Universidade Federal do ABC
Centro de Matemática, Computação & Cognição
Bacharelado em Ciência da Computação

Machine Learning aplicado ao tratamento de imagens

Erick Funier dos Santos
Paulo Ricardo Cunha

Santo André - SP, Abril de 2023

Erick Funier dos Santos
Paulo Ricardo Cunha

Machine Learning aplicado ao tratamento de imagens

Trabalho de Graduação em Computação apresentado como parte dos requisitos necessários para a obtenção do Título de Bacharel em Ciência da Computação.

Universidade Federal do ABC – UFABC

Centro de Matemática, Computação & Cognição

Bacharelado em Ciência da Computação

Orientador: Mario Alexandre Gazziro

Santo André - SP

Abril de 2023

Agradecimentos

Gostaríamos de expressar nossa sincera gratidão ao Professor Mario Alexandre Gazziro, nosso orientador de TCC, e a todos os docentes que nos apoiaram durante esta jornada acadêmica. Seus conselhos, orientações e conhecimentos foram inestimáveis na realização deste trabalho, e estamos profundamente gratos pela dedicação e paciência que demonstraram ao longo do processo. Sentimo-nos verdadeiramente afortunados por ter tido a oportunidade de aprender com vocês, e levamos conosco o conhecimento e as habilidades adquiridas para o resto das nossas vidas. Novamente, obrigado por todo o apoio e orientação prestados neste trabalho de conclusão de curso.

Resumo

Atualmente já são utilizados fundo *Chroma Key* em diversos ambientes de estúdio com iluminação profissional o que facilita a remoção do fundo via software, porém alguns pontos devem ser considerados. O custo desses estúdios pode se tornar alto, pois muitos cobram por horas de utilização. O projeto atual trata da utilização do modelo de redes neurais convolucionais para a remoção de sombras geradas pela iluminação não uniforme em fundos de *Chroma Key* de imagens de pessoas, com a ideia de facilitar o tratamento dessas fotos em pós produção, e assim, podendo inserir a pessoa em qualquer cenário, permitindo uso de iluminação ambiente convencional, o que diminui consideravelmente o custo de produção. Os resultados obtidos através das métricas (F1-Score e Jaccard Score) correspondem com o resultado visual, as médias ficaram acima de 0.992. De acordo com os objetivos apresentados no presente trabalho, o projeto foi realizado com sucesso.

Palavras-chaves: Aprendizado de Máquina, Redes Neurais Convolucionais, Remoção de Sombra, *Chroma Key*.

Abstract

Currently a Chroma Key background is already used in several studio environments with professional lighting which facilitates the removal of the background via software, but some aspects must be considered like the cost of these studios can be higher, as soon as many charge for hours of use . This paper propose the use of convolutional neural networks to remove shadows generated by non-uniform ambient light in Chroma Key backgrounds of images of people, with the idea of facilitating the treatment of these photos in post production, and thus, being able to insert the person in any scenario allowing the use of conventional ambient lighting which presents a considerably lower cost. The results obtained through the metrics (F1-Score and Jaccard Score) correspond with the visual result the averages were above 0.992. According to the objectives presented in this paper we reach completly success.

Keywords: Machine Learning, Convolutional Neural Network, Shadow Removal, *Chroma Key*

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AUC	Area Under the Curve
ASPP	Atrous Spatial Pyramid Pooling
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CRF	Conditional Random Field
DCNN	Deep Convolutional Neural Network
GPU	Graphics Processing Unit
LED	Light Emitting Diode
ReLU	Rectified Linear Unit
ResNet	Residual Network
ROC	Receiver Operating Characteristic
SE	Squeeze and Excite

Sumário

1 Introdução

Aprendizado de máquina é um campo dedicado a escrever programas que resolvem problemas sem codificar explicitamente a solução. Nos últimos anos, modelos baseados em redes neurais têm atraído atenção em vários problemas de aprendizado de máquina. O projeto atual trata da utilização do modelo de redes neurais convolucionais que, nos últimos anos, superaram o estado da arte em muitas tarefas de reconhecimento visual. Embora as redes convolucionais já existam há muito tempo, seu sucesso foi limitado devido ao tamanho dos conjuntos de treinamento disponíveis, ao tamanho das redes consideradas e, consequentemente, hardwares capazes de executar os códigos de treinamento da rede neural.

O avanço das redes neurais se deu com o trabalho de Krizhevsky realizando o treinamento supervisionado de uma grande rede com 8 camadas e milhões de parâmetros no conjunto de dados ImageNet com 1 milhão de imagens de treinamento (RONNEBERGER PHILIPP FISCHER, 2015). Desde então, redes ainda maiores e mais profundas foram treinadas. O uso típico de redes convolucionais é em tarefas de classificação, onde a saída para uma imagem é um rótulo de classe única. No entanto, em muitas tarefas visuais, especialmente no processamento de imagens, a saída desejada deve incluir localização, ou seja, um rótulo de classe deve ser atribuído a cada pixel.

No projeto, abordamos a remoção de sombras geradas pela iluminação não uniforme em fundos de *Chroma Key* de imagens de pessoas, com a ideia de facilitar o tratamento dessas fotos em pós produção, e assim, podendo inserir a pessoa em qualquer cenário.

2 Motivação e Objetivos

2.1 Motivação

Atualmente já são utilizados fundo *Chroma Key* em diversos ambientes de estúdio com iluminação profissional o que facilita a remoção do fundo via software, porém alguns pontos devem ser considerados. O custo desses estúdios pode se tornar alto, pois muitos cobram pelas horas de utilização, conforme Tabela 1.

Tabela 1 – Comparação de Custos

Custo Estúdio - (Investimento diário)			
Região	Modo	Valor	Fonte
São Paulo - SP	Diária (8h)	R$650,00	(ALUGUELDESALASP,)
Palhoça - SC	Diária (10h)	R$1.000,00	(LIVETVBRASIL,)
São Paulo - SP	Diária (8h)	R$990,00	(3TELAS,)
São Paulo - SP	Diária (8h)	R$640,00	(ESTUDIOI9,)
	Média	R$820,00	

Custo Individual (Investimento único)			
Item		Valor	
Fundo Chroma		R$250,00	
Câmera		R$600,00	
	Total	R$850,00	

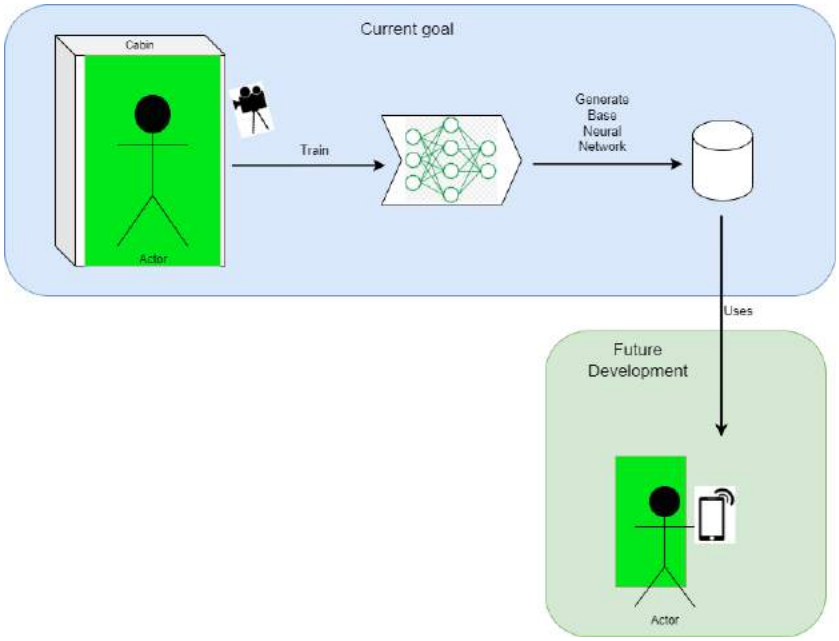
Fonte: Autoria Própria

Com a motivação de desenvolver um sistema portátil capaz de gravar vídeos utilizando um *smartphone* e o vídeo resultante conter apenas a pessoa que gravou com um fundo em *Chroma Key* sem sombras, para posterior utilização, a ideia então é implementar parte desse sistema utilizando um ambiente de treinamento. [1] Assim, seguindo a diretriz do Projeto de Graduação em Ciência da Computação, serão aplicados os conhecimento adquiridos durante o curso contribuindo para o aprofundamento do aprendizado.

