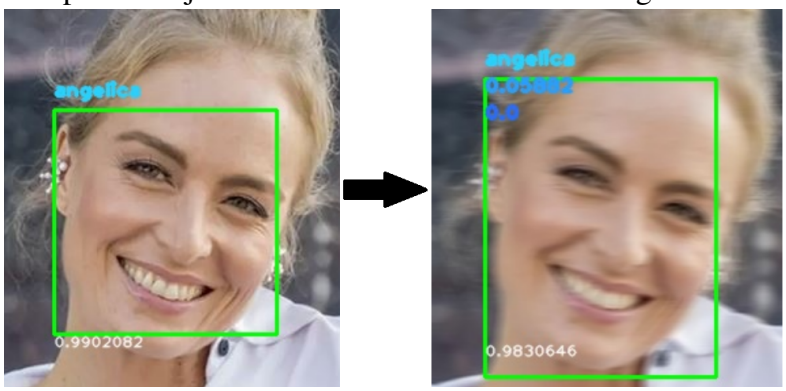
Tabela 4.13: Quantidade de rostos com reconhecimentos corretos e falhos para aumento de área vertical

Tipo de resultado	**+0%**	**+10%**	**+30%**	**+50%**	**+70%**
Bem-sucedido	325	357	373	377	394
Malsucedido	164	132	116	112	95
Rosto não detectado	15	15	15	15	15

Fonte: Os autores

identificação de um valor em torno de 0,77. Este valor foi alcançado mais rapidamente pelo aumento vertical, possivelmente porque a porção da cabeça cortada verticalmente pelo detector seja maior que o horizontal, uma vez que a área quadrada estabelecida pela Dlib use a altura do rosto detectado para estabelecer a largura. Um exemplo do aumento pode ser visto na Figura 4.2. Mesmo a imagem borrada não sendo tratada, com o aumento vertical de 30%, a probabilidade de identificação dada pela rede caiu em apenas 0,7%.

Figura 4.2: Exemplo da análise de uma imagem nítida e de uma imagem borrada não tratada com aumento da área vertical em 30%. O dois valores no topo do retângulo verde são, de cima para baixo, as componentes horizontal e vertical do vetor que representa a direção do kernel. O segundo valor em 0.0 indica que não há componente vertical, sendo portando, um vetor com ângulo de 0º. O valor de cor branca indica a probabilidade dada pela rede de ser a pessoa cujo nome está escrito acima do retângulo.



Fonte: Os autores

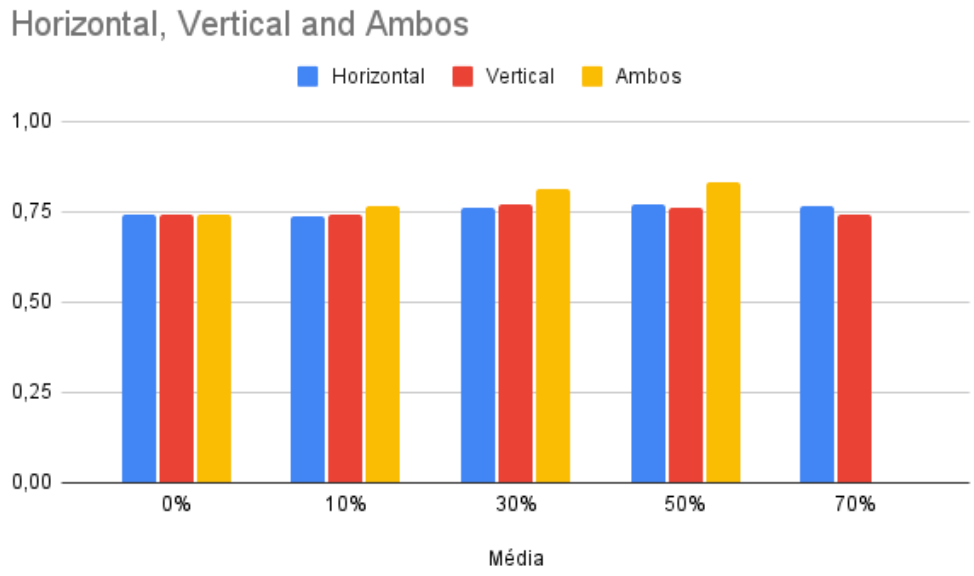
Outra evidência de que o aumento vertical favoreceu mais o reconhecimento pode ser visto na Tabela 4.13. Diferentemente do aumento horizontal, o aumento vertical melhorou o reconhecimento até com 70%. Embora a média final tenha diminuído, o aumento nos reconhecimentos corretos foi considerável. E novamente, não houve alteração na detecção. Porém, comparando, com a Tabela 4.1, os resultados obtidos com aumento vertical de 70% superou o resultado obtido pelo tratamento das imagens pelo modelo UnsupIMG utilizando *blind deblurring*.

4.2.3 Aumentos Horizontal e Vertical Simultaneamente

Após comprovado que o aumento de área de detecção pode melhorar o reconhecimento, foi testado o aumento de ambas as dimensões, a fim de preservar as formas da imagem no redimensionamento e verificar se os ganhos em adição de *features* superam as perdas de informação.

