UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

FRANCISCO LUCAS SOUSA NOBRE

DETECÇÃO E REMOÇÃO AUTOMÁTICA DE OBJETOS INDESEJADOS EM IMAGENS UTILIZANDO APRENDIZAGEM PROFUNDA

QUIXADÁ

2021

FRANCISCO LUCAS SOUSA NOBRE

DETECÇÃO E REMOÇÃO AUTOMÁTICA DE OBJETOS INDESEJADOS EM IMAGENS UTILIZANDO APRENDIZAGEM PROFUNDA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia De Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia De Computação.

Orientador: Prof. Dr. Cristiano Bacelar de Oliveira

QUIXADÁ

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

Nobre, Francisco Lucas Sousa.
Detecção e remoção automática de objetos indesejados em imagens utilizando aprendizagem profunda / Francisco Lucas Sousa Nobre. – 2021.
57 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Computação, Quixadá, 2021.
Orientação: Prof. Dr. Cristiano Bacelar de Oliveira.

1. Objetos-Detecção. 2. Aprendizagem profunda. 3. Visão computacional. 4. Imagens. 5. Segmentação. I. Título.

CDD 621.39

FRANCISCO LUCAS SOUSA NOBRE

DETECÇÃO E REMOÇÃO AUTOMÁTICA DE OBJETOS INDESEJADOS EM IMAGENS UTILIZANDO APRENDIZAGEM PROFUNDA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia De Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia De Computação.

Aprovada em: ___/___/___

BANCA EXAMINADORA

Prof. Dr. Cristiano Bacelar de Oliveira (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Paulo Armando Cavalcante Aguilar
Universidade Federal do Ceará (UFC)

Prof. Dr. João Vilnei de Oliveira Filho
Universidade Federal do Ceará (UFC)

À minha família, por sempre acreditar em mim e me apoiar, por sempre me darem a força e a motivação necessárias. Aos meus amigos que nunca deixaram eu me sentir cabisbaixo, que nesses dias tortuosos sempre me encorajaram e foram meus apoiadores.

“O mundo sempre parece mais brilhante quando
você acaba de fazer algo que não existia antes”
(Neil Gaiman)

RESUMO

Os avanços tecnológicos vêm crescendo ao longo das décadas e a tecnologia está se tornando cada vez mais presente em nosso cotidiano. Com isso os meios de gerar e adquirir informação estão cada vez mais diversos e de mais acessíveis, gerando uma quantidade massiva de dados.Isto torna possível a extração de informações a partir de diversas técnicas, para os mais diversos fins. Tendo isso em vista, este trabalho apresenta uma aplicação que utiliza técnicas de Visão Computacional juntamente com técnicas de Aprendizagem Profunda *(Deep Learning)* como meio para proteger a privacidade em umas desses tipos de dado: as imagens. Realizando a remoção de objetos indesejados, sem que haja um impacto significativo para o observador. Esta ferramenta pode ser utilizada para o lazer, na correção de imagens e remoção de objetos, sendo ainda uma opção alternativa a ferramentas já existentes como no *PHOTOSHOP*®. O sistema mostrou que, comparado as ferramentas presentes no *PHOTOSHOP*®, consegue obter resultados bons, para remover objetos em imagens, tendo a mínima participação do usuário, exigindo um nível técnico bem menor e gerando o resultado em tempo bastante reduzido.

Palavras-chave: Objetos-Detecção. Aprendizagem profunda. Visão computacional. Imagens. Segmentação

ABSTRACT

Technological advances have been growing over the decades and technology is becoming more and more present in our daily lives. With this, the ways of generating and acquiring information are increasingly diverse and more accessible, generating a massive amount of data. This massive amount of data makes it possible to extract information from various techniques, for the most diverse purposes. With this in mind, this work presents an application that uses Computer Vision techniques together with Deep Learning techniques as a way to protect the privacy of one of these types of data: images. Performing the removal of unwanted objects, without having a significant impact on the observer. Suck tool that can be used for leisure, in the correction of images and removal of objects, being also an alternative alternative to tools already existing as in *PHOTOSHOP*®. The system showed that, compared to the tools present in *PHOTOSHOP*®, it manages to obtain good results to remove objects in images, with minimal user participation, requiring a much lower technical level and generating the result in a very short time.

Keywords: Objects-Detection. Deep learning. Computer vision. Images. Segmentation

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SÍMBOLOS

CNN	*Convolutional Neural Network*
FCN	*Fully Convolutional Network*
ANN	*Artificial Neural Network*
COCO	*Common Objects in COntext*
YOLO	*You Only Look Once*
YOLACT	*You Only Look At CoefficienTs*
$R-CNN$	*Region Based Convolutional Neural Networks*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, uma grande parte da população mundial tem contado com a *Internet* e outras tecnologias que buscam facilitar e auxiliar a vida de milhões de usuários. Uma parcela dessas pessoas utiliza a tecnologia para registrar e compartilhar em redes sociais momentos como viagens, festas, formaturas, confraternização e etc, em imagens e vídeos, facilmente obtidas através de câmeras fotográficas e aparelhos celulares, gerando, assim, uma quantidade significativa de informações sobre os envolvidos.

Com tantas informações disponíveis é possível extrair dados que podem ser utilizados de inúmeras formas. Muitas vezes os dados capturados contêm não apenas informações do proprietário da mídia, mas, também, informações dos transeuntes presentes, involuntariamente, durante a captura das imagens, ou, ainda, símbolos ou marcas comerciais cuja divulgação pode não ser autorizada. A eventual divulgação de imagem de alguém capturada sem conhecimento prévio pode ser tida como um tipo de invasão de privacidade, como consta na Lei Geral de Proteção de Dados Pessoais (LGPDP) nº 13.709, de 14 de agosto de 2018 (BRASIL, 2018). Portanto, a captura de dados deve ser tratada com cuidado, sobretudo nos dias atuais, em que um dos objetos de maior valor comercial é a informação, principalmente quando falamos de dados como imagens ou vídeos, que podem ser alvos algoritmos de Aprendizagem de Máquina (*Machine Learning*) aplicados sobre os dados para extrair informações que podem ser usadas para diversos fins.

Tendo isso em vista, este trabalho apresenta uma aplicação que usa métodos de Visão Computacional em conjunto de técnicas de Aprendizagem Profunda (*Deep Learning*) para tentar proteger a privacidade de forma automática, removendo informações sensíveis contidas em imagens, sem que haja um impacto significativo para o observador. Dentre outros benefícios, isto pode diminuir o efeito nocivo caso ocorram vazamentos de dados ou algum acesso não autorizado. Assim, este trabalho buscou desenvolver um modelo computacional capaz de remover objetos com dano mínimos ao restante do conteúdo da imagem, que possa ser usado para diversos fins, inclusive para a proteção de identidade.

Este trabalho está organizado da seguinte forma: o Capítulo 2 traz os conceitos teóricos que fundamentam o trabalho; o Capítulo 3 apresenta os trabalhos relacionados; o Capítulo 4 apresentará os passos estabelecidos para a execução deste trabalho; o Capítulo 5 apresenta os resultados obtidos; o Capítulo 6 conclui o trabalho, resumindo os resultados e apresentando possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados os principais conceitos para o entendimento das tecnologias utilizadas na concepção da solução proposta. A seção inclui os seguintes temas: Aprendizagem Profunda, Visão Computacional, Detecção de Objetos e Remoção e Reconstrução, com destaque para as técnicas de detecção de objetos, fundamental para o conceito da solução proposta.

2.1 Visão Computacional

A Visão Computacional, como Gonzalez e Woods (2009) mencionam, busca utilizar computadores de forma a emular características da visão humana através de técnicas de aprendizagem, para, assim, extrair informações das imagens e, por consequência, fazer inferências com base nessas informações e tomar ações. Ainda segundo Gonzalez e Woods (2009), a fronteira entre as áreas de Visão Computacional e Processamento de Imagens não é clara, inclusive sendo uma questão que gera divergência de opiniões entre autores das áreas. Assim, ele propõe uma forma de alinhar as ideias para podermos definir de forma aproximada as duas áreas, sugerindo que Processamento de Imagens refere-se à quando a entrada do sistema é uma imagem e a saída é outra imagem, enquanto que em Visão Computacional o sistema recebe uma imagem e tem como saída uma interpretação dessa imagem. Assim, a área de Processamento de Imagem trabalha com as informações presentes em um imagem, e a Visão Computacional busca dar sentido a estas informações.

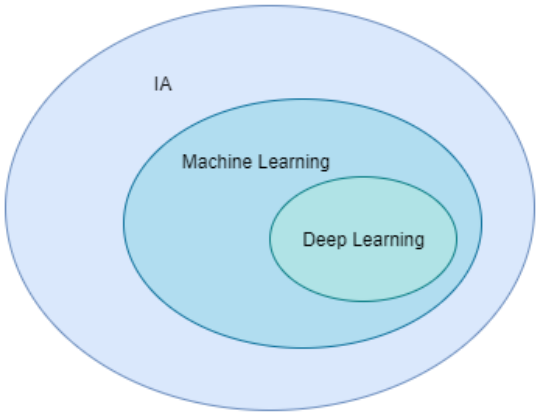
Uma tarefa da Visão Computacional é a detecção de objetos. Esta tarefa visa localizar e separar objetos de interesse contidos na imagem. O estado da arte da detecção de objetos atual é o *You Only Look Once - YOLO* (REDMON; FARHADI, 2018), que utiliza técnicas de Aprendizagem Máquina para localizar objetos. Mais detalhes serão apresentados na Seção 2.3.4

2.2 Inteligencia Artificial e Aprendizagem profunda

A Inteligência Artificial (IA) nasceu em meados da década de 1950, quando começaram a questionar a possibilidade de fazer um computador pensar. Segundo Rosebrock (2017), a Inteligência Artificial tem como objetivo realizar tarefas que são de natureza simples para humanos, mas desafiadoras para um computador fazer de forma automática. IA é uma área que engloba campos como Aprendizagem Máquina e Aprendizagem Profunda, conforme mostra a

Figura 1.

Figura 1 – Representação de subcampos entre Inteligencia Artificial, *Machine Learning* e *Deep Learning*.



Fonte: Adaptado de Chollet (2017)

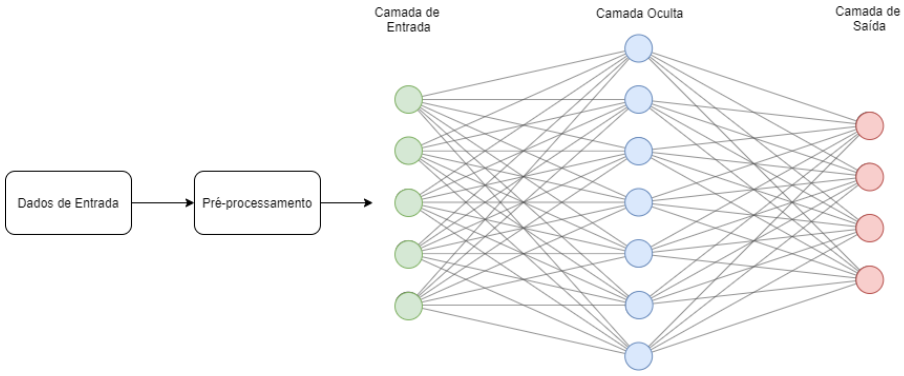
Aprendizagem de Máquina (*Machine Learning, em inglês*), segundo Geron (2017), é a arte de programar computadores para que eles possam aprender a partir de dados. Em Aprendizagem Máquina, o aprendizado pode ser feito de diversas formas, como o aprendizado supervisionado, para qual o aprendizado se dá a partir de algoritmos que são treinados com dados previamente categorizados e contém *labels*, vide algoritmos como *k-Nearest Neighbors* (LAAKSONEN; OJA, vol.3, 1996), *Support Vector Machines* (HEARST *et al.*, 1998), *Artificial Neural Network* (CHOLLET, 2017), e etc. Outra forma seria o aprendizado não supervisionado, em que as técnicas não necessitam desses *labels*, como clusterização (OYELADE *et al.*, 2019), visualização (GERON, 2017) e redução de dimensionalidade(GERON, 2017) e etc. Existe também o aprendizado semi-supervisionado, em que parte dos dados não contém *labels* e uma pequena porção contém; ainda assim, os algoritmos conseguem lidar com a presença desses dados não categorizados, a exemplo do *Deep Belief Networks (DBNs)* (YUMING HUA *et al.*, 2015). Dentro da Aprendizagem Máquina temos o subcampo do Aprendizagem Profunda que vem em constante desenvolvimento ao longo dos últimos anos.