2.2 Objetivo Geral

O campo de aprendizado de máquina fez grandes avanços nos últimos anos, especialmente para aplicativos de aprendizado profundo (SERMANET et al., 2013). Hoje, muitas atividades cotidianas são influenciadas por algoritmos de aprendizado, como: diagnósticos médicos (SPUHLER et al., 2020), seleção e recomendação de investimentos (KRAUS; FEUERRIEGEL, 2017), etc. Durante este período, é importante que as universidades contribuam para a formação de quadros nesta área. No campo técnico, o objetivo principal

Figura 1 – Aplicação completa



Fonte: Autoria Própria

desta proposta é, analisar o potencial de modelos de segmentação semântica de imagens. Para isso, propomos desenvolver uma extensão do modelo que, quando analisado, tem a capacidade de pegar uma imagem em material bruto e a partir desta utilizar somente o que deseja. O modelo proposto é comparado com outros dados modelos generativos em termos da qualidade das imagens produzidas e da capacidade de interpretação das representações obtidas.

2.3 Objetivos Específicos

São destacados os seguintes objetivos específicos:

- Montar um ambiente para treinamento da aplicação, incluindo cabine, iluminação e câmera.
- Gerar um *dataset* suficiente para treinamento da rede neural utilizando o ambiente montado.
- Propor uma adaptação de um modelo de rede neural para realizar a remoção de sombras em imagens com fundo de *Chroma Key*, deixando apenas a pessoa destacada.
- Analisar a acuidade do modelo proposto quando aplicado a imagens e vídeos gerados no ambiente montado.

3 Revisão Bibliográfica

3.1 Trabalhos Relacionados

Outros trabalhos já utilizaram redes neurais para remoção de *background*, com aplicações voltadas ao mercado de moda (LIANG, 2022), nesse caso, o objetivo é de comparar se a remoção de *background* poderia facilitar a análise computacional dos dados de moda de uma imagem, destacando apenas a pessoa e a roupa, apesar dos ganhos nos resultados, eventualmente a remoção do *background* causava perda de informação com diferentes tipos de roupas, pois o *background* não é controlado como o caso do *Chroma Key*. Com o objetivo de remover sombras de imagens essa aplicação (FAN; HAN; LI, 2019) utilizou redes convolucionais e um modelo com *encoder-decoder*, um bom resultado foi obtido considerando a forma utilizada, que é de classificar a sombra destacando-a através de filtros e então reconstruir a imagem.

Pode ser citada também uma aplicação (VARATHARASAN et al., 2019) que utilizou de redes neurais, inclusive comparando com o modelo DeepLabv3+, para classificar e detectar objetos em imagens com diversos *backgrounds*. Nesse caso, os melhores resultados foram obtidos utilizando o modelo DeepLab substituindo o *background* por *Chroma Key*, e por fim, realizando a identificação dos objetos em destaque.

3.2 Fundamentação Teórica

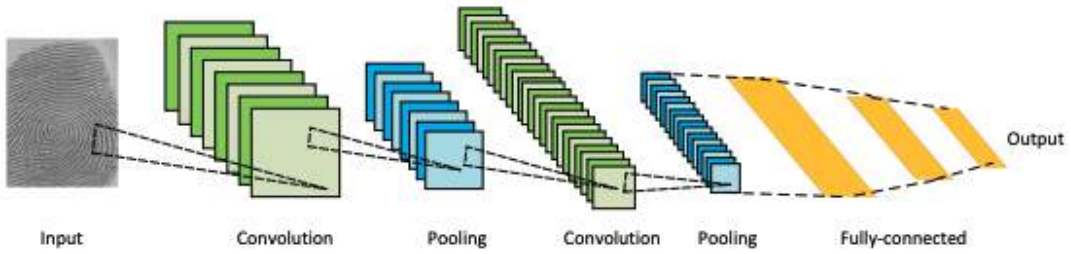
Em visão computacional (BARROW; TENENBAUM, 1981) um dos objetivos é conseguir extrair informações das imagens, isso pode ser realizado utilizando a segmentação da imagem (MINAEE et al., 2022), dessa forma a imagem pode ser dividida em diversos objetos. São utilizadas duas formas de segmentação, a de instância que classifica a imagem em um único rótulo e a semântica que faz a classificação a nível de pixel, identificando cada objeto baseado em um conjunto de *pixels* semelhantes, essa última segmentação é mais trabalhosa do que a primeira, se tratando da analise mais profunda da imagem.

Os modelos de segmentação de imagens, no geral, tem evoluído bastante, mas os modelos baseados em *Deep Learning* são os que mais obtiveram sucesso, eles utilizam a associação de diferentes modelos de redes neurais convolucionais, contendo *encoder* e *decoder* além de análise em multi escala. Um dos modelos apresentados no comparativo (MINAEE et al., 2022) que tiveram destaque utilizando o *dataset* PASCAL VOC-2012 (EVERINGHAM et al., 2010) foi o DeepLabV3+ (CHEN et al., 2018c) da família DeepLab (CHEN et al., 2018a) utilizando DCNN, *Atrous Convolution* e *Fully Connected CRF*.

3.2.1 *Convolutional Neural Networks* (CNN)

O modelo de rede neural CNN (FUKUSHIMA, 1980) é um dos mais utilizados para visão computacional, ele foi desenvolvido baseado em um modelo hierárquico de córtex receptivo visual proposto por Hubel e Wiesel (DH, 1962). As redes CNNs são compostas de três camadas:

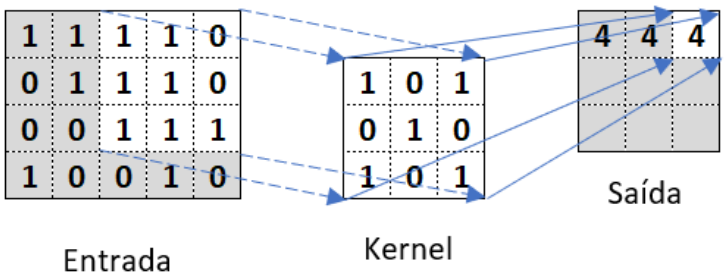
Figura 2 – *Arquitetura Convolutional Neural Network*



Fonte: (LECUN et al., 1998)

- Convolucional: Camada mais utilizada na rede, ela constrói os recursos informativos obtendo as informações úteis de cada imagem, isso é feito por meio de uma operação matemática de convolução, ou seja, são utilizadas duas matrizes uma matriz com os dados de entrada e uma matriz chamada de *kernel*, essa matriz percorre a matriz dos dados obtendo os dados úteis, a matriz resultante é um menor porém possui as informações mais importantes da primeira, essa matriz por sua vez pode ser utilizada na camada de ativação.(AGHDAM; HERAVI, 2017)

Figura 3 – Funcionamento da camada de convolução



Fonte: Autoria Própria

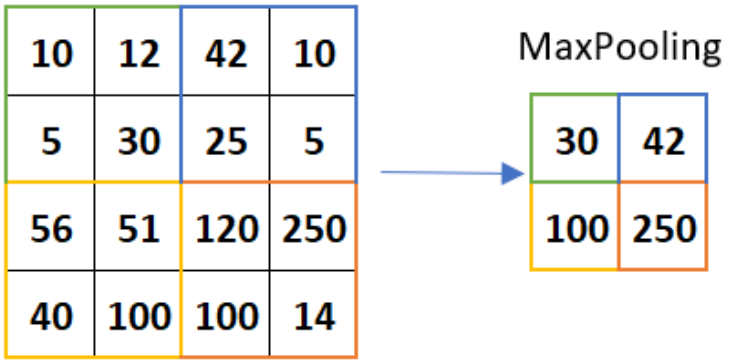
- Ativação: A camada de ativação provê a não-linearidade para a rede neural. A função de ativação mais utilizada é a ReLU (*Rectified Linear Unit*), a função ReLU pode ser representada por:

$$f(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0 \end{cases} \tag{3.1}$$

Com isso a saída da função ReLU será o máximo valor entre 0 e x.