Para este caso, foi executado *deblurring* das imagens. Os aumentos agora foram de 10%, 20%, 30%, 40% e 50%. Um resumo gráfico dos resultados com aumento de área pode ser visto na Figura 4.3.

Figura 4.3: Comparação de todos os resultados com aumentos de área para imagens não tratadas. Pode-se notar que a barra amarela (ambas as dimensões) aumenta mais do que a soma dos aumentos vertical e horizontal individualmente.



Fonte: Os autores

A Tabela 4.14 já demonstra que as médias superaram por um valor considerável os valores obtidos por ambos os aumentos aplicados separadamente. A melhora na média do reconhecimento por cada um ficou em torno dos 3% em relação às imagens do tamanho dado pela detecção da Dlib, enquanto os dois aumentos juntos, a melhora ficou em torno dos 10%, durante o caso de aumento de 50%. Isso evidencia que, para a rede de análise, é mais importante se ter *features* mais completas e mais deterioradas do que menos *features* e mais detalhadas. Mesmo sem nenhum tratamento, como pode ser visto na Tabela 4.6, o único modelo que supera essa média é o MSPL-GAN, quando não há aumento de área.

Como pode ser visto na Tabela 4.15, as falhas na detecção continuaram iguais. O aumento em 10% já apresentou um aumento significativo na quantidade de casos reconhe-

Tabela 4.14: Média de reconhecimentos bem-sucedidos para cada ângulo de kernel com aumentos horizontal e vertical de área da detecção antes do *deblurring*.

Ângulo	+0%	+10%	+20%	+30%	+40%	+50%
0	0,7753	0,7902	0,8165	0,8374	0,8503	0,8543
25	0,7462	0,7568	0,8026	0,8054	0,8308	0,8345
50	0,7232	0,7699	0,8095	0,8437	0,8341	0,853
75	0,7099	0,7682	0,7092	0,7818	0,8297	0,8102
100	0,6767	0,72	0,7676	0,752	0,7759	0,8153
125	0,7387	0,7434	0,773	0,8076	0,8258	0,8282
150	0,8377	0,7959	0,8328	0,877	0,8407	0,8431
175	0,778	0,7854	0,8095	0,8373	0,8509	0,844
200	0,7953	0,7637	0,8234	0,8323	0,8508	0,8516
225	0,7128	0,7889	0,7801	0,8068	0,8282	0,8482
250	0,7105	0,775	0,7513	0,7875	0,7798	0,7847
275	0,7126	0,6984	0,7591	0,7574	0,7829	0,8007
300	0,6701	0,7454	0,7649	0,8188	0,8143	0,8377
325	0,773	0,806	0,8237	0,8418	0,8111	0,8081
Média	0,7401	0.7664	0,7900	0,8149	0,8224	0,8301
Mediana	0,7309	0,7690	0,7913	0,8132	0,82895	0,8361

Fonte: Os autores

Tabela 4.15: Quantidade de rostos com reconhecimentos corretos e falhos para aumentos de área horizontal e vertical

Tipo de resultado	+0%	+10%	+20%	+30%	+40%	+50%
Bem-sucedido	325	362	383	389	394	392
Malsucedido	164	127	106	100	95	97
Rosto não detectado	15	15	15	15	15	15

Fonte: Os autores

cidos corretamente, e o aumento de 40% já foi o suficiente para igualar os resultados aos resultados obtidos pelo aumento vertical em 70%. O valor de aumento em 50% causou uma pequena piora nos resultados, o que também aconteceu com o aumento horizontal mas não com o vertical. Possivelmente, do mesmo jeito que o aumento da área em ambas as dimensões cause uma melhora superior a soma da melhora dos aumentos de cada dimensão individualmente, o fato do rosto ter sido diminuído nas duas dimensões também pode sofrer uma piora mais acentuada devido ao redimensionamento.

Apesar da melhora do reconhecimento, o aumento da área do rosto não resolve inteiramente o prejuízo causado pelos ângulos mais verticais ao reconhecimento, que continuam sendo prejudicados em algum nível em relação aos outros ângulos. Isso indica que o corte de *features* faciais causadas pela área estabelecida da Dlib não é inteiramente responsável por estes resultados, que podem vistos na Tabela 4.16. Do mesmo jeito que os melhores resultados pareciam estar relacionados a ângulos defasados de 20º das diagonais, os piores também parecem estar defasados de 20º dos mais verticais. Isso pode se

Tabela 4.16: Quantidade de reconhecimentos malsucedidos para cada ângulo de kernel com aumentos horizontal e vertical de área da detecção antes do *deblurring*. Os ângulos mais verticais estão em vermelho.