Segundo Chollet (2017), a Aprendizagem Profunda é uma nova abordagem sobre como representar a forma de aprendizado. Modelos de Aprendizagem Profunda fazem parte da família das *Artificial Neural Networks (ANNs)*. Segundo Rosebrock (2017), *ANNs* são algoritmos de Aprendizagem Máquina com inspiração na estrutura de funcionamento do cérebro humano, que irão aprender a partir de dados e se especializar em padrões. Ainda em (CHOLLET, 2017), o autor comenta que o termo "Profunda" de Aprendizagem Profunda não se dá pelo

aprofundamento de alguma compreensão mais alta alcançada pela abordagem, e sim pela forma como os modelos funcionam estruturalmente, em cima da ideia de camadas sucessivas de representações. Portanto, o "Profunda" se refere à presença destas várias camadas.

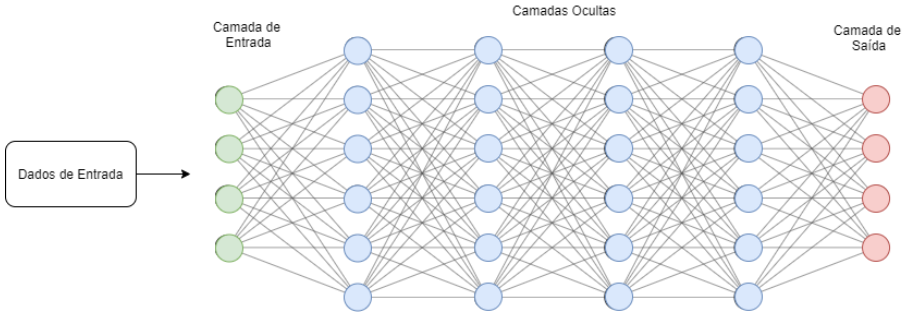
Segundo Chollet (2017), os modelos de Aprendizagem Profunda podem ter dezenas ou mesmo centenas de camadas sucessivas, enquanto outras abordagens possuem apenas 1 ou 2 camadas como vista na Figura 2, que são chamadas de aprendizado superficial por apresentarem baixa profundidade no modelo. A quantidade de camadas que contribuem para um modelo de Aprendizagem Profunda é o que denota a profundidade do modelo, como vemos na Figura 3, em que temos 4 camadas ocultas. Podemos citar exemplos de ANNs como *Convolutional Neural Network (CNN)*(LECUN *et al.*, 1998) e *Single Shot Detector* (SSD)(LIU *et al.*, 2016), amplamente utilizadas para trabalhar com imagens, muitas vezes utilizadas para fazer reconhecimento e classificação de objetos.

Figura 2 – Representação da camada de uma rede neural para Aprendizagem Máquina.



Fonte: Elaborado pelo autor.

Figura 3 – Representação da estrutura das camadas de Aprendizagem Profunda.



Fonte: Elaborado pelo autor.

2.3 Detecção de Objetos

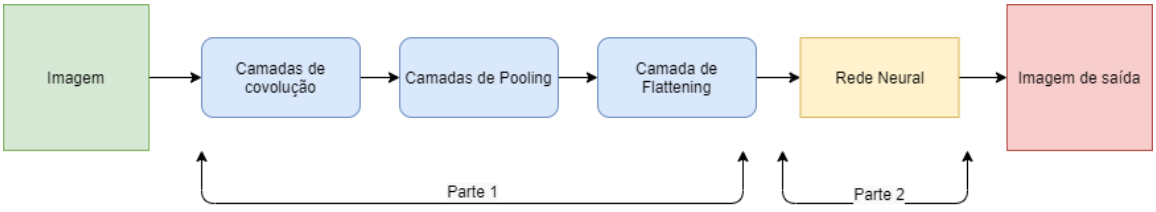
Técnicas de detecção de objetos, juntamente com a classificação de objetos, compõem algumas das tarefas mais comuns em Visão Computacional. A classificação de objeto é a tarefa de identificar as classes dos objetos que estão presentes na imagem. Já a detecção de objetos, é a tarefa de identificar objetos e localizá-los na imagem. Nesta seção, comentamos algumas técnicas para realizar a tarefa de detecção de objetos, como *Convolutional Neural Network (CNN)*, *Region-based Convolutional Neural Networks (R-CNN)*, *Single shot Detector (SSD)* e *You Only Look Once (YOLO)*.

2.3.1 Convolutional Neural Network (CNN)

As CNNs são redes neurais profundas usadas normalmente para a classificação de objetos em imagens. Contudo, elas não se limitam apenas a imagens, podendo, também atuar em áreas como Reconhecimento de voz ou Processamento de Linguagem Natural. O primeiro caso de sucesso de uma CNN foi apresentado por LECUN *et al.* (1998), com um modelo de 7 camadas divididas para convolução e uma rede neural *Full-Connected*. Segundo Ballard (2018), as CNNs são mais precisas e eficazes que as redes neurais clássicas.

A construção de uma CNN pode ser dividida em duas partes, uma parte que pré processa os dados da imagem e faz a convolução, a segunda é uma rede neural *Full-Connected*, como vemos na Figura 4. A primeira é parte é dada por camadas de convolução, camadas de *Pooling* e *Flattening* respectivamente. A rede neural que está na parte 2 da Figura 4 é uma *full-connected*, isso quer dizer que cada neurônio de uma camada está ligado a todos os neurônios da próxima camada. No final temos a saída da rede neural que irá classificar os objetos que se encontram na imagem.

Figura 4 – Representação de uma CNN.

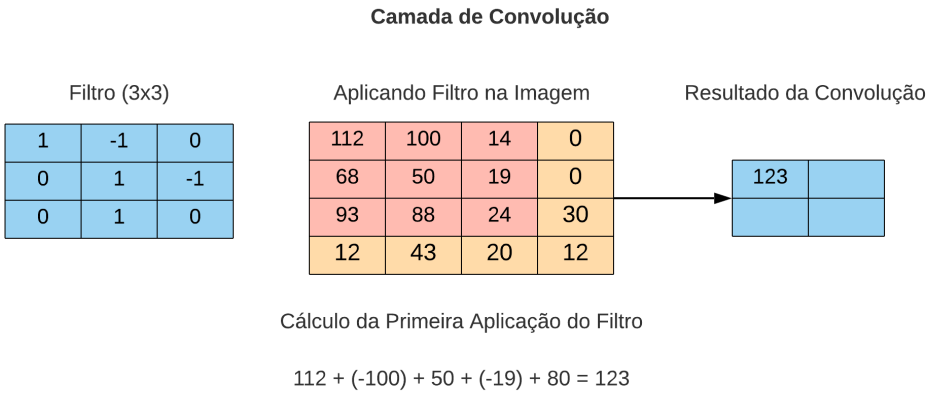


Fonte: Adaptado de Chollet (2017)

Segundo Geron (2017), a camada de convolução é o principal componente na criação de uma CNN. As camadas convolucionais têm como principal objetivo extrair características

da imagem de entrada. A convolução aplica filtros com valores aleatórios que passam por toda a imagem, em seguida, é selecionado o filtro com melhor desempenho para poder extrair características da imagem; essa seleção é feita internamente pela camada de convolução. Na Figura 5 temos uma visualização de como a camada funciona. Na camada temos um *Kernel* que seria nosso filtro, que irá percorrer a imagem realizando operações de multiplicação e soma, e como resultado temos o que chamamos de mapa de características. O mapa de características é representação das principais características da imagem, que foram extraídas pelo filtro.

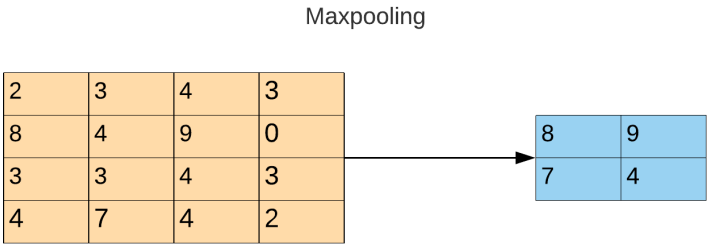
Figura 5 – Representação da camada de convolução, utilizando filtro 3x3



Fonte: Adaptado de Chollet (2017)

A camada de *pooling*, segundo Geron (2017), tem um conceito simples de reduzir a carga computacional. Na Figura 6 temos a representação de uma camada de *pooling*, utilizando a técnica de *MaxPooling*, a técnica consiste em pegar o maior numero de quadrante delimitado. Sewak *et al.* (2018) falam que para reduzir a carga, o *pooling* reduz a amostra de entradas da imagem, reduzindo a dimensionalidade, pois tornando o número de entradas menor existirá menos conteúdo para processar.

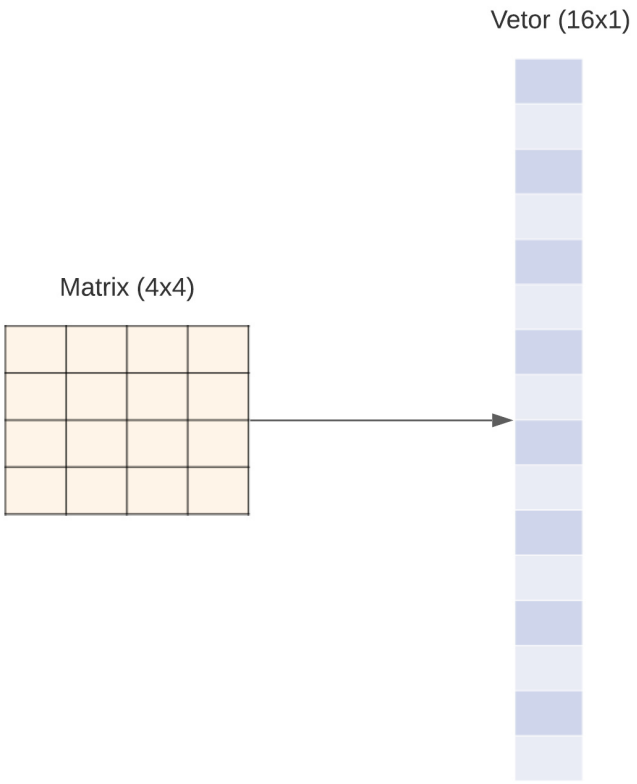
Figura 6 – Representação da camada de *pooling*, utilizando *Maxpooling* 2x2



Fonte: Adaptado de Chollet (2017)

A camada de *flattening* é uma camada dedicada a transformar a saída da camada de *pooling* de forma que ela se ajuste a entrada da rede neural, transformando uma imagem que possui um formato de matriz para o formato de vetor como mostra a Figura 7.

Figura 7 – Representação da camada de *flattening*



Fonte: Adaptado de Chollet (2017)

2.3.2 Region-based Convolutional Neural Networks (R-CNN)

Segundo Girshick *et al.* (2014), R-CNN é um sistema de detecção de objetos que combina técnicas de seleção de regiões com CNN. Ele veio com o propósito de ser uma alternativa para as CNNs normais, que são muito caras e lentas computacionalmente quando aplicadas à busca por objetos na imagem. Devido à varredura que acontece em busca dos melhores filtros há um grande número de operações que devem ser realizadas. R-CNN resolve esse problema usando uma proposta chamada *selective search*, que reduz a quantidade de caixas delimitantes dentro do conteúdo da imagem que irá alimentar o classificador em forma de CNN, diminuindo assim o número de operações que serão feitas. A *selective search* usa características da imagem como cor, intensidade, textura e etc, para fazer a localização das regiões das caixas. O classificador

baseado em CNN irá usar as caixas para fazer a classificação dos objetos na imagem e como saída da R-CNN temos os objetos selecionados e classificados.

2.3.3 Single shot Detector (SSD)

O SSD implementa, de forma equilibrada, precisão e velocidade na tarefa de detecção de objetos. Segundo Liu *et al.* (2016), SSD é uma técnica de detecção de objetos que usa uma única CNN pré-treinada para extrair os recursos da imagem. Em seguida, é aplicado um filtro sobre os recursos para prever caixas delimitantes na imagem e uma probabilidade de classificação de cada objeto dentro das caixas e quanto maior for a probabilidade de um objeto está dentro da região, mais a região irá se ajustar ao objeto. A CNN faz diversas combinações de previsões com diferentes resoluções para ter a capacidade de tratar objetos de tamanhos variados. A retirada de diversas etapas, como a geração de múltiplas propostas de filtros e reamostragem de pixels etc, torna mais simples e fácil o processo de treinamento, como Liu *et al.* (2016) comentam.