- *Pooling*: Camada responsável por reduzir o espaço dimensional da entrada, contribui diretamente com a redução dos recursos computacionais necessários de processamento (KRIZHEVSKY; SUTSKEVER; HINTON, 2017).

Figura 4 – Funcionamento o *pooling* obtendo o máximo valor do quadrante



Fonte: Autoria Própria

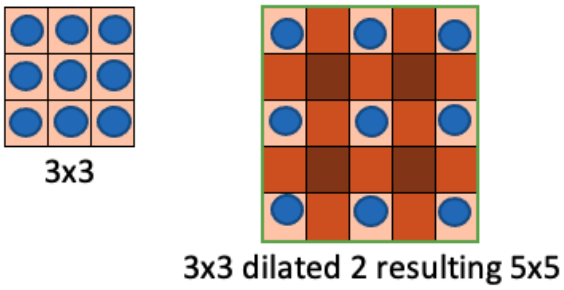
3.2.2 *Deep Convolutional Neural Networks* (DCNN)

O modelo DCNN segue a mesma base do CNN porém possui mais camadas internas, o que o torna profundo, ou seja, há mais camadas de convolução e mais camadas de ativação. A utilização de mais camadas reduz significativamente a taxa de erro desde que não seja atingido o chamado *"overfitting"* no qual a rede neural começa a gerar mais conexões do que o necessário, o que aumenta consideravelmente o tamanho da rede e diminui a acuidade. (HE et al., 2016)

3.2.3 *Atrous Convolution*

Os modelos da família DeepLab (CHEN et al., 2018a) utilizam dilatação nas camadas de convolução da CNN para obter melhor performance, essa dilatação faz com que um campo receptivo com 3 x 3 e dilatação 2 tenha o mesmo tamanho que um 5 x 5, porém utilizando apenas 9 parâmetros receptivos ao invés de 25 [5]. Ou seja, aumentando o campo receptivo sem adicionar nenhum custo computacional. Além disso, a utilização da dilatação proporciona o aumento no campo de visão dos filtros em qualquer camada da DCNN.

Figura 5 – Exemplo dilatação

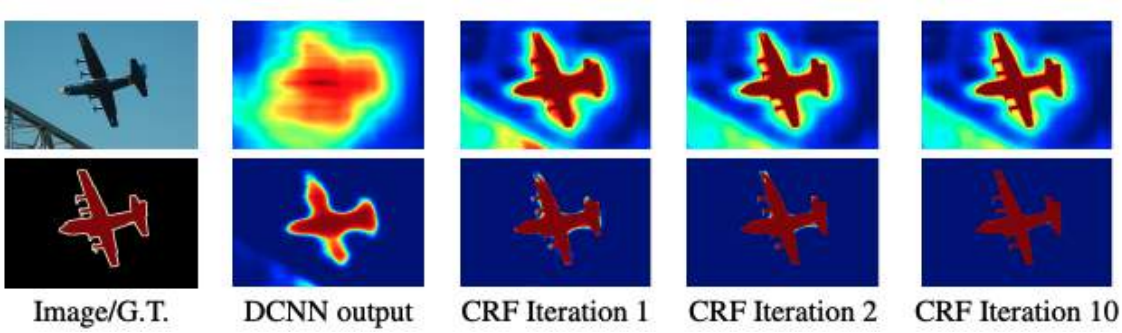


Fonte: Autoria Própria

3.2.4 *Fully Connected Conditional Random Field* (CRF)

Além das mudanças na DCNN o DeepLab (CHEN et al., 2018a) aplica camadas adicionais no fim da DCNN para obter maior nível de detalhamento [6], ela consegue remover os espúrios da falha na identificação de alguns *pixels* através de filtros Gaussianos. (KRäHENBüHL; KOLTUN, 2011)

Figura 6 – Iterações CRF na saída da DCNN



Fonte: (CHEN et al., 2018b)

3.2.5 DeepLabv3+

O DeepLabv3+ (CHEN et al., 2018c) é um modelo de segmentação semântica que utiliza uma variação de DCNN para segmentar imagens em regiões semânticas. Ele foi desenvolvido pela equipe do Google Research e é baseado na arquitetura DeepLabv3 (CHEN et al., 2018a), com algumas modificações significativas. O modelo DeepLabv3+ utiliza uma DCNN de múltiplas resoluções (*Multi-Resolution Atrous Convolutional or Atrous Spatial Pyramid Pooling* (ASPP)) para extrair características de várias escalas de uma imagem. Essas características são então combinadas e refinadas em um único mapa para segmentação.

A arquitetura do DeepLabv3+ é composta por várias camadas, incluindo a camada de entrada, a camada de pré-processamento, a camada de extração de características, a camada de decodificação e a camada de saída. A camada de entrada recebe a imagem de

entrada e a prepara para o processamento. A camada de pré-processamento é responsável por redimensionar a imagem de entrada para um tamanho adequado para a rede. A camada de extração de características é composta por várias camadas convolucionais e de *pooling*, que extraem características de várias escalas da imagem. A camada de decodificação utiliza informações de contexto de alta resolução para refinar a segmentação produzida pela camada de extração de características. Por fim, a camada de saída produz a segmentação final da imagem. (CHEN et al., 2018c)

Na entrada do modelo foi adotado uma rede neural como base, pois dessa forma é possível evitar que a rede fique saturada no início da implementação, portanto, utiliza-se uma rede neural de *backbone* já preparada para segmentação semântica, nesse caso foi utilizado o ResNet101 da família ResNet (HE et al., 2016) que possui 101 camadas.

Uma das principais modificações introduzidas no DeepLabv3+ em relação ao DeepLabv3 é um módulo de *decoder* para recuperar as informações de borda da imagem. Esse módulo utiliza a *atrous convolution* com taxas de dilatação crescentes para capturar informações de borda fina, o que ajuda a melhorar a precisão da segmentação em objetos pequenos e detalhes finos.

Outra modificação importante é o uso de um processo de fusão de características espaciais (*Spatial Feature Fusion*) para combinar características de diferentes resoluções. Esse processo utiliza também *atrous convolutions* com diferentes taxas de dilatação para extrair informações de diferentes escalas e, em seguida, funde essas informações em uma única saída. O DeepLabv3+ também utiliza um mecanismo de agrupamento de *pixels* sem perda de informação (*Global Pooling*) para capturar informações contextuais globais da imagem. Isso ajuda a melhorar a precisão da segmentação em objetos grandes e complexos. Por fim, o DeepLabv3+ (CHEN et al., 2018c) possui uma estrutura que pode ser dividida em duas partes, sendo a etapa de codificação e decodificação. A etapa de codificação pode ser resumida no modelo do DeepLabv3, e a decodificação que refina o resultado proveniente da etapa de codificação, aumentado a acuidade.