Ângulo	+0%	+10%	+20%	+30%	+40%	+50%
0	12	7	7	7	7	6
25	11	7	7	6	7	8
50	12	13	10	10	7	7
75	15	15	12	10	11	11
100	12	9	10	9	8	7
125	8	4	2	1	1	3
150	12	9	6	7	4	4
175	13	7	5	7	7	7
200	14	8	9	8	8	9
225	11	9	6	7	8	7
250	15	16	12	13	12	10
275	14	11	10	8	8	9
300	5	5	3	2	2	4
325	10	7	7	5	5	5

Fonte: Os autores

dever a inclinação da cabeça nas pessoas na maioria nas imagens.

4.2.3.1 Resultados após o deblurring

Para resumir os resultados após o tratamento das imagens por todos os modelos com todas as porcentagens de aumento de área, a Tabela 4.17 mostra apenas a média final do reconhecimento, agora sem foco no resultado por ângulo, e suas variações podem ser conferidas na Figura 4.4. Os resultados das imagens não tratadas foram os únicos que se mantiveram crescentes durante todos os aumentos de área. Proporcionalmente, as imagens não tratadas tiveram a melhora mais acentuada, superando os tratamentos das imagens em alguns casos. Do ponto de vista da média do reconhecimento, concluí-se que não se pode garantir que um modelo irá gerar melhores resultados ao se unir o seu tratamento com o aumento de área, e cada modelo possui um limite de aumento particular com o qual esses resultados poderão ser aprimorados.

A Tabela 4.18 mostra que todos os *datasets*, após apresentarem uma piora de resultados durante um aumento, não necessariamente o próximo aumento irá causar uma piora mais acentuada. Isso pode se dever a diversos fatores, como a inclusão e deterioração de *features*, treinamento dos modelos e suas próprias técnicas de *deblurring*.

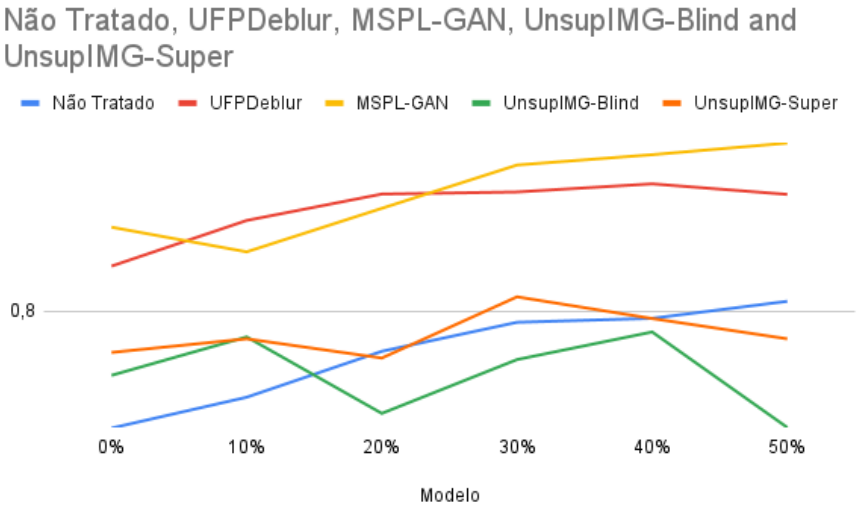
O modelo UnsupIMG por *blind deblurring* não apresentou desempenho satisfatório com aumento de área, tendo pior desempenho que a maioria dos casos de imagens não

Tabela 4.17: Média de reconhecimentos bem-sucedidos para cada aumento horizontal e vertical de área da detecção após do *deblurring*.

Modelo	+0%	+10%	+20%	+30%	+40%	+50%
Ñ-Tratado	0,7401	0.7555	0.7790	0.7942	0.7963	0.8054
UFPDeblur	0,8245	0,8500	0,8651	0,8663	0,8710	0,8649
MSPL-GAN	0,8462	0,8324	0,8569	0,8820	0,8881	0,8950
UnsupIMG-Blind	0,7666	0,7865	0,7474	0,7747	0,7891	0,7404
UnsupIMG-Super	0,7784	0,7855	0,7755	0,8078	0,7962	0,7855

Fonte: Os autores

Figura 4.4: Variação das médias ao longo dos aumentos de área de ambas as dimensões para imagens tratadas por *deblurring*.



Fonte: Os autores

tratadas, tanto em média quanto em quantidade de reconhecimentos. Porém, os resultados melhoraram a detecção significativamente. Quando utilizou *deblurring* supervisionado, o UnsupIMG apresentou uma melhora significativa na quantidade de reconhecimentos corretos em relação ao *blind deblurring*, mas aumentos acima de 20% já causaram deterioração dos resultados. Os resultados do UnsupIMG podem ter sido influenciados negativamente pelo tamanhos de kernel utilizados nos borramentos das imagens. O MPSL-GAN teve o desempenho mais eficientes em todos os aumentos, embora com leve oscilação entre piora e melhora durante o aumento da área. O UFPDeblur foi o que apresentou menos variação em cada aumento, mesmo melhorando em todos os casos exceto com aumento de 40%, e também foi o único modelo de *deblurring* que não apresentou melhora na detecção de rosto com os aumentos de área. O UFPDeblur é o único modelo de *deblurring* que não foi treinado com rostos, o que pode ajudar a explicar esse fenômeno na detecção. É interessante notar que não houve piora na detecção em nenhum caso mostrado na Tabela 4.18.

