

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS
GERAIS**

NÚCLEO DE EDUCAÇÃO A DISTÂNCIA

**Pós-graduação Lato Sensu em Inteligência Artificial e
Aprendizado de Máquina**

Amanda Costa Spolti

**Utilizando YOLOX e YOLOv5 para Detecção
de Máscaras**

Contagem, Brasil

Outubro, 2021

Amanda Costa Spolti

Utilizando YOLOX e YOLOv5 para Detecção de Máscaras

Trabalho de Conclusão de Curso apresentado
ao Curso de Especialização em Inteligência
Artificial e Aprendizado de Máquina, como
requisito parcial à obtenção do título de Es-
pecialista.

Contagem, Brasil
Outubro, 2021

Resumo

Com o constante avanço tecnológico, desde a recursos computacionais até a criação de algoritmos de Deep Learning cada vez mais avançados abriu-se um leque enorme de aplicações que façam uso dessas tecnologias. O objetivo deste trabalho é a aplicação de modelos de Deep Learning voltados para a área de visão computacional, para detectar o uso correto de máscaras. Para isso, foi utilizado duas arquiteturas, YOLOX e YOLOv5. A arquitetura YOLOX obteve uma AP total de 95,52%, map_5095 de 76,77% e map_50 de 95,51%. Já o YOLOv5 obteve 79,6%, 47,70% e 82,11% respectivamente. Os resultados mostraram a eficiência de ambas as arquiteturas para a detecção do uso correto de máscaras.

Palavras-chave: deep learning, yolox, yolov5, object detection, computer vision, visão computacional, detecção de objetos, detecção de máscaras

Lista de ilustrações

Figura 1 – Exemplo de uma imagem utilizada no treinamento do modelo.	10
Figura 2 – Figura que mostra as diferenças entre YOLOX e YOLOv3-5 (GE et al., 2021)	12
Figura 3 – Imagem original (exemplo 1)	14
Figura 4 – Inferência com YOLOX (exemplo 1)	14
Figura 5 – Inferência com YOLOv5 (exemplo 1)	15
Figura 6 – Imagem original (exemplo 2)	16
Figura 7 – Inferência com YOLOX (exemplo 2)	16
Figura 8 – Inferência com YOLOv5 (exemplo 2)	17
Figura 9 – Imagem original (exemplo 3)	18
Figura 10 – Inferência com YOLOX (exemplo 3)	18
Figura 11 – Inferência com YOLOv5 (exemplo 3)	19
Figura 12 – Imagem original (exemplo 4)	20
Figura 13 – Inferência com YOLOX (exemplo 4)	20
Figura 14 – Inferência com YOLOv5 (exemplo 4)	21

Lista de tabelas

Tabela 1 – Distribuição das imagens	11
Tabela 2 – Métricas obtidas	13

Sumário

1	INTRODUÇÃO	6
1.0.1	Visão Computacional e Detecção de Objetos	6
2	DESCRIÇÃO DO PROBLEMA E DA SOLUÇÃO PROPOSTA	8
3	COLETA E PREPARAÇÃO DE DADOS	9
4	DESENVOLVIMENTO	12
5	DISCUSSÃO DOS RESULTADOS	13
5.0.0.1	Exemplo 1	13
5.0.0.2	Exemplo 2	15
5.0.0.3	Exemplo 3	17
5.0.0.4	Exemplo 4	19
6	CONCLUSÃO	22
	REFERÊNCIAS	23

1 Introdução

No final do ano de 2019 a China reportou um caso grave e desconhecido de pneumonia à Organização Mundial de Saúde (OMS). Pouco tempo depois, a primeira morte causada por esse vírus até então desconhecido foi também reportada pela China. O vírus teve então sua nomenclatura definida: Sars-CoV-2 (Covid-19).

Em questão de poucos dias, o vírus já tinha chegado a outros continentes, deixando o mundo e grandes nações em alerta. Várias medidas foram tomadas para tentar amenizar a disseminação do vírus, como a quarentena, uso de máscara, controle de fluxo e distanciamento social. Hoje, mais de 2 anos depois do início da pandemia, e com a vacinação em alta, ainda se faz necessário o uso de grande parte dessas medidas.