O modelo pode exigir um hardware poderoso para executar a segmentação em tempo real em grandes conjuntos de dados. Em resumo, é uma arquitetura altamente eficaz para tarefas de segmentação de imagem, que pode ser ainda aprimorada com técnicas avançadas e hardware poderoso.(KHAN et al., 2020)

3.2.6 *Squeeze and Excite*

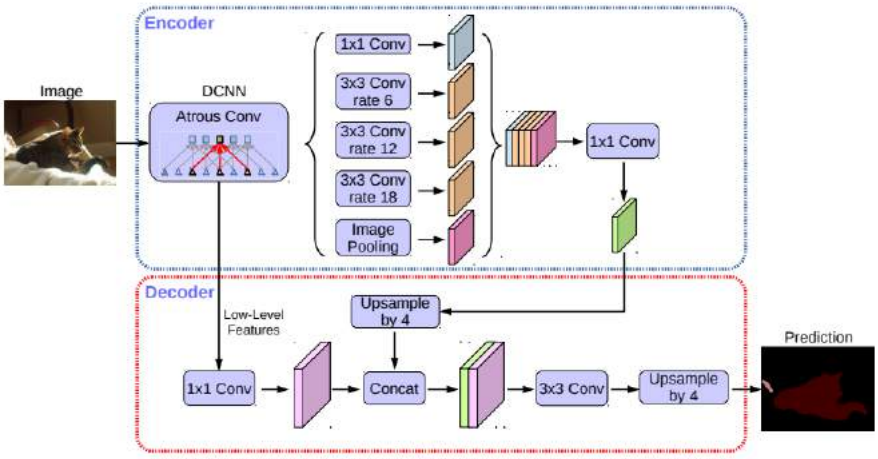
"Squeeze and Excite" é uma técnica de aprendizado profundo proposta em 2018 para aprimorar o desempenho de redes neurais convolucionais (CNNs). Essa técnica busca capturar e rebalancear as características mais importantes aprendidas pelos modelos, utilizando uma etapa de compressão dos recursos e outra de recalibração seletiva baseada

em canal. O rebalanceamento é realizado através de uma camada totalmente conectada, a qual pondera os recursos presentes em cada canal, levando em consideração a sua relevância. Dessa forma, a técnica amplifica as características mais importantes e suprime as menos importantes. O *Squeeze and Excite* pode ser aplicado em diversas tarefas de visão computacional, tais como classificação de imagem, detecção de objetos e segmentação semântica. (HU; SHEN; SUN, 2018)

3.2.7 Métricas

O pacote Scikit-Learn (PEDREGOSA et al., 2011) possui funções já implementadas para obter as métricas de análise, sendo Accuracy, F1 (F-Score), Jaccard, Recall e Precision das quais podemos destacar:

Figura 7 – Modelo DeepLabv3+



Fonte: (CHEN et al., 2018c)

3.2.7.1 F1 - F-score

O F-Score, também conhecido como medida F ou medida F1, é uma métrica que combina a precisão (*precision*) e o *recall* (revocação) de um modelo de classificação binária em um único valor. Essa métrica é muito utilizada em problemas de classificação binária desbalanceados, onde a proporção de exemplos positivos é significativamente menor do que a de exemplos negativos.

O F-Score é definido como a média harmônica entre a precisão e o *recall*. A precisão é a proporção de exemplos positivos classificados corretamente pelo modelo, enquanto o *recall* é a proporção de exemplos positivos reais que foram corretamente identificados pelo modelo.

O F-Score é calculado da seguinte forma:

F-Score = 2 * (precisão * *recall*) / (precisão + *recall*)

O valor do F-Score varia entre 0 e 1, sendo que 0 indica uma classificação completamente incorreta e 1 indica uma classificação completamente correta. Quanto maior o valor do F-Score, melhor é o desempenho do modelo na classificação dos exemplos positivos.

Uma das principais vantagens do F-Score é que ele considera tanto a precisão quanto o *recall* do modelo, o que permite avaliar a sua capacidade de classificar exemplos positivos de forma correta e completa. Além disso, o F-Score é uma métrica robusta e amplamente utilizada na avaliação de modelos de classificação binária, o que facilita a sua interpretação e comparação entre diferentes modelos.

No entanto, o F-Score também apresenta algumas limitações. Por exemplo, ele não considera os falsos positivos e falsos negativos de forma separada, o que pode ser importante em alguns contextos. Além disso, o valor do F-Score pode ser influenciado pelo desbalanceamento da base de dados, o que pode prejudicar a sua interpretação em alguns casos. Nesses casos, outras métricas, como a área sob a curva ROC (AUC-ROC), podem ser mais adequadas.

3.2.7.2 Jaccard

O índice de Jaccard, em síntese, é uma medida de similaridade que se baseia na proporção de elementos compartilhados entre dois conjuntos em relação ao número total de elementos únicos. Seu resultado é expresso em uma escala de 0 a 1, sendo que valores mais próximos de 1 indicam maior semelhança entre os conjuntos em questão.

Tomemos como exemplo dois conjuntos, A e B, compostos por elementos numéricos. Ao calcularmos o índice de Jaccard desses conjuntos, devemos primeiramente determinar quantos elementos estão presentes em ambos conjuntos e quantos estão presentes em pelo menos um deles. Uma vez estabelecido esse cômputo, podemos obter o resultado da fórmula $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$.

Importante frisar que o índice de Jaccard é sensível ao tamanho dos conjuntos. Isso significa que, em geral, conjuntos maiores tendem a apresentar índices de Jaccard menores, mesmo que compartilhem muitos elementos em comum. Para contornar essa limitação, outras medidas de similaridade, tais como a similaridade do cosseno, podem ser utilizadas.

É válido salientar que o índice de Jaccard mede a similaridade entre os conjuntos, não a dissimilaridade. Em outras palavras, quanto maior o índice de Jaccard, maior a similaridade entre os conjuntos. Para aferir a dissimilaridade, outras medidas devem ser utilizadas, tais como a distância de Hamming.

O índice de Jaccard possui inúmeras aplicações práticas em diversas áreas do conhecimento, como a análise de dados, a recuperação de informações, a mineração de dados, a bioinformática, entre outras. Por ser uma medida simples e eficaz, ele é amplamente utilizado para comparar a similaridade entre conjuntos de elementos.

4 Metodologia

O projeto atual visa treinar uma variação do modelo de rede neural DeepLabv3+ (CHEN et al., 2018c) para aprendizado aprofundado de imagem. Abaixo estão os passos mais importantes para realizar esta tarefa. Em primeiro lugar, é realizada uma pesquisa bibliográfica preliminar com o objetivo de aprofundar o conhecimento do aluno no campo da aprendizagem profunda, especialmente no que diz respeito aos modelos examinados neste projeto. Esta primeira etapa da pesquisa bibliográfica examina os fundamentos dos modelos de redes neurais e principalmente reforça os conceitos da teoria da informação. Ao longo da vida do projeto, o trabalho bibliográfico manterá o conteúdo atualizado com os avanços em modelos de *deep learning* e geração de dados. Ao mesmo tempo, os alunos estudam Keras(CHOLLET et al., 2015), Tensorflow(ABADI et al., 2015) e outras bibliotecas para implementar modelos de aprendizado em profundidade. Hoje, os artigos que tratam de *deep learning* são comumente acompanhados de implementações baseadas em bibliotecas de código aberto. O uso dessas bibliotecas acelera o desenvolvimento de novos modelos e ajuda a reproduzir os resultados publicados.

Os seguintes aspectos foram considerados na escolha desta biblioteca entre muitos outros relacionados a este tópico: tamanho da implementação (medido pelo número de modelos diferentes que a biblioteca implementa), documentação Disponibilidade e qualidade, suporte para implementação em GPUs, suporte da máquina de nível comunidade de aprendizagem, facilidade de aprendizagem (utilizando bibliotecas), facilidade de implementação de novos modelos. O objetivo desta fase é que os alunos desenvolvam implementações de modelos fluentes e ganhem experiência no treinamento de modelos de redes neurais multicamadas (modelos de aprendizado profundo). O treinamento desses modelos pode ser complicado devido ao grande número de hiper parâmetros que precisam ser ajustados.