Tabela 4.18: Quantidade de reconhecimentos bem-sucedidos (primeiro valor) e quantidade de imagens sem rosto (segundo valor) para cada aumento horizontal e vertical de área da detecção após do *deblurring*. Os números estão no formato (primeiro valor)/(segundo valor)

Modelo	+0%	+10%	+20%	+30%	+40%	+50%
Ñ-Tratado	325/15	362/15	383/15	389/15	394/15	392/15
UFPDeblur	445/11	450/11	452/11	457/11	455/11	459/11
MSPL-GAN	475/4	470/0	485/0	493/0	489/0	491/0
UnsupIMG-Blind	375/6	379/2	377/2	379/0	379/0	376/0
UnsupIMG-Super	431/5	439/0	445/0	437/0	414/0	413/0

Fonte: Os autores

4.3 Análise das Imagens Tratadas com Remoção de Fundo

Como último teste, foi testada a remoção de fundo das imagens de rostos a serem borradas. Isso para testar o impacto no reconhecimento tendo apenas o rosto em si, como se fosse um *prior* para a análise, e também para verificar se o fundo poderia ser responsável pelas oscilações entre melhora e piora dos resultados com aumentos de área. A remoção do fundo significa substituir o fundo por uma única cor, pois num caso real, não haverá um rosto com um fundo transparente. Foi escolhida a cor preta para os testes.

Para remover o fundo, foi utilizada a ferramenta *SegmentAnything* (SAM) (KIRILLOV et al., 2023) disponibilizada pela empresa Meta. A ferramenta utiliza máscaras aprendidas com o treinamento de mais de 1 bilhão de imagens para detectar diferentes objetos em imagem fornecida, isso a partir de pontos decididos pelo usuário, e assim, remover esses objetos. Pode-se adicionar pontos de adição, para manter o objeto, ou remoção, para excluir o objeto (pessoas são consideradas objetos para o SAM). Um exemplo pode ser visto na Figura 4.5.

Após a remoção do fundo das 9 imagens nítidas, o borramento foi novamente aplicado com mesmas direções e intensidades dos outros testes, e os *datasets* de tamanho normal e redimensionado (128×128) foram novamente confeccionados a partir dos mesmos passos já descritos em sessões anteriores.

4.3.1 Aumentos Horizontal e Vertical Simultaneamente

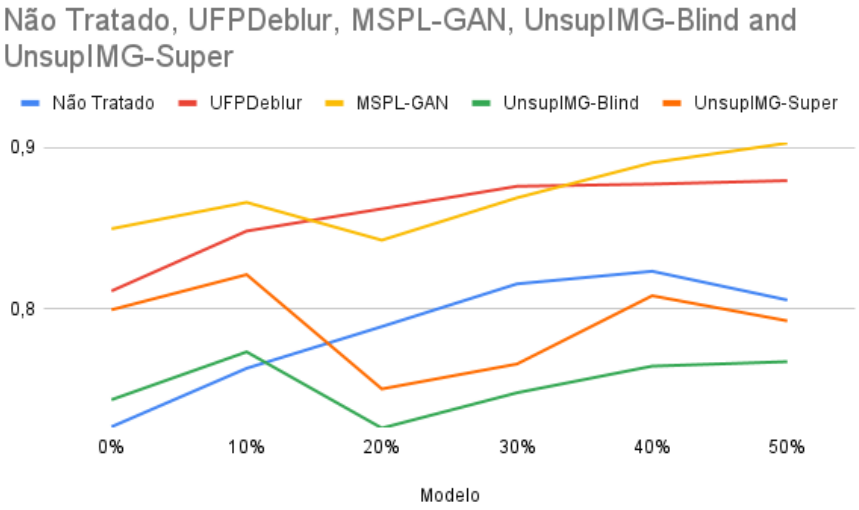
Foi feita a mesma análise com aumento de área, agora para as imagens com fundo removido. Os resultados podem ser vistos na Tabela 4.19. Os resultados com o fundo mantido também estão na tabela para facilitar a comparação, e suas variações podem ser

Figura 4.5: Exemplo de segmentação da ferramenta SAM. Os pontos verdes servem para manter o objeto, enquanto os rosados servem para excluir o objeto. O rosto foi recortado e foi gerada uma imagem com fundo transparente. O fundo preto foi adicionado manualmente. A imagem da direita é um exemplo de imagem nítida que foi submetida ao borramento para gerar o novo *dataset*.

Fonte: Os autores

conferidas na Figura 4.6.

Figura 4.6: Variação das médias ao longo dos aumentos de área de ambas as dimensões para imagens tratadas por *deblurring* e com fundo removido.