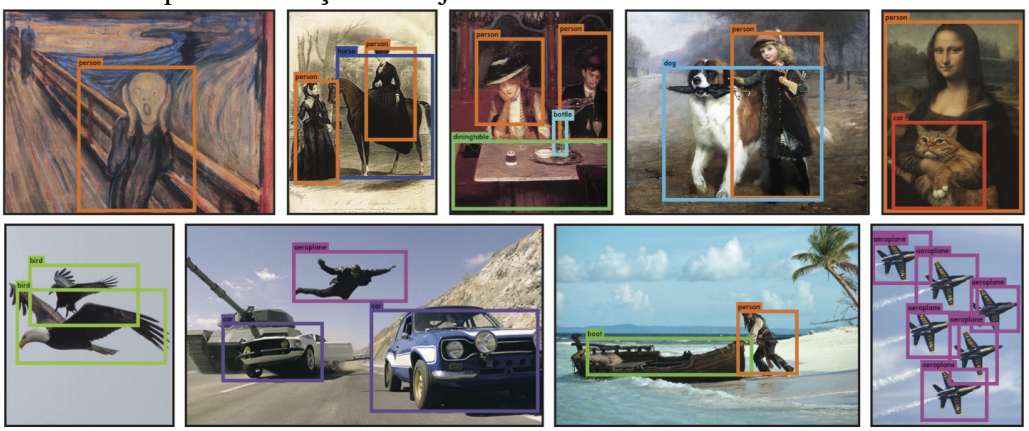
2.3.4 You Only Look Once (YOLO)

O *You Only Look Once (YOLO)* (REDMON; FARHADI, 2018) é um sistema de detecção de objetos em tempo real de última geração, sendo considerado o estado da arte da sua respectiva área. *YOLO* consegue ser extremamente rápido e possuir uma alta precisão. Enquanto os antigos métodos de detecção de objeto precisam rodar o classificador centenas ou milhares de vezes, o *YOLO* se sobressai por ser um método de detecção de passada única, que utiliza uma CNN como um extrator de características. De acordo com Redmon e Farhadi (2018), o *YOLO* aplica uma única rede neural para toda a imagem, em que a rede neural divide a imagem em regiões que são delimitadas por caixas e para cada caixa é dada uma probabilidade de conter um objeto; as áreas com maiores probabilidades são consideradas as áreas em que há objetos de interesse.

Segundo Redmon e Farhadi (2018), o *YOLO* se utiliza de uma rede neural profunda chamada *darknet*, que compartilha o mesmo nome do *framework* que é utilizado para implementar o detector. Esse *framework* foi criado pelos próprios Redmon e Farhadi (2018), sendo um *framework* de código aberto escrito em C, que tem suporte de GPU. A Figura 8 mostra exemplos de detecção utilizando *YOLO*.

Com todas essas características, o *YOLO* se mostra uma ótima técnica para problemas que têm como requisito uma solução com alto desempenho, devido aos seus elevados resultados

Figura 8 – Exemplo de detecção de objetos com YOLO.



Fonte: Retirada de Redmon e Farhadi (2018)

em quesitos de velocidade, precisão e ainda contar com a possibilidade de explorar sua arquitetura pelo fato de ser código aberto.

2.4 Segmentação de Objetos

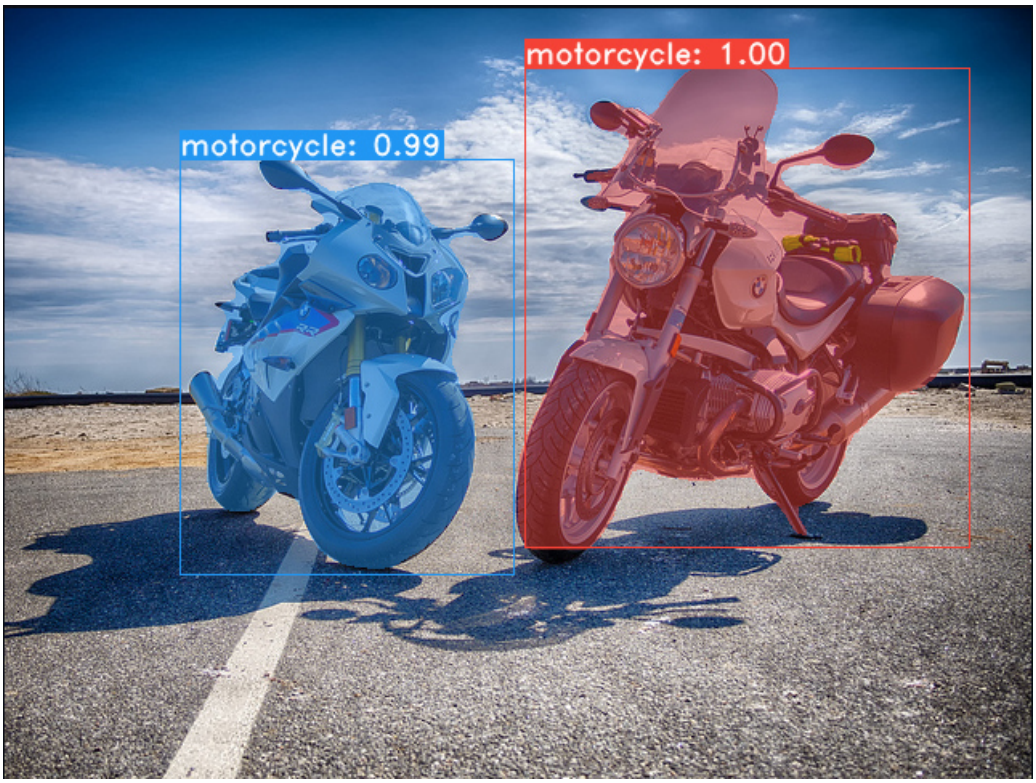
A tarefa de segmentação de objetos é um passo maior que a detecção de objetos, sendo que se torna uma tarefa ainda mais complexa, por buscar máscaras e não apenas detectar *Bounding Boxes*, que são retângulos que delimitam o objetos através de coordenadas. Nesta seção será apresentado o algoritmo *YOLACT - You Only Look At CoefficienTs* Bolya *et al.* (2019) que foi utilizado neste trabalho, como o segmentador de objetos.

2.4.1 YOLACT - You Only Look At CoefficienTs

O *YOLACT* de Bolya *et al.* (2019) é um segmentador de estágio único e alta performance, pensado para atuar no ambiente de tempo real. Sua arquitetura é uma extensão da arquitetura utilizada para detecção de objetos. O autor apresenta um modelo totalmente convolucional simples para a tarefa de segmentação de objetos em tempo real. Isso é possível por causa da realização em paralelo de sub-tarefas para gerar mascaras de protótipos, que podem ser pensadas como instâncias de localização espacial e a realização da previsão de coeficientes de mascarás, que são informações para cada ancora que codificar as instâncias.

A rede irar começar a aprender a localizar mascaras para posições em que instâncias visuais, espacial e semânticas semelhantes começarem a aparecer no protótipo. Na Figura 9 podemos verificar uma imagem com objetos segmentados.

Figura 9 – Exemplo de Segmentação de objetos *YOLACT*



Fonte: Retirada de Bolya *et al.* (2019)

2.5 Técnicas de Remoção e Reconstrução

A tarefa de reconstrução de imagens é a dada pelo processo de restaurar uma imagem danificada, a partir de elementos encontrados no conteúdo da imagem original. Nesta seção, serão apresentados o *Seam Carving* e o *Inpainting*, que são algumas técnicas de reconstrução.

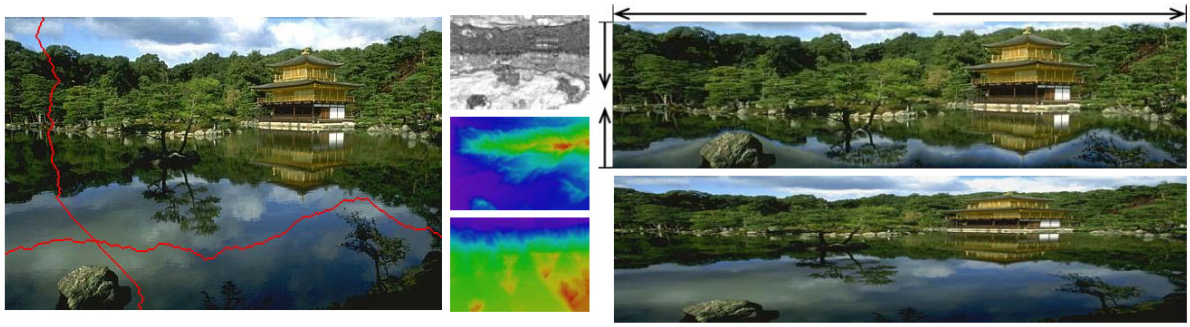
2.5.1 Seam Carving

O *Seam Carving* é um algoritmo de redimensionamento de imagens que considera e preserva o conteúdo da imagem. Apresentado por Avidan e Shamir (2007) como um algoritmo simples de operação de imagens, o *Seam Carving* busca encontrar as costuras (*seams*) da esquerda para a direita ou de cima para baixo. Para achar as *seams* ótimas, o algoritmo aplica uma função de energia no início da execução para destacar as regiões com alta energia, ou seja, com maior importância visual.

Podemos fazer duas operações com as *Seams*, adicionar ou remover, tanto da es-

querda para a direita, como de baixo para cima. Adicionar irá aumentar o tamanho da imagem e remover irá diminuir. O *Seam Carving* trabalha preservando o conteúdo da imagem que apresenta maior quantidade de energia e manipula o conteúdo com menor impacto visual, ou seja, as operações serão feitas sobre o conteúdo com menor quantidade de energia, ou seja aquele em que a *seam* que possui a menor quantidade de informação relevante de acordo com o mapa de energia, será removida.

Figura 10 – Etapas do *Seam Carving*, a esquerda as costuras(*Seams*) no centro os mapas de energia e as etapas intermediarias a direita temos a saída.



Fonte: Retirada de Avidan e Shamir (2007)

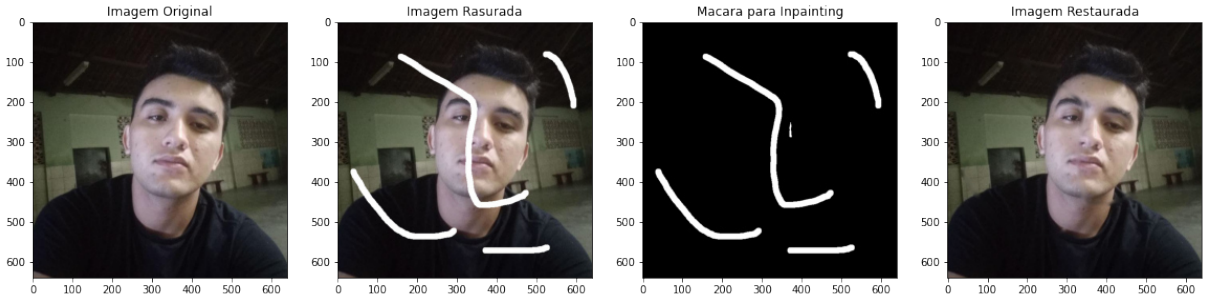
O *Seam Carving* pode ser usado como uma forma de remover objetos de imagens sem agredir de maneira brusca o conteúdo. Ele também pode ser usado para proteger um objeto de ser removido, dando para o objeto um peso, que pode ser atribuído com o auxílio de uma máscara. Com o peso aplicado, temos o aumento da energia daquele objeto, fazendo com que o *Seam Carving* o interprete como uma área de grande relevância para o conteúdo da imagem. Podemos aplicar valores mais altos ao conteúdo que se deseja proteger e valores menores para aqueles que queremos remover. Podemos ver as etapas do *Seam Carving* na Figura 10

2.5.2 Inpainting

O *Inpainting* é um processo de reconstrução digital de partes rasuradas ou perdidas de imagens e vídeos. Dada uma imagem que esteja danificada, podemos fazer o uso das técnicas de *Inpainting* para realizar diversas tarefas desde a restauração de fotografias, filmes e pinturas, a remoção de oclusões como texto, legendas, selos e publicidade, entre outros, como dito por Oliveira *et al.* (2001).

A implementação das técnicas *Inpainting* possuem diversas variações. Uma bastante conhecida por ser uma das mais simples é a técnica cuja ideia básica é substituir áreas em que os *pixels* foram danificados pela média dos vizinhos, para que a área reconstruída possa se camuflar com as áreas em bom estado dos vizinhos. O intuito da técnica é obter a informação que mais se aproxima da informação perdida da imagem original. As técnicas são muito variadas, então são selecionadas com base no objetivo que se quer alcançar e o tipo de imagem que será trabalhado. Na Figura 11 temos um exemplo de *Inpainting*.

Figura 11 – Exemplo de *Inpainting*.



Fonte: Elaborado pelo autor.

3 TRABALHOS RELACIONADOS

Nesta seção, são apresentados trabalhos que contribuem para o desenvolvimento da aplicação proposta neste trabalho, bem como uma breve comparação entre cada um deles em relação ao trabalho proposto.