O método será dividido da seguinte forma: Etapa de preparação e etapa de execução

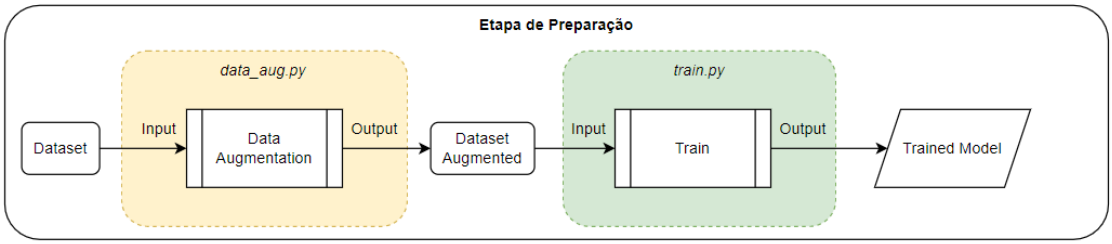
4.1 Etapa de preparação

Nessa etapa foi feito o processo para preparação das imagens para a rede neural, composto por: Aquisição dos dados, tratamento dos dados, implementação do modelo para da rede neural. [8]

4.1.1 Aquisição dos Dados

Para o treinamento da rede neural precisaremos obter uma grande quantidade de imagens que possuem sombra, dessa forma a rede identificará o padrão que queremos

Figura 8 – Etapa de Preparação



Fonte: Autoria Própria

remover da imagem em si. Para obter o melhor resultado, optaremos em gerar nosso próprio *dataset* no ambiente da cabine [9] que será montado, pois dessa forma a rede ficará otimizada para tal ambiente, que poderá ser reproduzido em estruturas semelhantes.

Figura 9 – Cabine de gravação

Fonte: Autoria Própria

4.1.1.1 Ambiente

Foi montada uma cabine para gravarmos vídeos no fundo de *Chroma Key* para obter as imagens com sombra e utilizar na rede. A cabine conta com iluminação interna sendo 8 barras de *led* para deixar a luz uniforme, com essa iluminação algumas sombras já começam a aparecer, além disso, utilizamos dois *spots* de luz que conseguem gerar uma diferença de intensidade na iluminação externa, consequentemente gerando sombras mais fortes em regiões específicas conforme o posicionamento dos *spots*.

Utilizamos um computador com uma *webcam* (LOGITECH, 2021) de lente grande angular [10] gravamos o cenário com uma pessoa se movimentando e capturamos as imagens dos vídeos gravados.

Figura 10 – Logitech C930e grande angular

Fonte: Autoria Própria

4.1.2 Tratamento dos Dados

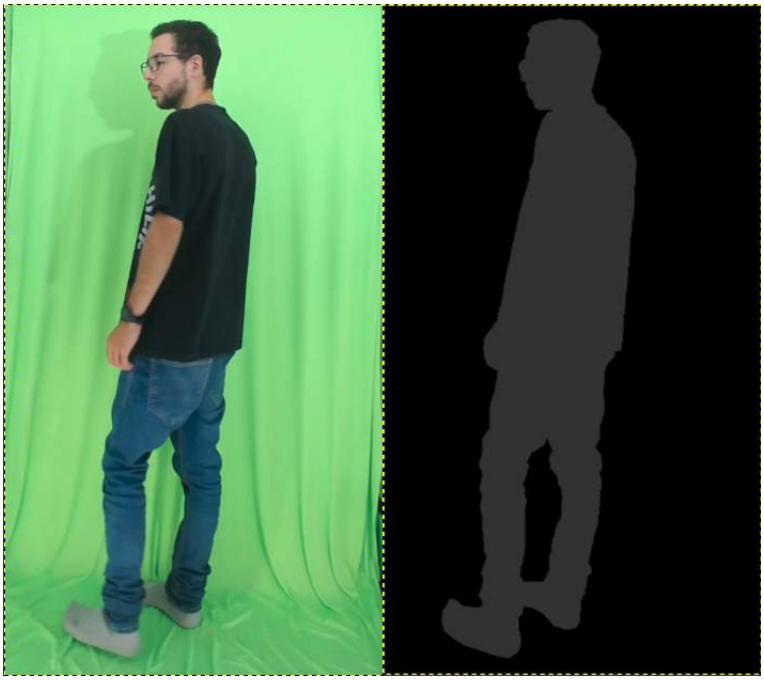
A partir dos vídeos foi possível obter cerca de 100 imagens específicas do cenário com a cabine, produzimos um código para aumentar o *dataset*, gerando 5 imagens com pequenas variações a partir da imagem de entrada[11], incrementando assim, o *dataset* em cerca de 500 imagens. Mediante o emprego dessas medidas, obteve-se significativa redução temporal, uma vez que a edição das imagens é um processo manual, em cada imagem é necessário produzir a máscara destacando a pessoa o fundo [12]. Tal máscara é a predição que a rede neural, uma vez treinada, conseguirá obter.

O código que realiza expansão do *dataset* (*data_aug.py*) realiza a separação das imagens de treino e de teste com a biblioteca *sklearn* (PEDREGOSA et al., 2011), as manipulações das imagens são feitas com as bibliotecas *cv2* (opencv-python versão em *python* do algoritmo opencv) (BRADSKI, 2000) e *albumentations* (BUSLAEV et al., 2020).

Figura 11 – Dataset aumentado via código

Fonte: Autoria Própria

Figura 12 – Na esquerda a imagem original, na direita a máscara, detalhe para as cores do fundo (000000) e da pessoa (010101)



Fonte: Autoria Própria

4.1.3 Implementação do Modelo

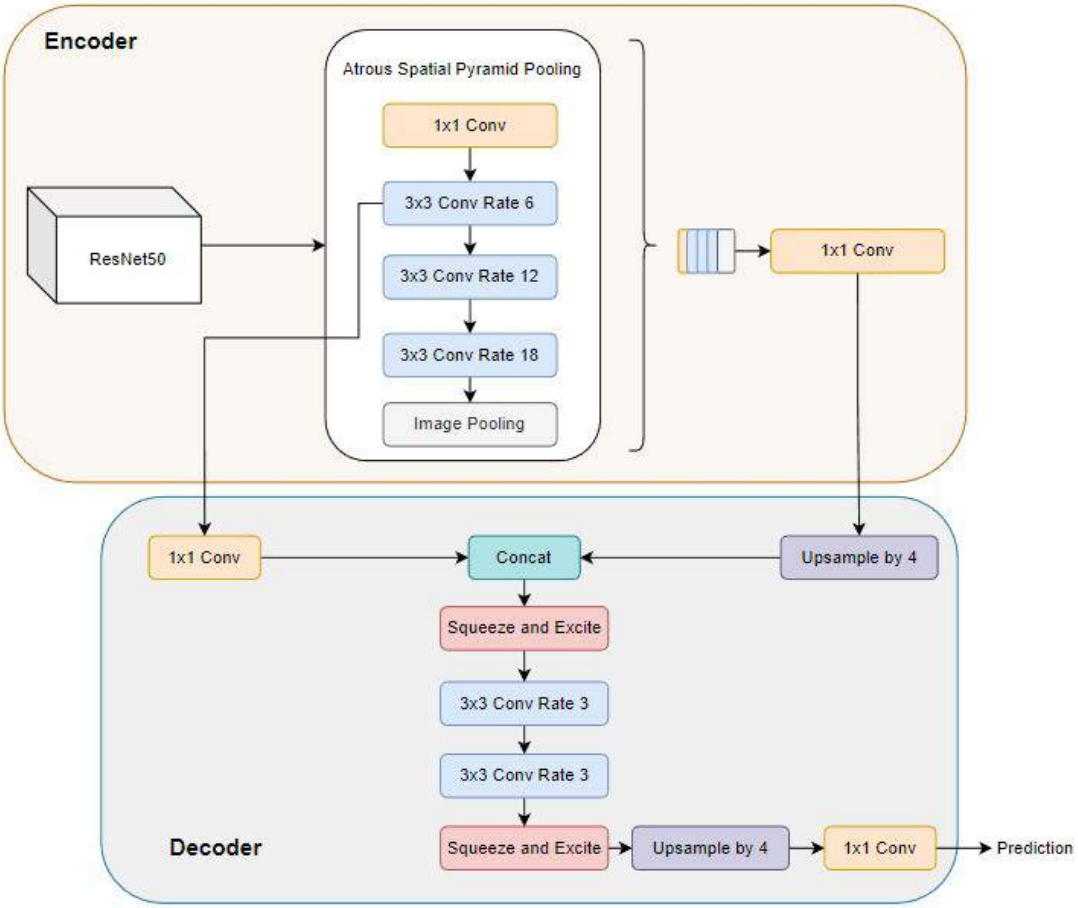
O código implementado é um modelo de segmentação semântica com DeepLabv3+ (CHEN et al., 2018c) usando o TensorFlow (ABADI et al., 2015). Ele é composto por funções que carregam e pré-processam dados de treinamento e validação, criam o modelo DeepLabv3+ (CHEN et al., 2018c), compilam o modelo e definem as métricas, definem as funções de *callback* para monitoramento de treinamento e salvamento do modelo, e treinam o modelo usando os dados de treinamento e validação fornecidos.