Fonte: Os autores

Os resultados para as imagens não tratadas sem fundo foram cerca de 2% melhor, sendo que a única piora no caso em que não houve aumento de área. Para o UFPDeblur, a melhora ocorreu a partir do aumento de 30%, mas diferentemente da versão com fundo, a imagem sem fundo não piorou do aumento de 40% para 50%. O MSPL-GAN sofreu mais oscilações entre pioras e melhoras, tornando-se mais instável do que quando tratou as imagens com fundo, mas ainda apresentou o melhor resultado de todos, sendo o seu tratamento com aumento de 50% na área o único caso que obteve uma média de reco-

Tabela 4.19: Média de reconhecimentos bem-sucedidos para cada aumento horizontal e vertical de área da detecção de imagens com e sem fundo.

Modelo	+0%	+10%	+20%	+30%	+40%	+50%
Casos sem imagem de fundo						
Ñ-Tratado	0,7345	0,7664	0,7900	0,8149	0,8224	0,8301
UFPDeblur	0,8107	0,8468	0,8606	0,8748	0,8762	0,8784
MSPL-GAN	0,8482	0,8646	0,8412	0,8675	0,8899	0,9027
UnsupIMG-Blind	0,7491	0,7756	0,7338	0,7529	0,7676	0.7701
UnsupIMG-Super	0.7997	0,8204	0,7550	0,7688	0,8079	0,7933
Casos com imagem de fundo						
Ñ-Tratado	0,7401	0.7555	0.7790	0.7942	0.7963	0.8054
UFPDeblur	0,8245	0,8500	0,8651	0,8663	0,8710	0,8649
MSPL-GAN	0,8462	0,8324	0,8569	0,8820	0,8881	0,8950
UnsupIMG-Blind	0,7666	0,7865	0,7474	0,7747	0,7891	0,7404
UnsupIMG-Super	0,7784	0,7855	0,7755	0,8078	0,7962	0,7855

Fonte: Os autores

nhecimento bem-sucedido acima de 90%. Apesar disso, a melhora do MSPL-GAN com a remoção de fundo foi pouco significativa. O UnsupIMG por *blind deblurring* sofreu piora em todos os casos com a remoção de fundo exceto pelo aumento de 50% de área. É interessante notar que este modelo sofreu piora e melhora exatamente nos mesmos aumentos conforme a área de detecção foi sendo aumentada, tanto para os casos com fundo removido como para os casos com fundo presente, o que indica que, para o UnsupIMG, o fundo não é responsável pelas oscilações nas médias de reconhecimento. Ao utilizar *deblurring* supervisionado, o UnsupIMG apresentou melhoras e pioras ao se remover o fundo de uma maneira quase aleatória, mostrando que, para o reconhecimento facial, este modelo não se beneficia da remoção de fundo.

De forma geral, a remoção de fundo é benéfica quando não se há tratamento das imagens borradas. O impacto na média dos reconhecimentos bem-sucedidos foi baixa, tanto positiva quanto negativamente. Esse resultado pode se dever ao fato de tanto os modelos de *deblurring* quanto à rede de análise não terem sido treinados exclusivamente com imagens sem fundo, mas também à área de detecção.

Quanto à quantidade de imagens reconhecidas e rostos não detectados, os resultados estão na Tabela 4.20.

Analisando-se estes resultados, pode-se ver que alguns casos onde a média de reconhecimento diminuiu incluíram também mais imagens com reconhecimento bem-sucedido, o que significa que essas imagens, agora reconhecidas, provavelmente se encontram no limiar entre bem e malsucedido devido à sua deterioração pelo borramento, ou seja, embora reconhecíveis, o reconhecimento ainda está sendo consideravelmente

Tabela 4.20: Quantidade de imagens com reconhecimento bem-sucedido (primeiro valor) e imagens sem rosto detectados (segundo valor) para cada aumento horizontal e vertical de área da detecção de imagens com e sem fundo. Os números estão no formato (primeiro valor)/(segundo valor)

Modelo	**+0%**	**+10%**	**+20%**	**+30%**	**+40%**	**+50%**
Casos sem imagem de fundo						
Ñ-Tratado	348/19	384/19	395/19	396/19	399/19	400/19
UFPDeblur	436/8	448/8	451/8	449/8	446/8	452/8
MSPL-GAN	476/7	444/5	483/2	493/2	484/2	494/2
UnsupIMG-Blind	380/6	386/6	359/3	388/2	390/2	381/3
UnsupIMG-Super	424/7	413/4	431/3	444/2	430/2	441/2
Casos com imagem de fundo						
Ñ-Tratado	325/15	362/15	383/15	389/15	394/15	392/15
UFPDeblur	445/11	450/11	452/11	457/11	455/11	459/11
MSPL-GAN	475/4	470/0	485/0	493/0	489/0	491/0
UnsupIMG-Blind	375/6	379/2	377/2	379/0	379/0	376/0
UnsupIMG-Super	431/5	439/0	445/0	437/0	414/0	413/0