3.1 *Inpainting* in Omnidirectional Images for Privacy Protection

No trabalho UPENIK *et al.* (2019), os autores elaboram um sistema de que irá tratar da proteção de informação através de métodos de *inpainting* para imagens omnidirecionais, que são imagens que tem uma imersão maior quando olhadas de diferentes ângulos. São usados métodos para que a remoção seja um processo reversível sobre a imagem, utilizando, para isso, três métodos de *Inpainting* diferentes para chegar no resultado esperado.

Diferente do apresentado por UPENIK *et al.* (2019), que lida apenas com correções aplicáveis a imagens produzidas com fontes omnidirecionais, o trabalho aqui apresentado lida com todos os tipos de imagens onde seja possível detectar objetos específicos automaticamente.

3.2 Image *Inpainting* Based on Inside–Outside Attention and *wavelet* Decomposition

No trabalho de He *et al.* (2020) temos uma abordagem diferente para realizar o *Inpainting*. Nesse trabalho os autores não apenas consideram o conteúdo ao redor do que foi removido, mas, também o contexto do que foi removido da pintura. Assim, o contexto ausente auxilia no processo de reconstrução, para, dessa forma, tentar tornar a pintura mais fidedigna possível.

A partir da implementação de um discriminador de componentes de textura via *wavelets* (GONZALEZ; WOODS, 2009), He *et al.* (2020) obtiveram uma melhora no desempenho e conseguiram mostrar que, ao contrário de outros métodos de reconstrução mais antiquados, seu método apresentou uma melhora significativa para restaurar imagens com estruturas mais complexas.

3.3 Flow-edge Guided Video Completion

O trabalho de Gao *et al.* (2020) foca no preenchimento de objetos em vídeos com alto desempenho, apresentando um cuidado especial sobre a coerência temporal do vídeo, que

segue com a ideia de que deve-se levar em conta a coerência das imagens, quando postas em conjunto na formação do video final. Ele apresenta a ideia de que não basta apenas trabalhar individualmente bem sobre uma única imagem por vez e, sim, de que devemos olhar para o contexto do vídeo e do aspecto temporal quando queremos excluir um objeto e posteriormente preencher o espaço deixado por ele. Com isso, ele garante que haja o mínimo possível de perda de informação ou quaisquer distorções no vídeo. Além disto, o algoritmo também consegue estimar o conteúdo externo das imagens através das informações nelas contidas para, assim, expandir seu conteúdo.

O preenchimento de vídeo *Video Completion* é a ação de preencher os espaços deixados vazios em *frames* de vídeo com conteúdos sintetizados a partir de outras informações retiradas do alvo original. Segundo Gao *et al.* (2020), ele herda os mesmos problemas que ocorrem com o preenchimento de imagens e traz um novo, que é a coerência temporal do vídeo.

A aplicação apresentada neste trabalho busca trazer a automação do processo de identificar os objetos a serem removidos ou protegidos nos cenários, divergindo de Gao *et al.* (2020), que necessitam que seja preciso demarcar de alguma forma externa os objetos de interesse.

3.4 Image Inpainting for Irregular Holes Using Partial Convolutions

No trabalho de Liu *et al.* (2018), os autores explicam que os métodos de *inpainting* que utilizam técnicas de aprendizagem profunda usam CNN sobre as imagens danificadas, para assim poder extrair informações de *pixels* válidos, a fim de utilizar a informação para reconstruir a imagem. Porém, essas abordagens normalmente usam a média de *pixels* como base para o *inpainting*. Esse tipo de abordagem comumente deixa artefatos nas áreas preenchidas, que posteriormente torna necessário algum pós-processamento, de forma a amenizar o impacto dos artefatos na imagem. Com isso em mente, Liu *et al.* (2018) fazem uso extensivo de uma operação de convolução mascarada ou reponderada, como também incluem um mecanismo que cria máscaras para as próximas convoluções. Eles utilizam operações de convolução parcial empilhadas e etapas de atualização de máscaras para realizar o *Inpainting*.

Neste trabalho, é utilizado o estudo de Liu *et al.* (2018) como base para criação do sistema proposto. O sistema proposto neste trabalho diverge de Liu *et al.* (2018) quando abre para a automatização que vem com o acréscimo de um algoritmo de detecção de objetos, não necessitando, assim, da parte manual de identificação do objeto a ser removido.

3.4.1 Generative Image Inpainting with Contextual Attention

O trabalho de Yu *et al.* (2018), traz uma estrutura de *inpainting* que utiliza atenção contextual para deixar o resultado mais natural. Nesse trabalho temos uma arquitetura baseada em duas redes neurais geradoras e duas descriminadoras. Uma das rede geradoras é usada para a reconstrução da parte mais bruta da imagem e a outra para partes em que é necessário um grau de refinamento. O nome dado para esse estilo de arquitetura é rede padrão de estrutura grosso e fino. Os dois discriminadores trabalham de modo semelhante, um deles sobre escopo global e outro sobre um escopo local da imagem pós preenchimento. O global age sobre a imagem completa, enquanto o local apenas pega a região que foi preenchida.

Esse trabalho possui um mecanismo de atenção contextual, utilizado para emprestar a informação contextual de locais espacialmente distantes, que será utilizada na reconstrução de pixeis ausentes. A informação que a atenção contextual adquire é utilizada na segunda rede geradora que trabalha com o refinamento. Já os descriminadores, tanto global quanto local, são utilizados para melhorar os detalhes de textura no local em que os pixeis são gerados. Podemos ver uma exemplo de *inpainting with contextual attention* na Figura 12.

Figura 12 – Exemplo de *inpainting* utilizando *Generative Image Inpainting with Contextual Attention*

Fonte: retirado de Yu *et al.* (2018)

3.5 Comparação entre os trabalhos relacionados

O Quadro 1 traz uma visão sobre algumas das principais características de cada trabalho relacionado apresentado. Assim, temos uma visão geral do tipo de tecnologia e da área de atuação de cada trabalho.

Quadro 1 – Quadro comparativo entre os trabalhos relacionados e este trabalho.

TRABALHO	**Principais técnicas aplicadas**	**Objetos de estudo**
UPENIK *et al.* (2019)	*Inpainting*	Imagens
He *et al.* (2020)	*Inpainting* e discriminadoror via wavelet	Imagens
Gao *et al.* (2020)	*Completion Video*	Vídeos
Liu *et al.* (2018)	*Partial convolution*, *Inpainting*	Imagens
Yu *et al.* (2018)	*Inpainting With Contextual Attention*	Imagens
Este Trabalho	*YOLO*, *YOLACT*, *Inpainting with Contextual Attention*	Imagens

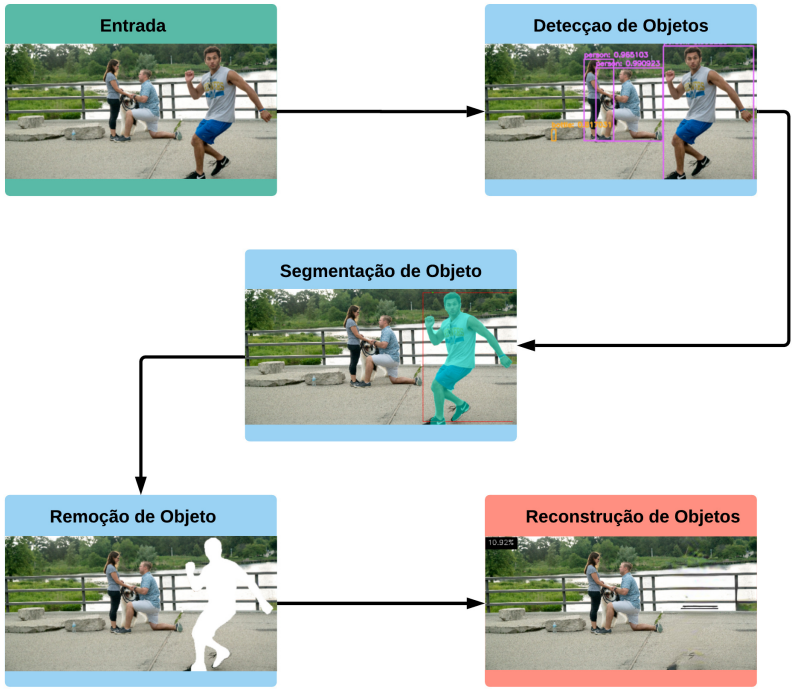
Fonte: Elaborado pelo autor.

4 METODOLOGIA

A concepção da aplicação teve como base os requisitos de cada um dos problemas que precisam ser atendidos para definição do processo final, levando em conta as várias técnicas usadas. Com esses requisitos definidos, podemos estabelecer um fluxo inicial para o funcionamento da aplicação, sendo possível fazer conjecturas sobre os resultados iniciais. Com isto, realizou-se uma revisão bibliográfica que partiu do estudo do estado da arte das técnicas que são usadas para resolver os problemas já mencionados, buscando entender quais serão as melhores métricas de avaliação que possam facilitar o desenvolvimento da aplicação final.

Assim, temos a concepção de como o fluxo da aplicação deve funcionar, como também o entendimento dos requisitos que a aplicação deve satisfazer para alcançar o resultado desejado. Desta forma, este trabalho propõe uma aplicação com processos bem definidos que iniciam com a detecção de objetos em uma imagem, que será usada como entrada em um processo de segmentação de objetos. A saída desta etapa será o objeto demarcado com uma máscara que será enviada para o processo de remoção de conteúdo para remover o objeto demarcado. Em seguida, a imagem resultante será usada pelo processo de reconstrução da imagem, que irá reconstruir a imagem sem o objeto removido e, por fim, teremos como saída uma imagem reconstruída. As etapas do sistema proposto são mostradas na Figura 13.

Figura 13 – Visão geral do sistema proposto



Fonte: Elaborado pelo autor.

4.1 Visão Geral do Sistema

O modelo proposto para a aplicação é composto de uma série de etapas para tratar o problema apresentado. As etapas que devem ser executadas são:

1) Detecção de objetos: utilizando o *YOLO*;
2) Seleção do objeto por parte do usuário: através de ação de *click* ou digito;
3) Segmentação: utilizando o *YOLACT*;
4) Remoção: utilizando técnicas de processamento de imagens tais como: combinações de imagens e dilatação;
5) Restauração: utilizando a técnica proposta por Yu *et al.* (2018);

De inicio, aplicação deve receber uma imagem que será carregada pela etapa de detecção de objetos. O YOLO trata a detecção de objetos como um problema de regressão, simultaneamente aprendendo as coordenadas das caixas delimitadoras e as probabilidades de rótulos de classes correspondentes presentes na imagem, caso existam objetos válidos a serem detectados.

Antes da segunda etapa ser inciada, é necessário a intervenção pela parte do usuário, para escolher qual dos objetos detectados pelo *YOLO* na etapa anterior, será removido. Após o usuário escolher o objeto, esse objeto será recortado a partir das coordenadas das caixas delimitadoras e inserido na segunda etapa. Esse recorte é feito para diminuir a quantidade de tempo e processamento na etapa de segmentação, devido a não ser necessário toda a imagem, como também para aumentar a precisão da segmentação, que funciona melhor quando o objeto possui um destaque maior.

Na Segunda etapa, o sistema irá receber o objeto recortado, e irá passar pelo *YOLACT* o segmentador de Bolya *et al.* (2019), treinado com a base de dados COCO, sendo capaz de segmentar mais de 80 tipos de diferentes de objetos. O *YOLACT* processa simultaneamente diversos protótipos de mascaras e encontra os coeficientes dos protótipos de máscaras daquele objeto. No final, ele segmenta combinando linearmente as mascaras e os coeficientes, as instâncias dessas combinações que atingirem um limiar determinado do algoritmo serão segmentados.

Já na etapa de remoção, será realizada a substituição de todos os pixeis de uma área delimitada por pixels brancos, área essa delimitada pela máscara resultante referente ao objeto segmentado na segunda etapa. A máscara passa por uma operação morfológica de dilatação que irá suavizar as bordas, aumentando o tamanho da máscara, buscando reduzir os ruídos, que podem aparecer na etapa de reconstrução, causados pela segmentação. Após a operação, a

máscara pode ser usada para substituir os pixels do objeto.

No passo final, a etapa de reconstrução, é utilizada a técnica de *Inpainting With Contextual Attention* de Yu *et al.* (2018), que receberá a imagem com o objeto removido e a máscara dilatada utilizada na etapa de remoção. A técnica utilizar duas redes neurais geradoras e duas discriminadoras para entender o contexto, realizar a reconstrução da área demarcada pela pela máscara e fazer ajustes mais finos na área reconstruída. No final, temos como saída a imagem sem o objeto anteriormente escolhido, e com a área do objeto reconstruída.