Seguindo o modelo [7], foram feitas algumas mudanças, utilizou-se o *encoder* pré produzido ResNet50 diferentemente do ResNet101 utilizado pela equipe desenvolvedora do DeepLabv3+ (CHEN et al., 2018c), ambos pertencem à mesma família ResNet (HE et al., 2016), porém o ResNet50 possui 50 camadas enquanto o ResNet101 possui 101 camadas resultando no dobro de *floating point operations per second* (FLOPS). O desempenho do ResNet50 se mostrou pouco inferior ao ResNet101 (TAHIR; IFTIKHAR; MUMRAIZ,

2021), e dois pontos foram considerados, quanto maior a densidade da rede menor é a classificação de elementos relevantes da imagem, considerando que esses estão em maior escala, no nosso caso a separação é basicamente da pessoa para um fundo quase uniforme, para fundos com muita variação, seria interessante utilizar rede mais densa, além disso, redes com maior densidade são mais complexas de serem processadas por terem mais FLOPS, a ResNet50 apresenta, praticamente, metade do tamanho da ResNet101. (KELEK; CALIK; YILDIRIM, 2019)

Após o ResNet50, seguimos para o módulo Atrous Spatial Pyramid Pooling (ASPP), seguindo a mesma quantidade de convoluções e *image pooling* do DeepLabv3+. Por fim, finalizando o *Encoder* com a última convolução 1x1. [13]

Figura 13 – *Encoder* e *Decoder*



Fonte: Autoria Própria

No *Decoder* foi feito uma mudança em relação ao DeepLabv3+, utilizando um recurso de *Squeeze and Excite* (SE) que se mostrou eficiente para otimização da rede (HU; SHEN; SUN, 2018). Seu objetivo é realizar a calibração dos recursos da rede, dando ênfase à aqueles com mais informações e menos ênfase à aqueles com menos informação relevante da rede. Isso é feito em duas etapas, a etapa de *Squeeze*, que realiza a criação de um descritor de canais através de um mapa de recursos, esse por sua vez carrega informações globais

da rede que são aproveitadas pelas camadas inferiores. Na etapa de *Excitation*, ocorre efetivamente a recalibração da rede, utilizando o mapa gerado anteriormente são obtidas ativações de amostras específicas baseado na dependência de canais e essas ativações rebalanceiam a rede neural. No fim, foi mantido o último *UpSample by* 4 e adicionado uma última Conv 1x1 na saída do *decoder*. [13]

4.1.4 Treinamento da Rede

A primeira parte do código é composta por importações de bibliotecas necessárias, incluindo o próprio TensorFlow, as funções de *callback* e a arquitetura do modelo. Em seguida, há duas funções de configuração de GPU/CPU. A função *set_gpu_option* é usada para configurar as GPUs disponíveis no sistema e limitar o uso de memória disponível, enquanto a função *set_cpu_option* é usada para configurar o uso da CPU no sistema.

O código também define algumas funções auxiliares para manipular dados, como *create_dir* para criar um diretório, *shuffling* para embaralhar dados, *load_data* para carregar os dados de treinamento e validação, *read_image* para ler uma imagem e normalizá-la, *read_mask* para ler uma máscara e *tf_parse* para pré-processar as imagens e máscaras usando as funções de leitura de imagem e máscara.

As funções *tf_dataset* e *tf_parse* são usadas para criar o conjunto de dados TensorFlow, que é usado para treinar o modelo. A função *tf_dataset* cria o conjunto de dados a partir dos dados de entrada e saída usando *tf_parse* para pré-processar os dados. A função *tf_parse* usa as funções *read_image* e *read_mask* para pré-processar as imagens e máscaras e, em seguida, normaliza os dados.

Por fim, a função *main* é responsável por definir os parâmetros de treinamento, criar o modelo e compilá-lo com a função de perda *dice_loss* e métricas como coeficiente Dice, IoU, recall e precisão. Em seguida, ele define as funções de *callback* para monitorar o treinamento e salvar o modelo periodicamente. Por fim, ele treina o modelo usando os dados de treinamento e validação fornecidos.

O *hardware* utilizado para treinamento da rede possui as seguintes configurações:

- Intel(R) Core(TM) i9-10980xe
- Nvidia RTX 3090 (CUDA 11.2.67)
- 128GB de Ram DDR4 4000mhz
- SSD NVMe 1tb

A versão de cada biblioteca utilizada está na Tabela 2:

Tabela 2 – Versões das bibliotecas

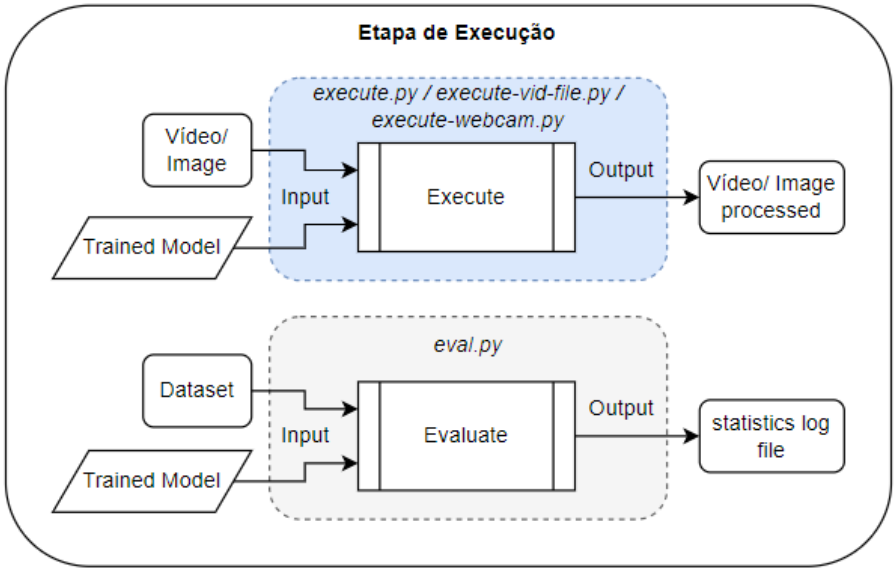
Biblioteca	Versão
numpy	1.23.3
opencv-python	4.6.0.66
tqdm	4.64.1
tensorflow	2.10.0
scikit-learn	1.1.2

Fonte: Autoria Própria

4.2 Etapa de execução

Nessa etapa [14] foi feito o processo de execução, ou seja, de utilização do modelo gerado na preparação, esse modelo pode ser utilizado nos códigos para processamento da imagem, seja uma imagem fixa, ou vídeo em mp4 ou, por fim, um vídeo em *stream* de uma *webcam*. Além disso, nessa etapa é feita a análise dos resultados com o modelo pronto, porém será abordado no tópico Análise de Resultados [5].