Fonte: Os autores

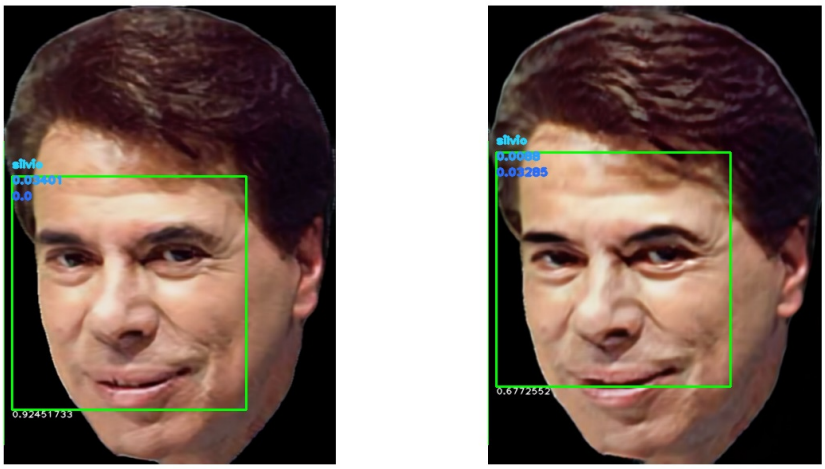
prejudicado pelo borramento, o que pode estar diminuindo a média final da análise.

O UnsupIMG, para ambas as suas modalidades, teve a detecção prejudicada pela remoção de fundo, e os casos onde mais imagens foram reconhecidas corretamente dependem do aumento de área de detecção aplicado. O MPSL-GAN teve a detecção prejudicada também, embora na média dos casos de cada aumento, o reconhecimento tenha sido levemente favorecido pela remoção do fundo. O UFPDeblur melhorou levemente a detecção, mas o reconhecimento foi prejudicado em todos os aumentos de área, possivelmente pelo fato de o modelo ter sido treinado com borramento real usando paisagens de vários tipos, exatamente o que foi removido das imagens.

As imagens não tratadas tiveram a detecção de rosto piorada e o reconhecimento melhorado, o que é um caso interessante considerando que a rede da Dlib é treinada para localizar pontos que caracterizam um rosto humano, e portanto, teoricamente, o fundo não deveria impactar na detecção. Isso indica que alguns modelos de *deblurring* podem ter tido sua detecção prejudicada pela Dlib. Foi constatado que a área de detecção tende a diminuir em images tratatas com *deblurring* quando o fundo é removido. A diminuição possivelmente ocorre pelo fato de um fundo se misturar com a face detectada, fazendo com que o detector facial perceba o borramento total como um rosto maior do que realmente é, e aumenta a área de detecção, caindo nos casos de aumento de área. Um exemplo pode ser visto na Figura 4.7. É possível notar que um *kernel* de 75º, consideravelmente mais vertical que um *kernel* de 0º, causou diminuição da área de detecção. Assumindo

que a hipótese sobre o rosto se misturar com incentive o aumento da área seja verdadeira, então o *kernel* com ângulo mais vertical é mais nocivo ao reconhecimento e à detecção não por ser mais prejudicial em sua natureza, mas porque o rosto possui uma altura maior que a largura. Portanto, um borramento horizontal causaria um "aumento"do rosto em maior intensidade, auxiliando a detecção mais do que um borramento vertical.

Figura 4.7: Diminuição de área de detecção causada por um kernel de 75º em relação a um kernel com ângulo 0º. Imagens tratadas com o modelo UFPDeblur.



Fonte: Os autores

5 CONCLUSÕES

Este trabalho foca na análise de reconhecimento facial para imagens borradas com borramento por movimento (*motion blur*) e a melhora na eficácia dessa análise, e também da detecção após o tratamento dessas imagens por modelos de *deblurring*.

Foi comprovado que o tratamento das imagens em si tem um impacto positivo relevante no reconhecimento e na detecção facial. Além disso, a análise dos ângulos de borramento de kernel mostram que, de forma geral, os kernels com ângulos mais verticais são mais nocivos para ambos reconhecimento e detecção. É possível constatar o prejuízo que os ângulos verticais causam também, de forma resumida, através da comparação de valores das componentes da intensidade de kernel por área de rosto (K.x e K.y), e este prejuízo também é diferente dependendo do modelo de *deblurring* utilizado.

Também foram feitos testes adicionais combinados com o *deblurring* para avaliar o impacto nos resultados. Ficou comprovado que a área de detecção dada por uma rede neural não necessariamente irá gerar o melhor resultado de reconhecimento, e testes com o aumento dessa área mostraram que é mais importantes incluir mais *features* de um rosto do que termos menos *features* de menor qualidade (devido a dimensão da imagem em questão). Porém, ao se aplicar o aumento de área com imagens tratadas, a melhora depende do modelo utilizado, pois o tratamento de modelo em conjunto com cada área de aumento gera um resultado diferente durante a análise pela rede neural, e o reconhecimento e a detecção não são afetados igualmente em cada caso.

Um último teste envolvendo remoção de fundo da imagem para tentar salientar os rostos para as redes e para os modelos de *deblurring*, também combinado com aumento da área de detecção de rosto, se demonstrou útil para casos muito particulares, em geral causando algumas melhoras e alguns prejuízos nas imagens tratadas por um mesmo modelo. Mas de forma geral, pouco eficiente quando comparado ao *deblurring* e ao aumento de área de detecção. Também foi constatada a possibilidade da rede de detecção ser responsável pela baixa melhora destes casos.