5 RESULTADOS

Nesta seção são apresentados os resultados obtidos a partir do sistema proposto neste trabalho. Com o intuito de demonstrar uma forma de utilizar este sistema foi montado o conjunto de imagens em torno de imagens que apresentem *Photobombing*, para que os testes de perfomace fossem aplicados e os seus resultados classificados, para assim descobrir quais seriam os objetos que tiveram mais sucesso de serem removidos. *Photobombing* seria o ato de aparecer em um fotografia/imagem de forma a causar humor como definido em no Dicionário Online de Cambridge (DICTIONARY,).

Os testes iniciais da solução proposta foram realizados sobre um banco de dados de teste composto de imagens extraídas online de bancos de dados conceituados de Visão Computacional como Imaginet e *COCO* - (*Common Objects in COntext*).

As imagens resultantes obtidas na saída do sistema foram classificadas como RUIM, RAZOÁVEL ou BOA, baseado na observação do autor, com o intuito de avaliar o resultado final. Os critérios adotados foram os seguintes:

- RUIM: A imagem de saída apresenta notáveis deformidades ou incongruências, o local do objeto possui grande disparidade para o restante do conteúdo da imagem, ou partes do objetos não foram excluído;
- RAZOÁVEL: A imagem apresenta pequenas deformidades perceptíveis ao observador, que pode ser capaz de causar algum estranhamento ao observar a imagem;
- BOA: na imagem não é perceptível a remoção do objeto, ou, de forma muito sutil percebe-se mudanças em texturas ao longo do local do objeto removido.

As imagens em que os resultados são considerados "aceitáveis", são classificados como RAZOÁVEL ou BOA. Isso significa que a imagem resultante da aplicação do sistema, foi considerada adequada à proposta deste trabalho. Já RUIM, significa que o resultado é completamente inadequado à proposta inicial deste sistema, tendo falhado em obter bons resultados.

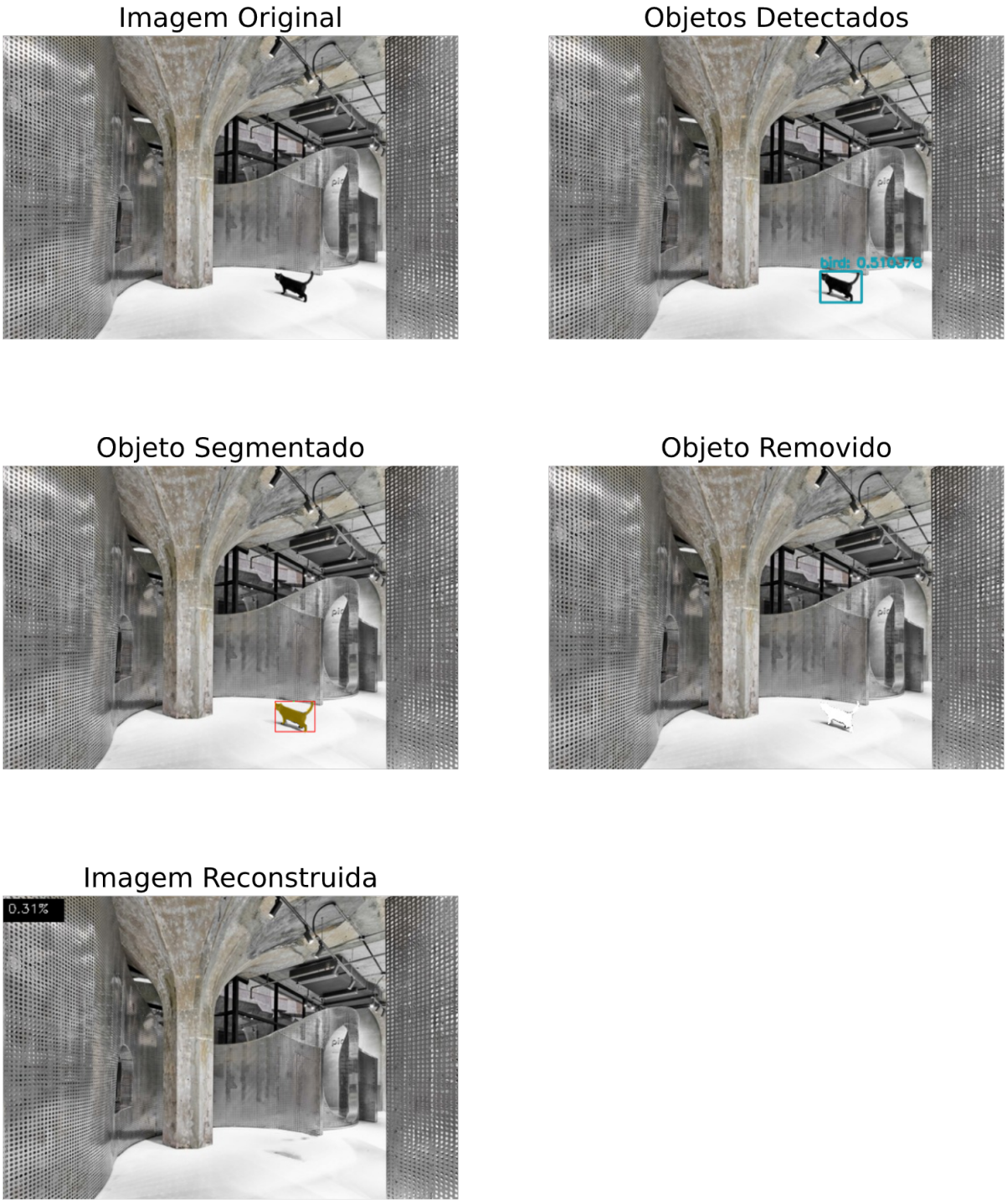
5.1 Apenas um objeto retirado

Da Figura 14 a Figura 24, temos exemplos de apenas um objeto sendo retirado.

Figura 14 – Resultados com remoção de apenas 1 objeto - Exemplo 01

Fonte: imagem Original - https://www.tediado.com.br/01/25-animais-campeoes-de-photobomb/

Figura 15 – Resultados com remoção de apenas 1 objeto - Exemplo 02

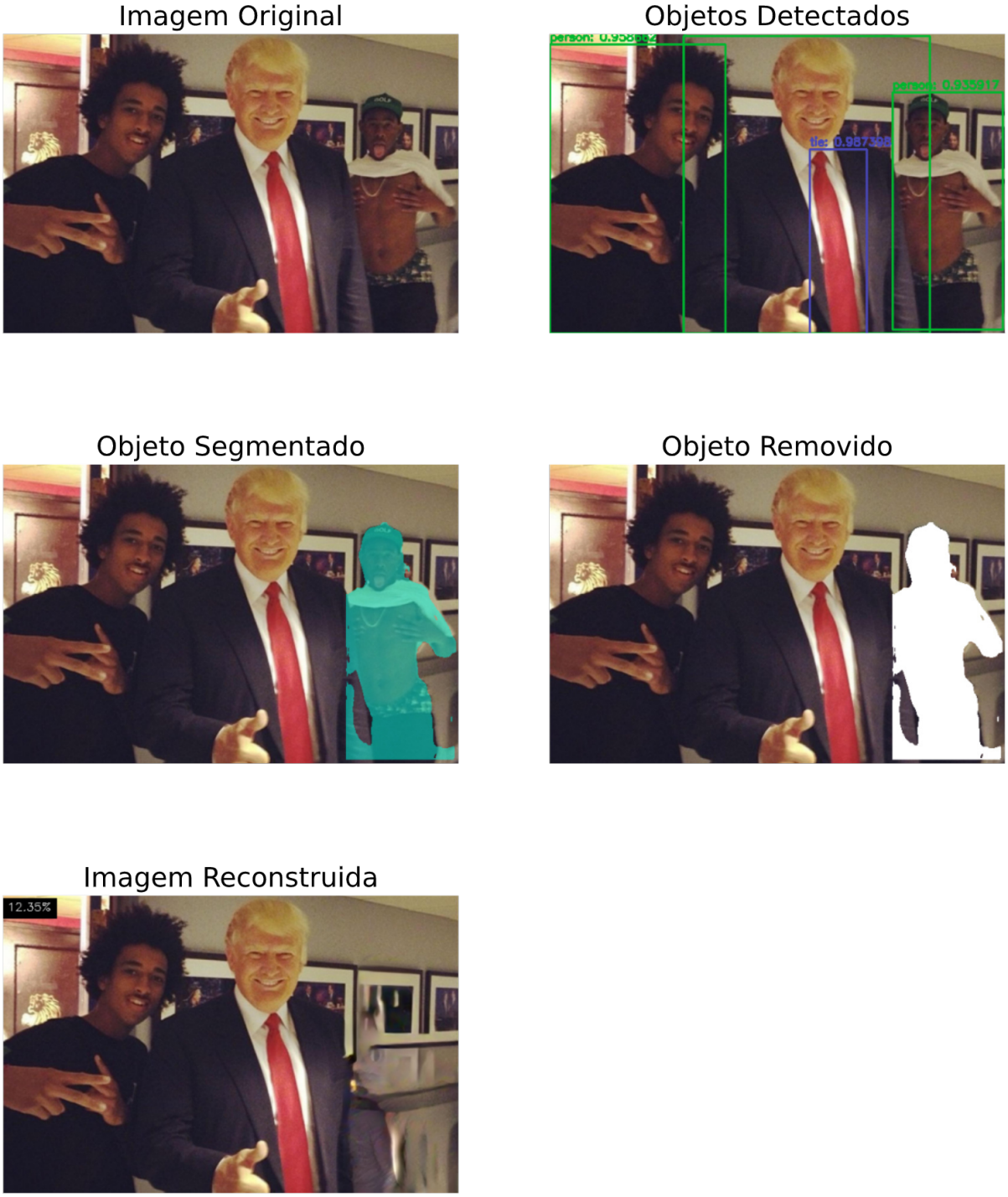


Fonte: imagem Original - https://www.sohu.com/a/438401207_120969244

Figura 16 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 03

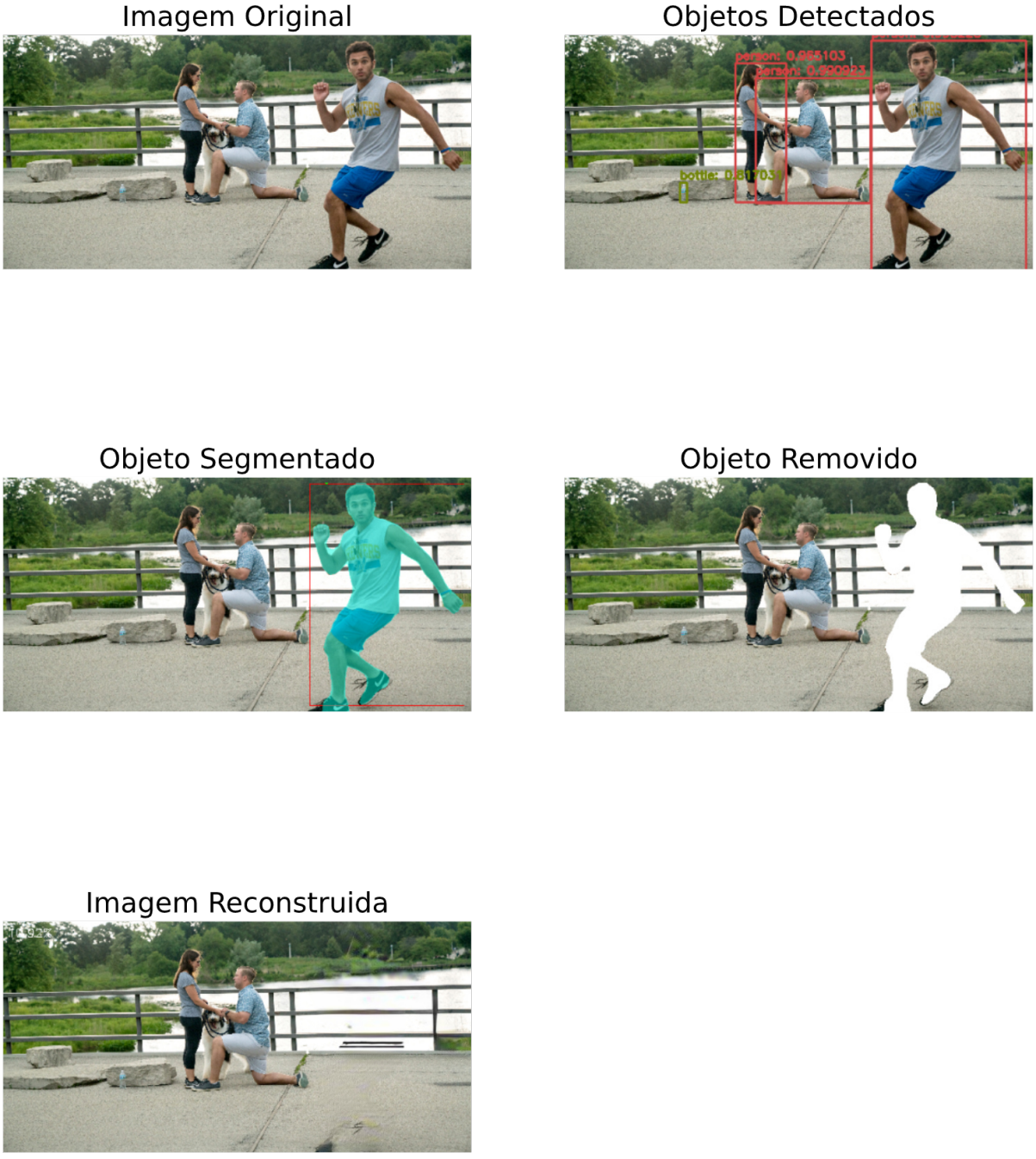
Fonte: imagem Original - https://www.liveabout.com/very-funny-photobombs-taken-at-the-beach-1923865