Figura 14 – Etapa de Execução



Fonte: Autoria Própria

4.2.1 Execução

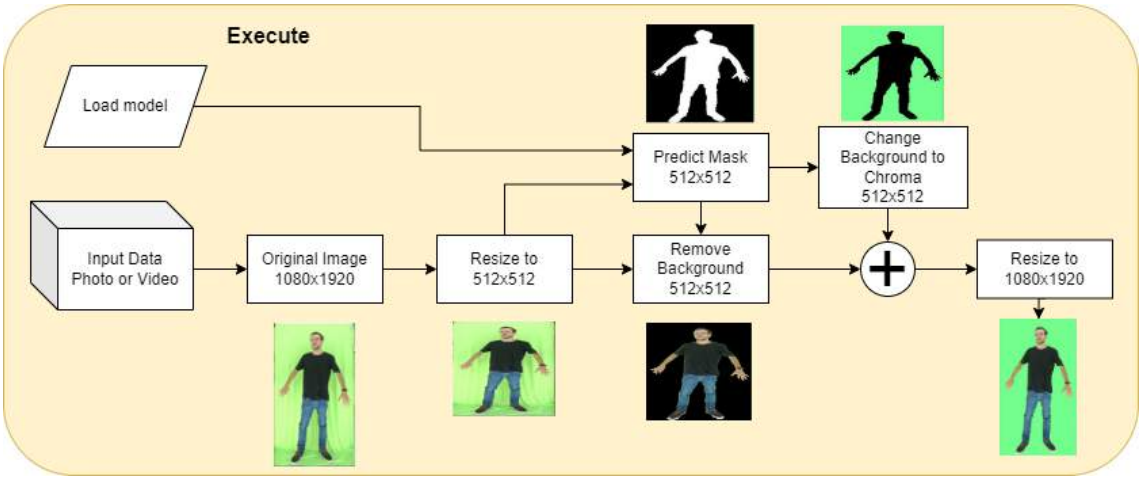
Para a execução, foram feitos três códigos específicos:

- *execute.py* - realiza o processamento de fotos provenientes de um repositório.
- *execute-vid-file.py* - realiza o processamento de um vídeo no formato mp4, gerando o vídeo resultante no mesmo formato.

- *execute-webcam.py* - realiza o processamento de um vídeo em tempo real, proveniente de uma *webcam* conectada ao PC.

O processo de execução é semelhante em ambos os códigos. Primeiro o modelo treinado é carregado, em seguida é feita a entrada dos dados, seja de fotos ou vídeo, Para o caso da foto, um *loop* percorre todas as fotos presentes no diretório gerando as predições e em seguida montando as fotos resultantes. Nos vídeos o processo é feito *frame* a *frame*, predição e reconstrução do *frame*. [15]

Figura 15 – Etapa de Execução

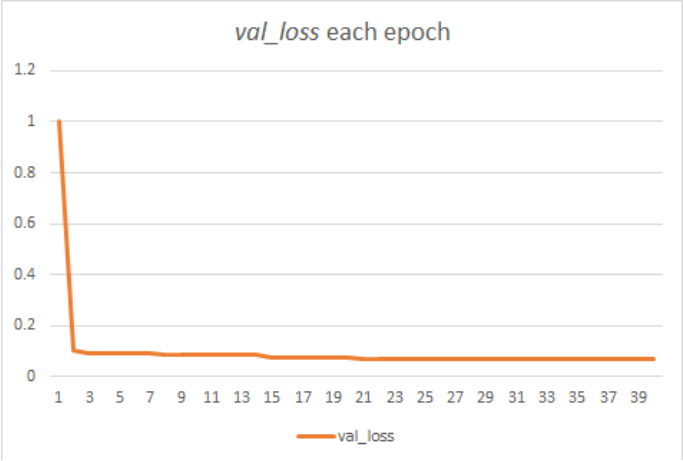


Fonte: Autoria Própria

5 Análise de Resultados

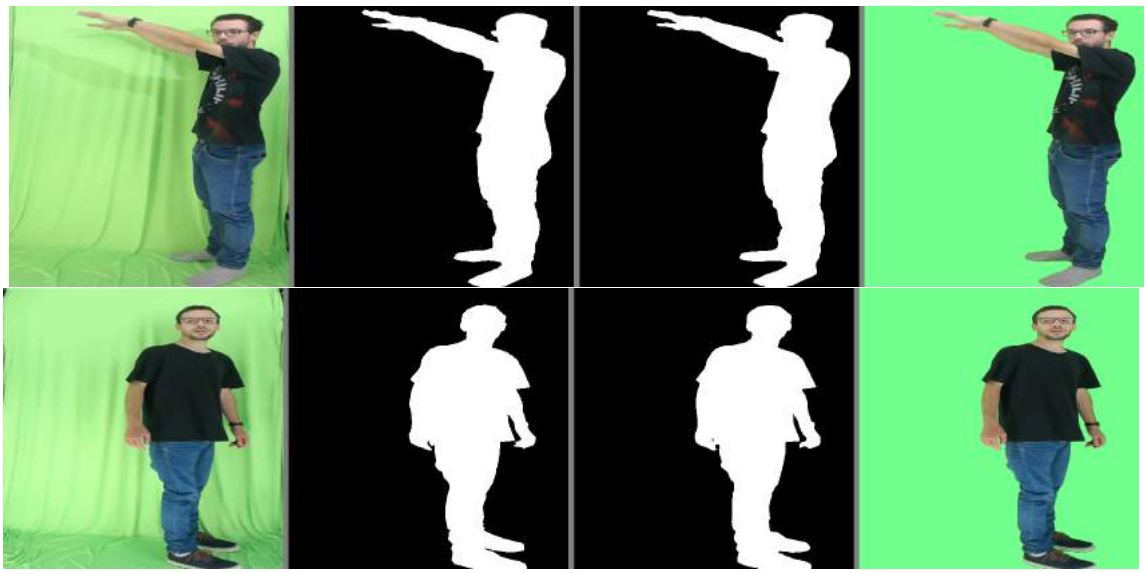
Para o treinamento da rede, verificamos que com o *dataset* atual a rede parava de evoluir após a *Epoch* 40, conforme a Figura [16]. O tempo total gasto para treinamento da rede foi de cerca de 15 minutos sendo 22 segundos por *Epoch* com o *dataset* aumentado para 582 imagens.

Figura 16 – *val_loss* vs *Epochs*



Fonte: Autoria Própria

Figura 17 – Resultados obtidos com a aplicação, da esquerda para a direita, imagem original, predição, predição refinada, e imagem final



Fonte: Autoria Própria

Visualmente o desempenho da aplicação se mostrou bem eficiente, não são notadas regiões que sobrepõe o fundo gerado pela imagem da pessoa. Não há qualquer alteração de

cor e contraste que modifique a imagem de entrada, portanto, uma vez que a pessoa esteja com a iluminação adequada as sombras que aparecerem no fundo serão removidas. 17

No que diz respeito ao vídeo, o resultado visual permaneceu bastante eficiente, não há perdas de *frames* no vídeo de entrada para o vídeo de saída, nem qualquer modificação de cor e contraste. Para o caso do *stream* de vídeo da *webcam*, foi observado o *delay* de processamento da aplicação, que leva em torno de 40 ms, mas, uma vez com o *stream* em andamento, não há perda de *frames*.

Utilizando as métricas de análise foi possível obter os seguintes resultados com algumas imagens do *dataset*, conforme Tabela [3] e Figura [18].

Figura 18 – F1 vs Jaccard

Fonte: Autoria Própria

Tabela 3 – Resultados com imagens do *dataset*

Image	F1	Jaccard
1	0.996620178	0.993263126
2	0.997276306	0.994567409
3	0.996635437	0.993293439
4	0.995609283	0.991256955
5	0.996067047	0.992164909
6	0.994762421	0.989579420
7	0.996971130	0.993960553
8	0.995956421	0.991945411
9	0.996902466	0.993824062
10	0.996505737	0.993035809
Média	0.996330643	0.992689109

Fonte: Autoria Própria

Os resultados obtidos através das métricas correspondem com o resultado visual, as médias ficaram acima de 0.991. Pode-se destacar o F1 (F-Score) e o índice de Jaccard para

avaliar a precisão de modelos de segmentação de imagem porque levam em consideração tanto as regiões positivas quanto as negativas e medem diretamente a sobreposição entre as máscaras segmentadas e as máscaras verdadeiras, fornecendo uma medida direta da precisão da segmentação. Pela Figura [18] é possível observar valores próximos entre o F1 e o Jaccard, porém pelo fato do Jaccard considerar duas áreas da imagem no cálculo, sendo a área de interseção e a área de união, enquanto o F1 considera apenas a área de interseção e a soma da quantidade de *pixel* de cada área separadamente, o F1 acaba tendo resultados mais próximos a 1, pois o Jaccard se torna mais preciso ao utilizar a união das imagens pois o cálculo em relação à quantidade de *pixels* é abordado na imagem montada, dessa forma, mais eficiente. Além disso, por utilizar o Recall, o F1 score acaba sendo influenciado negativamente pelo desbalanceamento do *dataset*, no caso o F1 indica que o *dataset* está calibrado, não há pouca informação e nem muita, o suficiente para o resultado obtido.