Este trabalho é limitado pela necessidade de imagens nítidas para comparação dos resultados. Embora o reconhecimento ainda possa ser feito se treinando o classificador com imagens nítidas diferentes das borradas, não seria possível gerar borramento por movimento com outros ângulos. Sem versões diferentes da mesma imagem, as comparações de resultados não seriam válidas devido à incosistência de ângulos, perspectivas e pessoas nas imagens.

Possíveis continuações do estudo poderiam estar relacionadas com a intensidade de kernel por área de rosto, numa tentativa de determinar qual a intensidade máxima que um kernel poderia ter antes de causar falhas na detecção. Já está claro que este valor vai depender do modelo de *deblurring* em questão. Uma vez determinado, poderia ser feita uma análise relativa a distâncias de um rosto em relação à câmera, e ver se as falhas de detecção se manteriam constantes em várias distâncias diferentes. Outra possibilidade seria juntar os testes realizados nesse trabalho com a exploração da componente temporal, ou seja, a componente que relaciona diferentes *frames* de um vídeo e analisa a mudança no deslocamento de *features* de um *frame* para outro, o que é uma informação útil para melhorar o *deblurring*.

REFERÊNCIAS

ABADI, M. et al. **TensorFlow: A system for large-scale machine learning**. 2016.

ANDRYCHOWICZ, M. et al. **Learning to learn by gradient descent by gradient descent**. 2016.

ANWAR, S.; HUYNH, C. P.; PORIKLI, F. **Image deblurring with a class specific prior**. 2018. 2112-2130 p.

ASTAWA, I. N. G. et al. Face images classification using vgg-cnn. **Knowledge Engineering and Data Science**, v. 4, n. 1, p. 49–54, 2021. ISSN 2597-4637. Available from Internet: <https://journal2.um.ac.id/index.php/keds/article/view/19547>.

BAHAT, Y.; EFRAT, N.; IRANI, M. Non-uniform blind deblurring by reblurring. In: **IEEE**. [S.l.: s.n.], 2017. p. 3286–3294.

BEN-EZRA, M. **Light Efficient Flutter Shutter**. 2015.

BRADSKI, G. The OpenCV Library. **Dr. Dobb's Journal of Software Tools**, 2000.

BUI, H. M. et al. Using grayscale images for object recognition with convolutional-recursive neural network. In: **2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)**. [S.l.: s.n.], 2016. p. 321–325.

CHAKRABARTI, A. A neural approach to blind motion deblurring. In: **IEEE**. [S.l.: s.n.], 2016. p. 221–235.

CHAOYANG, Z.; GUANGJUN, L.; QIFENG, Q. On psf for lined motion blurred image. In: **2008 27th Chinese Control Conference**. [S.l.: s.n.], 2008. p. 357–360.

CHO SEO-WON JI, J.-P. H. S.-J.; JUNG, S.-W.; KO., S.-J. Rethinking coarse-to-fine approach in single image deblurring. In: **IEEE**. [S.l.: s.n.], 2021. p. 4641–4650.

CHOLLET, F. **keras**. [S.l.]: GitHub, 2015. <https://github.com/fchollet/keras>.

DASGUPTA, P. B. Comparative analysis of non-blind deblurring methods for noisy blurred images. **International Journal of Computer Trends and Technology**, Seventh Sense Research Group Journals, v. 70, n. 3, p. 1–8, mar. 2022. ISSN 2231-2803. Available from Internet: <http://dx.doi.org/10.14445/22312803/IJCTT-V70I3P101>.

DINH, L.; KRUEGER, D.; BENGIO, Y. Nice: Non-linear independent components estimation. In: **IEEE**. [S.l.: s.n.], 2014. ArXix preprint arXiv:1410.8516.

FANG, Z. et al. Self-supervised non-uniform kernel estimation with flow-based motion prior for blind image deblurring. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2023. p. 18105–18114.

HACOHEN, Y.; E, S.; D., L. Deblurring by example using dense correspondence. In: **IEEE**. [S.l.: s.n.], 2013. p. 2384–2391.

HORé, A.; ZIOU, D. Image quality metrics: Psnr vs. ssim. In: **2010 20th International Conference on Pattern Recognition**. [S.l.: s.n.], 2010. p. 2366–2369.

HUANG, G. et al. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. **Tech. rep.**, 10 2008.

KARAALI, A.; JUNG, C. R. Svbr-net: A non-blind spatially varying defocus blur removal network. In: **2022 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2022. p. 566–570.

KARNEWAR, A.; WANG, O. Msg-gan: Multi-scale gradients for generative adversarial networks. In: **IEEE**. [S.l.: s.n.], 2020. p. 7799–7808.

KARRAS, T. A. T.; LAINE, S.; LEHTINEN, J. **Progressive growing of GANs for improved quality, stability, and variation**. 2017. Http://arxiv.org/abs/1710.10196. 2023-08-28.