Figura 17 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 04



Fonte: imagem Original - https://www.reddit.com/r/photobomb/comments/6b8yvt/now_thats_what_you_call_a_photobomb/

Figura 18 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 05



Fonte: imagem Original - https://www.today.com/style/jogger-photobombs-couple-hilarious-engagement-photos-t157744

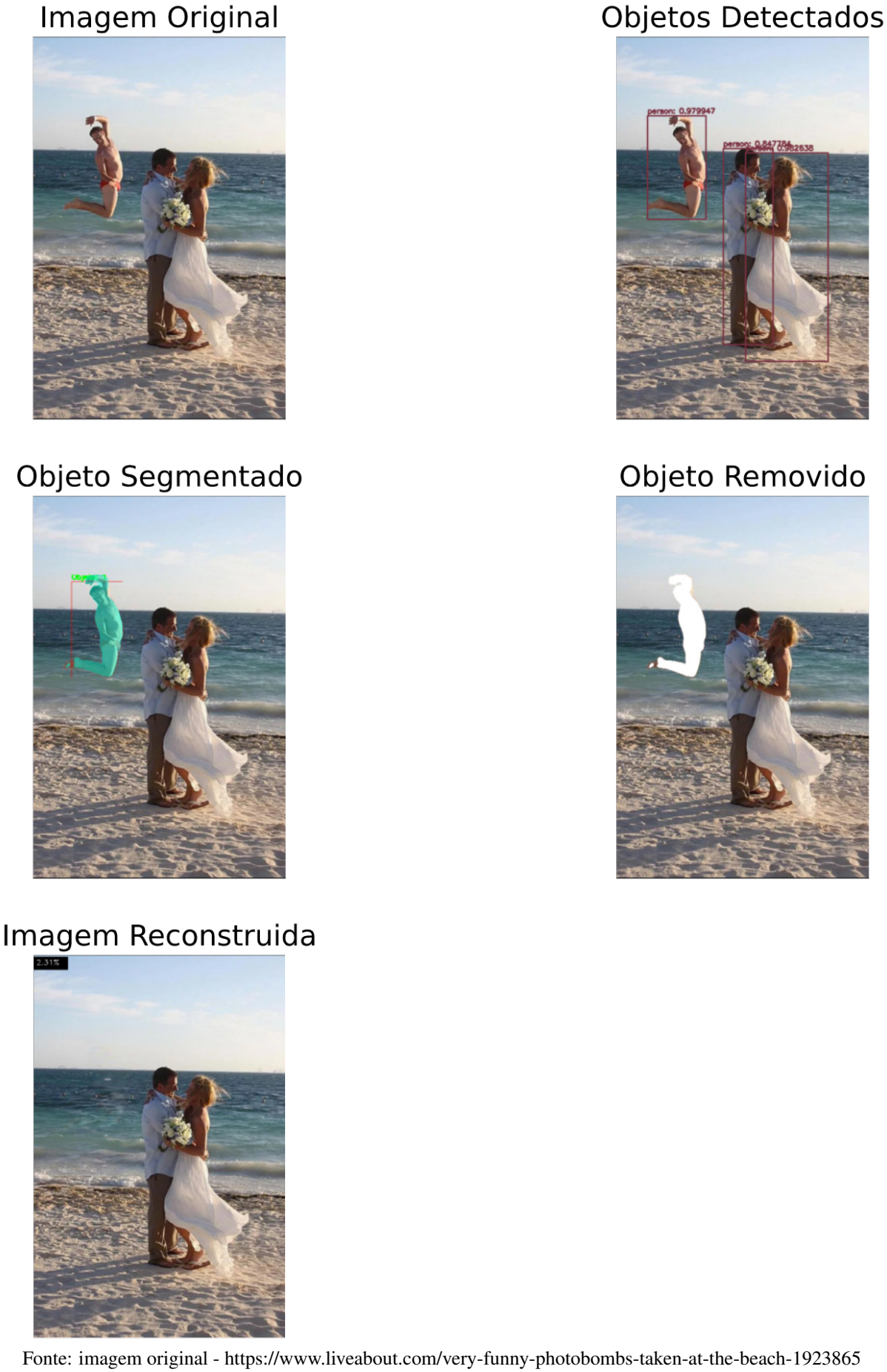
Figura 19 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 06

Fonte: imagem original - https://zen.yandex.ru/media/blogfamily/samye-nelepye-svadebnye-fotografii-5c0d076\protect\@normalcr\relax86bc00a9389c16

Figura 20 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 07

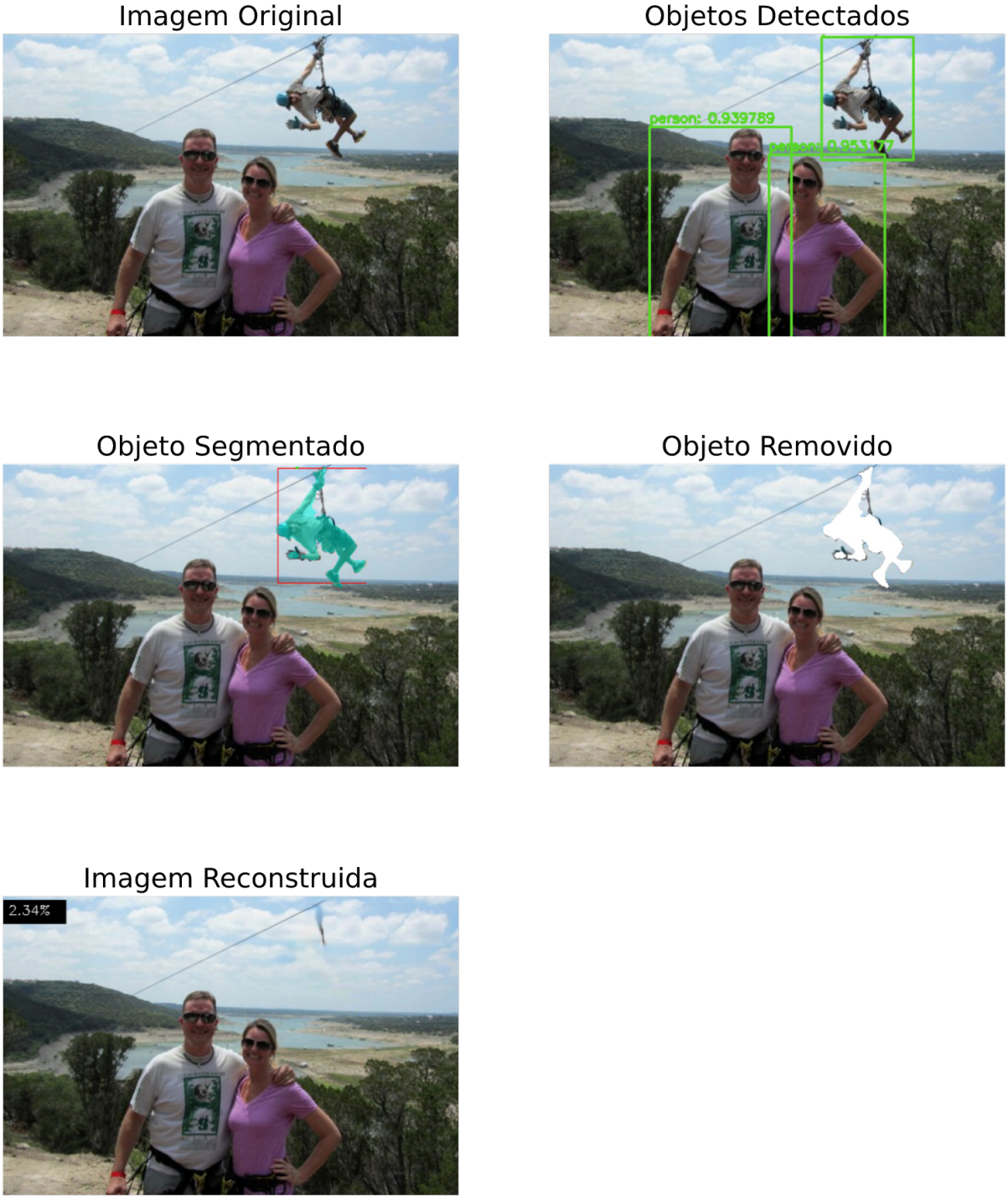
Fonte: imagem original - https://www.liveabout.com/the-top-funny-animal-photobombs-on-the-internet-1923875

Figura 21 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 08



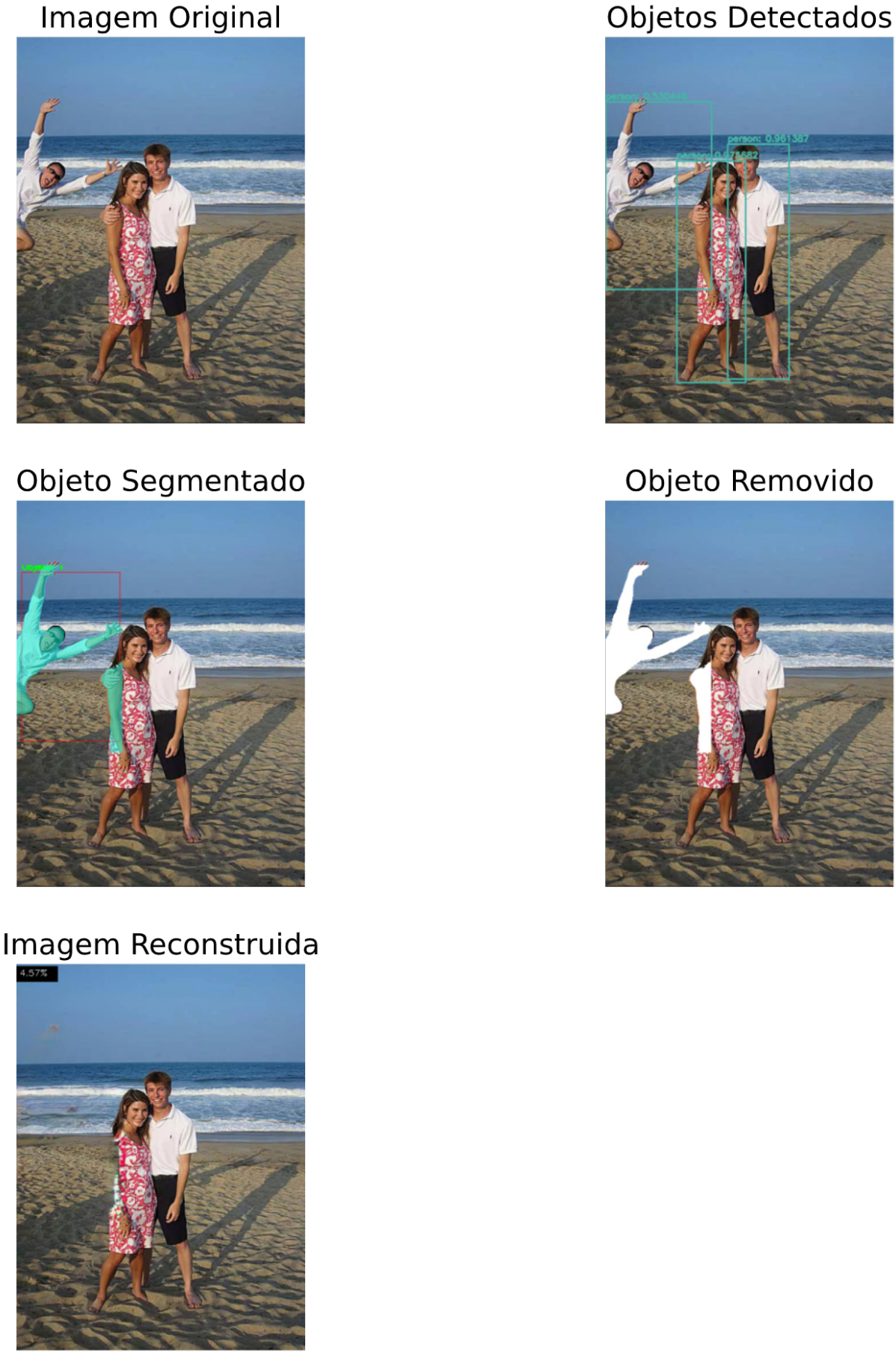
Fonte: imagem original - https://www.liveabout.com/very-funny-photobombs-taken-at-the-beach-1923865