6 Trabalhos Futuros

Com o ambiente e modelo, já é possível continuar as implementações de projetos futuros até obter um aplicação mobile capaz de realizar a remoção do fundo com um cenário simples de *Chroma Key* sem iluminação profissional. Em termos de otimização, obter um *dataset* mais amplo no cenário da cabine irá proporcionar maior flexibilidade para a rede, sendo capaz de ter excelentes resultados com várias pessoas, porém o processo para criar as máscaras do *dataset* ainda é bem demorado.

Uma das recomendações para trabalhos futuros será a realização da análise de *overfitting* dos treinamentos, monitorando o ponto exato de divergência entre as curvas de treinamento e aprendizado único.

O modelo implementado foi feito exclusivamente para a execução em um Computador com placa gráfica, no caso do desenvolvimento Mobile, haverá a necessidade de portabilidade para o ambiente do *Smartphone*, dependendo do sistema operacional, como as bibliotecas utilizadas são todas de código aberto, isso se torna mais fácil.

7 Conclusão

Conforme os objetivos definidos no trabalho em questão, este foi concluído com êxito. O ambiente para treinamento foi montado e utilizado, ficando disponível para trabalhos futuros, com esse ambiente foi gerado um *dataset* suficiente para o treinamento do modelo proposto baseado no DeepLabv3+. Os resultados obtidos foram condizentes com as expectativas estabelecidas, tendo sido removidos não somente as sombras, todo o *background Chroma Key* gerando como resultado uma imagem ou vídeo com a figura humana sobre um fundo *Chroma* inteiramente uniforme.

A utilização de ferramentas para aumentar o *dataset* contribuiu muito para a evolução do trabalho, pois quando maior e mais variado for, dentro do cenário da cabine proposto, melhor será o treinamento da rede neural e consequentemente terá resultados ainda melhores.

Referente ao modelo proposto, apesar desse ser o primeiro trabalho da motivação inicial, tal modelo poderá ser facilmente reutilizado considerando o rápido treinamento da rede e a rápida predição do resultado. Isso por que as modificações realizadas são voltadas para melhoria de desempenho e aplicação ao cenário do projeto.

Referências

3TELAS. *3telas*. Disponível em: <https://www.3telas.com.br/>. Citado na página 2.

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. Citado 2 vezes nas páginas 11 e 14.

AGHDAM, H. H.; HERAVI, E. J. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2017. ISBN 331957549X. Citado na página 5.

ALUGUELDESALASP. *alugueldesalasp*. Disponível em: <https://www.alugueldesalasp.com.br/>. Citado na página 2.

BARROW, H.; TENENBAUM, J. Computational vision. *Proceedings of the IEEE*, v. 69, n. 5, p. 572–595, 1981. Citado na página 4.

BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. Citado na página 13.

BUSLAEV, A. et al. Albumentations: Fast and flexible image augmentations. *Information*, v. 11, n. 2, 2020. ISSN 2078-2489. Disponível em: <https://www.mdpi.com/2078-2489/11/2/125>. Citado na página 13.

CHEN, L.-C. et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 40, n. 4, p. 834–848, 2018. Citado 3 vezes nas páginas 4, 6 e 7.

CHEN, L.-C. et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 40, n. 4, p. 834–848, 2018. Citado na página 7.

CHEN, L.-C. et al. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: FERRARI, V. et al. (Ed.). *Computer Vision – ECCV 2018*. Cham: Springer International Publishing, 2018. p. 833–851. ISBN 978-3-030-01234-2. Citado 6 vezes nas páginas 4, 7, 8, 9, 11 e 14.

CHOLLET, F. et al. *Keras*. GitHub, 2015. Disponível em: <https://github.com/fchollet/keras>. Citado na página 11.

DH, W. T. H. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol.*, v. 160, n. 1, p. 106–154, 1962. Citado na página 5.

ESTUDIOI9. *estudioi9*. Disponível em: <https://locacao.estudioi9.com.br/>. Citado na página 2.

EVERINGHAM, M. et al. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, v. 88, n. 2, p. 303–338, jun. 2010. Citado na página 4.

FAN, H.; HAN, M.; LI, J. Image shadow removal using end-to-end deep convolutional neural networks. *Applied Sciences*, 2019. Citado na página 4.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernetics*, v. 36, n. 4, p. 193–202, 1980. Citado na página 5.

HE, K. et al. Deep residual learning for image recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 770–778. Citado 3 vezes nas páginas 6, 8 e 14.

HU, J.; SHEN, L.; SUN, G. Squeeze-and-excitation networks. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2018. p. 7132–7141. Citado 2 vezes nas páginas 9 e 15.

KELEK, M. O.; CALIK, N.; YILDIRIM, T. Painter classification over the novel art painting data set via the latest deep neural networks. *Procedia Computer Science*, v. 154, p. 369–376, 2019. ISSN 1877-0509. Proceedings of the 9th International Conference of Information and Communication Technology [ICICT-2019] Nanning, Guangxi, China January 11-13, 2019. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050919308221>. Citado na página 15.

KHAN, Z. et al. Evaluation of deep neural networks for semantic segmentation of prostate in t2w mri. *Sensors*, v. 20, n. 11, 2020. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/20/11/3183>. Citado na página 8.

KRäHENBüHL, P.; KOLTUN, V. Efficient inference in fully connected crfs with gaussian edge potentials. In: SHAWE-TAYLOR, J. et al. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2011. v. 24. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2011/file/beda24c1e1b46055dff2c39c98fd6fc1-Paper.pdf>. Citado na página 7.

KRAUS, M.; FEUERRIEGEL, S. Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems*, Elsevier BV, v. 104, p. 38–48, dec 2017. Disponível em: <https://doi.org/10.1016/j.dss.2017.10.001>. Citado na página 2.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 60, n. 6, p. 84–90, may 2017. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/3065386>. Citado na página 6.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998. Citado na página 5.

LIANG, J. *The research of background removal applied to fashion data: The necessity analysis of background removal for fashion data*. Tese (Doutorado), 2022. Disponível em: <http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-320601>. Citado na página 4.

LIVETVBRASIL. *livetvbrasil*. Disponível em: <https://livetvbrasil.com.br/site/aluguel-estudio/>. Citado na página 2.

LOGITECH. *Logitech c930e*. [S.l.], 2021. Disponível em: <https://www.logitech.com/content/dam/logitech/pt_br/video-collaboration/pdf/c930e-datasheet.pdf>. Citado na página 13.

MINAEE, S. et al. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 44, n. 7, p. 3523–3542, 2022. Citado na página 4.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado 2 vezes nas páginas 9 e 13.

RONNEBERGER PHILIPP FISCHER, T. B. O. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, p. 234–241, 2015. Citado na página 1.

SERMANET, P. et al. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. Citado na página 2.

SPUHLER, K. et al. Full-count PET recovery from low-count image using a dilated convolutional neural network. *Medical Physics*, Wiley, v. 47, n. 10, p. 4928–4938, aug 2020. Disponível em: <https://doi.org/10.1002/mp.14402>. Citado na página 2.

TAHIR, H.; IFTIKHAR, A.; MUMRAIZ, M. Forecasting covid-19 via registration slips of patients using resnet-101 and performance analysis and comparison of prediction for covid-19 using faster r-cnn, mask r-cnn, and resnet-50. In: *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*. [S.l.: s.n.], 2021. p. 1–6. Citado na página 15.

VARATHARASAN, V. et al. Improving learning effectiveness for object detection and classification in cluttered backgrounds. In: *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*. IEEE, 2019. Disponível em: <https://doi.org/10.1109/reduas47371.2019.8999695>. Citado na página 4.