KING, D. E. Dlib-ml: A machine learning toolkit. In: . [S.l.: s.n.], 2009. v. 10, p. 1755–1758.

KIRILLOV, A. et al. **Segment Anything**. 2023.

KULIN TARIK KAZAZ, E. d. P. M.; MOERMAN, I. **A Survey on Machine Learning-Based Performance Improvement of Wireless Networks: PHY, MAC and Network Layer**. 2021. Https://www.mdpi.com/2079-9292/10/3/318B26-electronics-10-00318.

KUPYN VOLODYMYR BUDZAN, M. M. O.; MISHKIN, D.; MATAS, J. Blind motion deblurring using conditional adversarial networks. In: **IEEE**. [S.l.: s.n.], 2018. p. 8183–8192.

LECUN, Y. B. Y.; HINTON, G. Deep learning. In: **Nature**. [S.l.: s.n.], 2015. p. 436–444.

Lee, T. B.; Jung, S. H.; Heo, Y. S. Progressive semantic face deblurring. **IEEE Access**, v. 8, p. 223548–223561, 2020.

LIANG JIEZHANG CAO, G. S. K. Z. J.; GOOL, L. V.; TIMOFTE, R. Image restoration using swin transformer. In: **IEEE**. [S.l.: s.n.], 2021. p. 1833–1844.

LIU, Q.; WU, Y. Supervised learning. 01 2012.

MCCARTH, J. **What is Artificial Intelligence?** 1997. Http://www-formal.stanford.edu/jmc/whatisai/whatisai.html.

MINAEE, S. et al. **Going Deeper Into Face Detection: A Survey**. 2021.

PAN, J.; Z, H.; Z, Y. M. S. Deblurring face images with exemplars. In: **European conference on computer vision**. [S.l.: s.n.], 2014. p. 47–62.

PAN, S. J.; YANG, Q. A survey on transfer learning. **IEEE Transactions on Knowledge and Data Engineering**, v. 22, p. 1345–1359, 2010. Available from Internet: <https://api.semanticscholar.org/CorpusID:740063>.

PARKHI, O. M.; VEDALDI, A.; ZISSERMAN, A. Deep face recognition. In: **British Machine Vision Conference**. [S.l.: s.n.], 2015.

PASZKE, A. et al. **PyTorch: An Imperative Style, High-Performance Deep Learning Library**. 2019.

REN J. YANG, S. D. D. W. W.; CAO, X.; TONG, X. Face video deblurring using 3d facial priors. In: **IEEE**. [S.l.: s.n.], 2019. p. 9388–9397.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. **Neural Networks**, Elsevier BV, v. 61, p. 85–117, jan. 2015. ISSN 0893-6080. Available from Internet: <http://dx.doi.org/10.1016/j.neunet.2014.09.003>.

SHEN W.-S. LAI, T. X. Z.; KAUTZ, J.; YANG, M.-H. Deep semantic face deblurring. In: **CVPR**. [S.l.: s.n.], 2018. p. 8260–8269.

SUIN, M.; PUROHIT, K.; RAJAGOPALAN, A. Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In: **IEEE**. [S.l.: s.n.], 2020. p. 3606–3615.

SUN, W. C. J.; XU, Z.; PONCE, J. Learning a convolutional neural network for non-uniform motion blur removal. In: **IEEE**. [S.l.: s.n.], 2015. p. 769–777.

SWE, T. T. Detection of faces from images using haar cascade classifier. In: **Iconic Research And Engineering Journals**. [S.l.: s.n.], 2020. p. 174–178.

USAMA, M. et al. Unsupervised machine learning for networking: Techniques, applications and research challenges. **IEEE Access**, v. 7, p. 65579–65615, 2019.

WALDO, J. **A comparative study of back propagation and its alternatives on multilayer perceptrons**. 2022.

WANG, B.; XU, F.; ZHENG, Q. **A survey on facial image deblurring**. 2023. Https://arxiv.org/abs/2302.05017. 2023-09-01.

WANG, X. et al. **A Survey of Face Recognition**. 2022. Https://arxiv.org/abs/2212.13038.

WOLF, T. H. L.; MAOZ, I. Face recognition in unconstrained videos with matched background similarity. In: **IEEE**. [S.l.: s.n.], 2011.

XIA, Z.; CHAKRABARTI, A. **Training Image Estimators without Image Ground-Truth**. 2019.

XUE, Z. **Blind Image Deblurring: a Review**. 2022.

YASARLA, R.; PERAZZI, F.; PATEL, V. M. Deblurring face images using uncertainty guided multi-stream semantic networks. In: **IEEE**. [S.l.: s.n.], 2020. p. 6251–6263.

YOON, S.; SHAHAB, C. **The clustered aggregation (cag) technique leveraging spatial and temporal correlations in wireless sensor networks**. 2007.

ZENG, T.; DIAO, C. Single image motion deblurring based on modified densenet. In: **2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)**. [S.l.: s.n.], 2020. p. 521–525.

ZHANG, K. et al. **Deep Image Deblurring: A Survey**. 2022. Https://arxiv.org/abs/2201.10700. 2022-01-26.