Figura 22 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 09



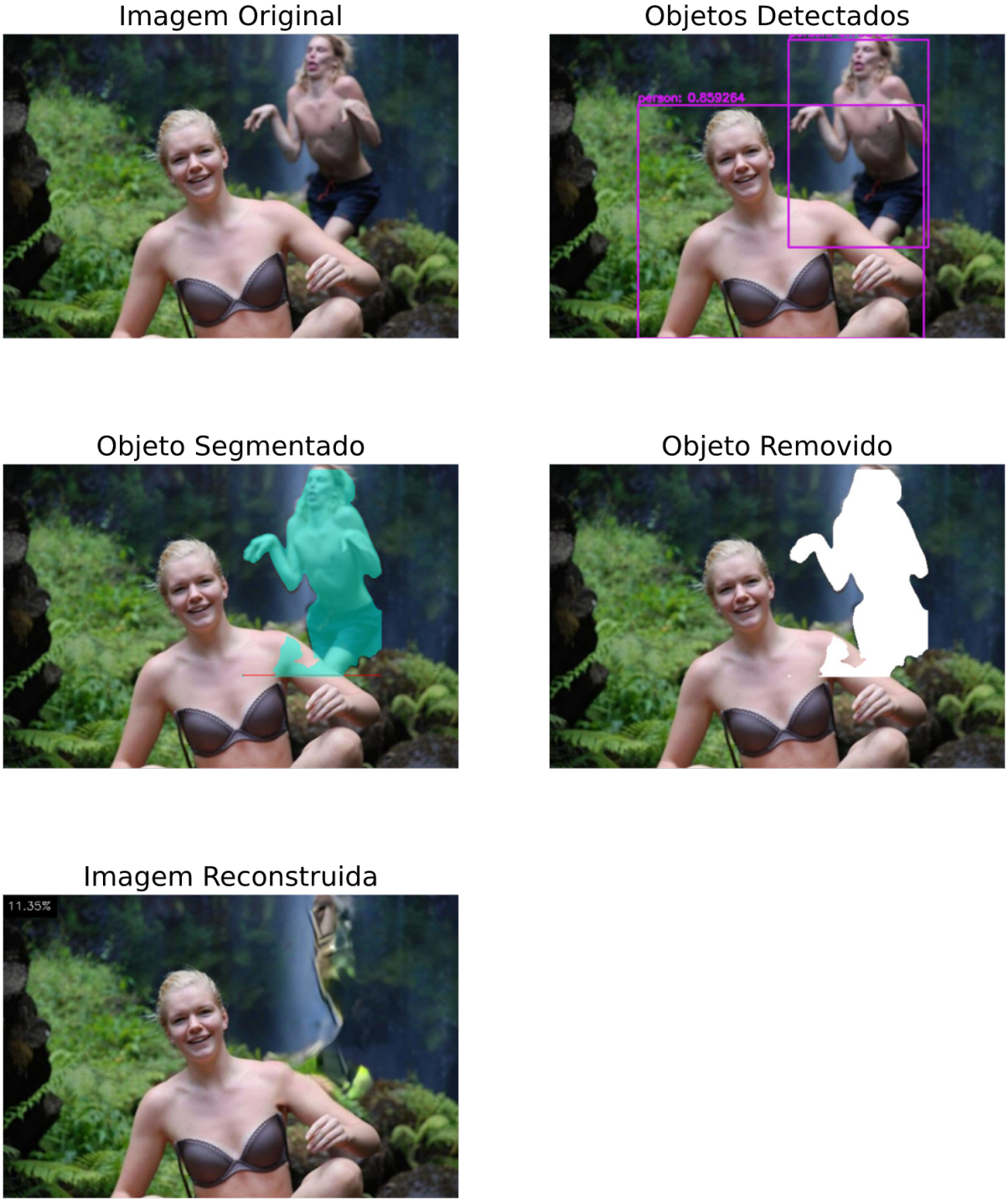
Fonte: imagem original - http://ilexcinema.com.br/2019/07/18/5-dicas-para-fotos-e-videos-incriveis/

Figura 23 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 10



Fonte: imagem original - https://www.liveabout.com/very-funny-photobombs-taken-at-the-beach-1923865

Figura 24 – Resultados de imagens que foi retirado apenas 1 objeto - Exemplo 11



Fonte: imagem original - https://funnyjunk.com/funny$_p$*ictures*/3720170/*Jurasic* + *park*/

Tabela 1 – Resumo das informações das Figuras 14 a 27.

Figuras	Área do Objeto Removido	Resultado
Figura 14	1,13%	BOA
Figura 15	0.13%	BOA
Figura 16	5.42%	BOA
Figura 17	11.34%	RAZOÁVEL
Figura 18	10.92%	BOA
Figura 19	0.89%	BOA
Figura 20	8.04%	BOA
Figura 21	2.31%	BOA
Figura 22	2.34%	BOA
Figura 23	4.57%	RUIM
Figura 24	11.35%	RAZOÁVEL

Fonte: Elaborado pelo autor.

Nas figuras que tiveram apenas um objeto retirado, foi possível detectar que as maiores influencias que a imagem sofre, é sobre os aspectos das texturas e tamanho do objeto retirado. Podemos supor, baseado nisso que quanto mais uniforme forem as texturas do meio, que objeto estava situado, melhor será sua reconstrução baseada no contexto simples desas texturas. Como também vemos que o tamanho é de fato um fator a se considerar, já que na maioria dos resultados positivos, foram figuras que tinham objetos relativamente menores a serem removido. Podemo verificar na Tabela 1 podemos ver uma visão geral dos resultados das figuras que tiveram apenas um objeto removido.

5.2 Mais de Um Objeto Para Ser Retirado

Nesta seção vamos mostrar exemplos de objetos diferentes retirados da mesma imagem, objetos com reflexos e sobras presentes e de objetos sobrepostos.

5.2.1 Objetos Diferentes Retirados da Mesma Imagem

Na Figura 25 e Figura 26 removemos dois objetos que se originam da mesma imagem. Os objetos removidos, no caso dos pássaros preto e azul, respectivamente das figuras 25 e 26, ocupavam 0.93% e 14.87% da área total das suas respectivas imagens. A textura que existe no local em que ambos os pássaros são removidos é simples, além de ser borrada, que diminua muito o nível de detalhes, e isto ajuda para a reconstrução. Esta imagem demonstra que é possível retirar objetos diferentes da mesma imagem, sem afetar os demais objetos, ou seja podemos retirar esses objetos sem ter grandes alterações ao redor.

Figura 25 – Resultado de objetos retirados da mesma imagem - Exemplo 01.

Fonte: imagem original - https://blog.nature.org/science/2019/07/22/why-do-little-birds-mob-big-birds/

Figura 26 – Resultado de objetos retirados da mesma imagem - Exemplo 02.

Fonte: imagem original - https://blog.nature.org/science/2019/07/22/why-do-little-birds-mob-big-birds/

5.2.2 Objetos com reflexo

Na Figura 27 é possível verificar que o tamanho diminuto do objeto ajuda a reconstrução junto com as texturas simples que ele está em contato, porém mesmo o objeto sendo retirado efetivamente, seu reflexo permanece na piscina, sendo esse um caso, qual a etapa de detecção não foi capaz de detectar o reflexo na piscina, como nem mesmo reconhecer que pertence ao objeto retirado.

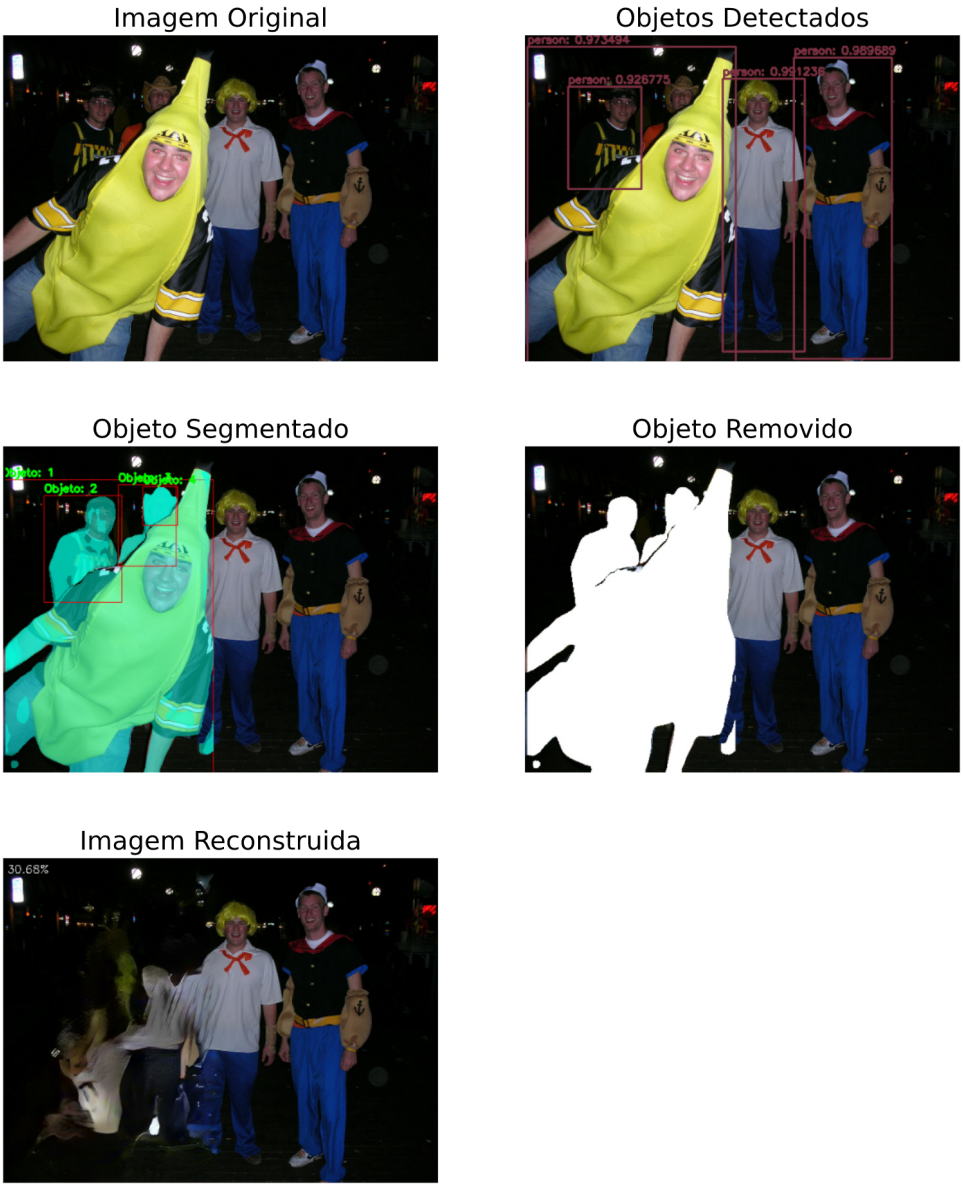
Figura 27 – Resultado de imagem com objetos que possuem reflexos.

Fonte: imagem original - https://www.20min.ch/story/bradley-cooper-versaut-selfie-mit-brad-pitt-647002348796

5.2.3 Objetos Sobrepostos

O tamanho do objeto a ser retirado é demasiado grande, que é a maioria dos casos de sobreposição, tornando o processo de reconstrução muito complexo, além do objeto se sobrepor com outros objeto que não queremos retirar, o sistema tenta restaurar a área do objeto da melhor maneira, juntamente com os objetos que estão sobrepostos, mas esses objetos acabam se tornando distorcidos devido a grande areá que o algoritmo tenta reconstruir, com a pouca informação sobre os objetos que foram sobrepostos.

Figura 28 – Resultado de imagens com objetos sobrepostos



Fonte: imagem original - https://en.wikipedia.org/wiki/Photobombing

Tabela 2 – Mais de Um Objeto Para Ser Retirado figuras 25 a 28

Figuras	Área do Objeto Removido	Resultado
Figura 25	0.93%	BOA
Figura 26	14.87%	RAZOÁVEL
Figura 27	1.25%	BOA
Figura 28	30.68%	RUIM

Fonte: Elaborado pelo autor.

Na Tabela 2, temos uma visão geral sobre os resultados das figuras de 25 a 28, que representam casos de mais de um objeto sendo retirado. Na maioria das imagens que tiveram objetos removido que ocupavam uma areá menor a 10% tivemos uma alta taxa de sucesso e resultados classificados em sua maioria como BOA e algumas RAZOÁVEL, como podemos observar o Capitulo 5, também tivemos a contra parte para objetos que ocupavam uma areá superior a 10% que apresentavam notas em geral RUIM e algumas RAZOÁVEL. Também é notado que para objetos que tinham áreas superiores a 20% sempre aprestaram notas RUIM.

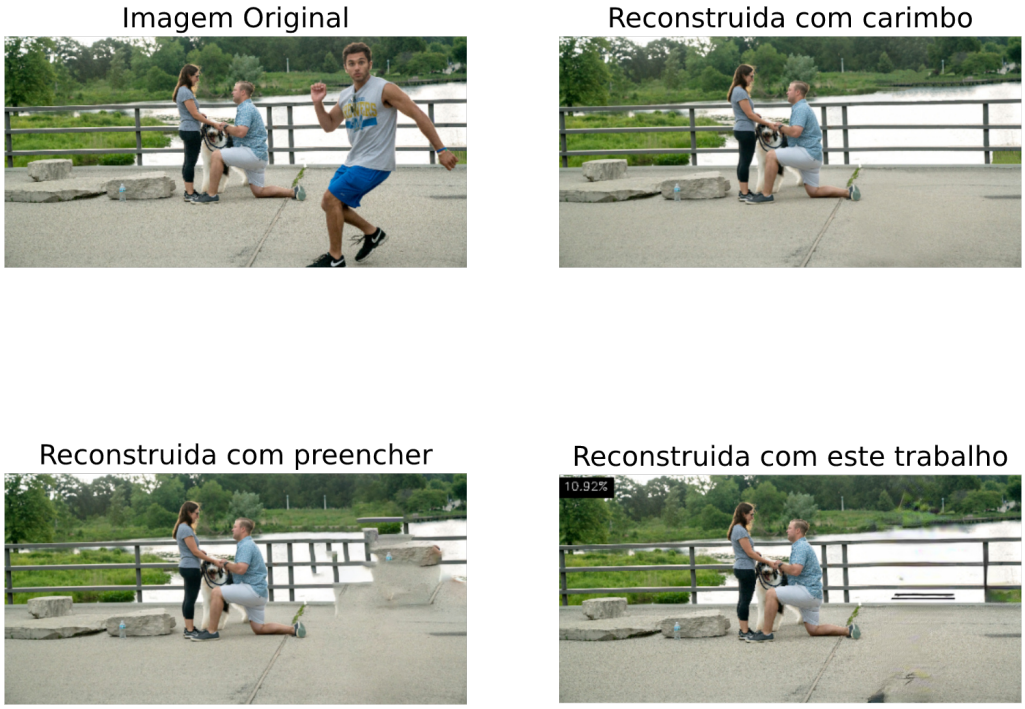
5.3 Comparação com ferramentas do Photoshop

Na Figura 29 e 30, temos uma comparação dos resultados de ferramentas de reconstrução do *Photoshop*®, uma das ferramentas de edição de imagens mais usadas no mundo, com o sistema desenvolvido. As ferramentas que foram utilizadas foram o carimbo e o preencher. Todos os resultados das figuras 29 e 30 adquiridos por meios dessas ferramentas foram obras de uma estudante de *Design Digital*, que trabalha profissionalmente com *PHOTOSHOP® 2020*.

O carimbo, é uma ferramenta mais manual para realização do processo de reconstrução, em que tem-se que analisar o entorno do objeto removido e copiar o conteúdo dos arredores para o local da reconstrução, sendo um trabalho demorado e que exigir uma técnica mais apurada para melhores resultados, levando dezenas de minutos para um bom trabalho, dependendo da quantidade de detalhes da imagem a ser reconstruída.

Preencher seria uma ferramenta de reconstrução que age de uma forma mais automática, para qual é necessário uma segmentação manual do objeto que será removido, após a segmentação ser realizada um simples comando *SHIFT+F5* realiza o processo de reconstrução. Em media leva-se apenas alguns segundos para realização da reconstrução, mas a ferramenta demanda um segmentação precisa e muitas vezes pode devolver um resultado não muito natural. Para objetos que não tem um alto nível de detalhamento demonstra um bom desempenho como mostrado na figura 31.

Figura 29 – Primeira comparação entre resultados das ferramentas do Photoshop para reconstrução e o sistema desenvolvido. Respectivamente temos da esquerda para direita e de cima para baixo, a imagem original, a imagem reconstruída utilizando carimbo, a imagem reconstruída utilizando preencher e por ultimo a imagem resultante deste trabalho.



Fonte: Elaborado pelo autor.

Quadro 2 – Quadro de comparação figura 29

Ferramenta	**Tempo Aproximado**	**Resultado**	**Complexidade de Uso**	***Segmentação***
Carimbo	20 minutos	BOM	Alta	Manual
Preencher	15 Segundos	RUIM	Baixa	Manual
Este trabalho	10.6 segundos	BOM	Baixa	Automática

Fonte: Elaborado pelo autor.

No Quadro 2 podemos verificar a comparação entre as ferramentas, na qual podemos verificar que o sistema desenvolvido consegue apresentar um resultado bom em comparação com o resultado da ferramenta carimbo, em um tempo muito menor, como também exigir menos técnica o que diminui muito a complexidade de uso em comparação ao carimbo. Quando comparado com a ferramenta preencher temos uma pequena diferença no tempo para aplicação, com ambos possuindo uma baixa complexidade para se usar, com preencher temos uma resultado ruim em comparação com o sistema desenvolvido.

Na Quadro 3 podemos verificar mais exemplos de comparações entre as ferramentas e o sistema desenvolvido, sobre a Figura 30, no qual podemos verificar que o sistema desenvolvido consegue apresentar um resultado bom em comparação com o resultado da ferramenta carimbo, mas trás pontos negativos quando se trata do reflexo, o sistema desenvolvido continua

executando em um tempo muito menor, como também exigir menos técnica o que diminui muito a complexidade de uso em comparação ao carimbo.

Figura 30 – Segunda comparação entre resultados das ferramentas do Photoshop para reconstrução e o sistema desenvolvido neste trabalho.

Fonte: Elaborado pelo autor.

Figura 31 – Terceira comparação entre resultados das ferramentas do Photoshop para reconstrução e o sistema desenvolvido neste trabalho.

Fonte: Elaborado pelo autor.

Quadro 3 – Quadro de comparação figura 30

Ferramenta	**Tempo Aproximado**	**Resultado**	**Complexidade de Uso**	***Segmentação***
Carimbo	7 minutos	BOM	Alta	Manual
Preencher	15 Segundos	RUIM	Baixa	Manual
Este Trabalho	16.6 segundos	BOM	Baixa	Automática

Fonte: Elaborado pelo autor.

6 CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma proposta de sistema capaz de remover objetos de imagens do modo mais automático possível de forma a não danificar perceptivelmente o conteúdo restante da imagem. Para isso foi utilizado técnicas de Visão Computacional e *Deep Learning*, tais como detecção de objetos, segmentação de objetos e técnica de *Inpainting*. Através da combinação das técnicas *YOLO*, *YOLACT* e *GENERATIVE IMAGE INPAINTING*.

O sistema demonstrou uma eficiência elevada para a retirada de objetos que possuíam um área que em geral é entorno ou menor a 10% da área total da imagem, assim podemos estabelecer que o sistema é adequado para remoções de objetos que não sejam demasiados grandes, devido que todos os testes feitos com objetos que ocupavam mais de 20% da imagem tiveram bons resultados . O sistema também mostrou que, comparado a ferramentas presentes no *PHOTOSHOP*® um dos mais conhecidos editores de imagens, consegue bons resultados em imagens que tiveram objetos removidos, com o mínimo de participação do usuário.

Como trabalhos futuros, podemos destacar a utilização ou o acoplamento de técnicas que possam trabalhar sobre sombras e reflexos, para que o sistema seja capaz de detectar sombras e reflexos dos objetos removidos e removê-los juntamente. Como também a extensão para trabalhar com a remoção de objetos em tempo real como (GAO *et al.*, 2020) e técnicas que possam contornar de forma mais eficiente o problema de objetos sobrepostos.

REFERÊNCIAS

AVIDAN, S.; SHAMIR, A. Seam carving for content-aware image resizing. In: ACM TRANS. GRAPH. [*S. l.*]: ACM, 2007. v. 26, n. 3, p. 10.

BALLARD, W. **Hands-On Deep Learning for Images with TensorFlow**. Birmingham: Packt, 2018.

BOLYA, D.; ZHOU, C.; XIAO, F.; LEE, Y. J. Yolact: Real-time instance segmentation. In: **ICCV**. [*S. l.*: *s. n.*], 2019.

BRASIL. **Lei Nº 13.709**. Brasília, DF, 2018. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/L13709.htm. Acesso em: 10 jan. 2021.

CHOLLET, F. **Deep Learning with Python**. [*S. l.*]: Manning, 2017. ISBN 9781617294433.

DICTIONARY, C. **photobombing**. Disponível em: https://dictionary.cambridge.org/dictionary/english/photobombing. Acesso em: 10 jun. 2021.

GAO, C.; SARAF, A.; HUANG, J.-B.; KOPF, J. Flow-edge guided video completion. In: **Proc. European Conference on Computer Vision (ECCV)**. [*S. l.*: *s. n.*], 2020.

GERON, A. **Hands-on machine learning with Scikit-Learn and TensorFlow**. Beijing: O'Reilly,, 2017. Includes QR code.

GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: **Computer Vision and Pattern Recognition**. [*S. l.*: *s. n.*], 2014.

GONZALEZ, R.; WOODS, R. **Processamento Digital De Imagens**. [*S. l.*]: Addison Wesley Bra, 2009. ISBN 9788576054016.

HE, X.; CUI, X.; LI, Q. Image inpainting based on inside–outside attention and wavelet decomposition. In: IEEE ACCESS. **Institute of Electrical and Electronic Engineers**. [*S. l.*], 2020. p. 1–1.

HEARST, M. A.; DUMAIS, S. T.; OSUNA, E.; PLATT, J.; SCHOLKOPF, B. Image inpainting based on inside–outside attention and wavelet decomposition. In: IEEE. **IEEE Intelligent Systems and their Applications**. [*S. l.*], 1998. p. 18–28.

LAAKSONEN, J.; OJA, E. Classification with learning k-nearest neighbors. In: ICNN'96. **Proceedings of International Conference on Neural Networks**. [*S. l.*], vol.3, 1996. p. 1480–1483.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998.

LIU, G.; REDA, F. A.; SHIH, K. J.; WANG, T.-C.; TAO, A.; CATANZARO, B. Image inpainting for irregular holes using partial convolutions. In: ARXIV. [*S. l.*], 2018.

LIU, W.; ANGUELOV, D.; ERHAN, D.; SZEGEDY, C.; REED, S.; FU, C.-Y.; BERG, A. C. Ssd: Single shot multibox detector. **Lecture Notes in Computer Science**, Springer International Publishing, p. 21–37, 2016. ISSN 1611-3349.

OLIVEIRA, M.; BOWEN, B.; MCKENNA, R.; CHANG, Y.-S. Fast digital image inpainting. In: **Proceedings of the International Conference on Visualization, Imaging and Image Processing**. [*S. l.*: *s. n.*], 2001. p. 261–266.

OYELADE, J.; ISEWON, I.; OLADIPUPO, O.; EMEBO, O.; OMOGBADEGUN, Z.; AROMOLARAN, O.; UWOGHIREN, E.; OLANIYAN, D.; OLAWOLE, O. **Data Clustering**: Algorithms and its applications. In: **ICCSA**. [*S. l.*: *s. n.*], 2019. p. 71–81.

REDMON, J.; FARHADI, A. **YOLOv3**: An incremental improvement. In: ARXIV. [*S. l.*], 2018.

ROSEBROCK, A. **Deep Learning for Computer Vision with Python**. [*S. l.*]: PyImageSearch, 2017.

SEWAK, M.; KARIM, M. R.; PUJARI, P. **Practical Convolutional Neural Networks**. [*S. l.*]: Packt Publishing, 2018. ISBN 1788392302.

UPENIK, E.; AKYAZI, P.; TUZMEN, M.; EBRAHIMI, T. Inpainting in omnidirectional images for privacy protection. **ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**, p. 2487–2491, 2019.

YU, J.; LIN, Z.; YANG, J.; SHEN, X.; LU, X.; HUANG, T. S. Generative image inpainting with contextual attention. In: **arXiv**. [*S. l.*: *s. n.*], 2018.

YUMING HUA; JUNHAI GUO; HUA ZHAO. Deep belief networks and deep learning. In: **Proceedings of International Conference on Intelligent Computing and Internet of Things**. [*S. l.*: *s. n.*], 2015. p. 1–4.