Pesquisadores e cientistas se uniram nesse combate ao Covid-19 em várias disciplinas, entre elas visão computacional. Essa área da inteligência artificial, trata como os computadores podem interpretar imagens digitais. A área de visão computacional cresceu bastante nos últimos anos, especialmente pelo crescente avanço do Aprendizado Profundo (Deep Learning)([ULHAQ et al., 2020](#)).

Hoje, mesmo com o estágio avançado de vacinação, o mundo pós COVID-19 ainda é incerto devido as variantes que vão surgindo. Com isso, medidas para prevenção e controle poderão ainda ser necessárias no futuro, como, o uso correto de máscaras. Visão computacional é uma excelente aliada para auxiliar na fiscalização dessas medidas, analisando imagens de câmeras e gerando alertas em tempo real.

1.0.1 Visão Computacional e Detecção de Objetos

Com o constante avanço na área computacional como capacidade de processamento e armazenamento de dados, ela vem sendo constantemente aplicada aos mais diversos tipos de problemas em análises de imagens. Técnicas de visão computacional têm mostrado enorme escopo em várias áreas de aplicação, especialmente na área da saúde e na pesquisa médica. Existe uma infinidade de novas abordagens de visão computacional em aplicativos de saúde que incluem, mas não se limitam a, diagnóstico de doenças, prognóstico, cirurgia, terapia e análise de imagens médicas ([GAO et al., 2018](#)). No combate ao Covid-19, visão computacional vêm sendo aplicada em 3 vertentes principais, sendo elas: diagnóstico, prevenção e controle, e gestão clínica e tratamento ([ULHAQ et al., 2020](#)).

Em um tempo não muito distante, apesar de já se falar em deep learning, problemas na área de visão computacional não eram nada triviais por diversos fatores: escassez de dados com suas classificações, falta de recursos computacionais, redes neurais complexas e densas o que fazia com que o tempo de treinamento fosse muito elevado. Com o passar dos

anos e com o desenvolvimento de redes profundas mais rápidas, recursos computacionais melhores e mais acessíveis bem como também a grande evolução em base de dados para problemas de visão computacional fez com que essa área desse um salto enorme e abrisse um leque de oportunidades.

Hoje plataformas oferecem recursos computacionais com excelente desempenho e com um preço acessível na nuvem. Cientista de dados que trabalham na área de visão computacional podem facilmente testar diferentes abordagens, treinando algumas redes de forma muita mais eficiente, diferente do passado onde realizar apenas um treinamento era uma tarefa que levava muito tempo.

Outro ponto muito importante, é a tecnologia móvel com câmeras de excelente qualidade enriquecendo cada vez mais a base de dados de fotos e vídeos do mundo. Como atualmente muitas redes possuem variações de redes mais densas a redes mais leves, dispositivos móveis possuem a capacidade de rodar aplicações de visão computacional em tempo real.

Detecção de objetos é uma das grandes áreas em visão computacional. Seu conceito básico consiste em analisar uma imagem e identificar onde objetos se localizam nessa imagem e qual o nome do objeto detectado. Existem diversos algoritmos para detecção de objetos em imagens, mas as mais utilizadas atualmente pertencem a família de redes YOLO (You Only Look Once).

2 Descrição do Problema e da Solução Proposta

O cenário atual de pandemia nos mostrou a importância do uso correto de máscaras para diminuir a disseminação do vírus, assim como o distanciamento social. Medidas essas que foram constantemente reafirmadas e recomendadas pela Organização Mundial de Saúde (OMS). Estabelecimentos tiveram que seguir essas normas para poderem estar abertos ao público. Tal trabalho de fiscalização é feito por cada estabelecimento de forma manual, ou seja, existem pessoas encarregadas de fazerem essa verificação e alertar um cliente caso ele não cumpra essas medidas.

Da mesma forma, quando falamos do mercado de trabalho, a pandemia promoveu uma evolução digital nunca vista antes, forçando empresas a aderirem ao método de trabalho remoto. Muitas dessas empresas inclusive adotou tal método permanentemente. Porém ainda existem trabalhos onde o trabalho presencial é essencial, e as empresas são responsáveis por garantirem a segurança de seus funcionários principalmente nesse momento adverso que o mundo enfrenta. Dessa forma, sistemas de segurança para garantir que medidas estão sendo seguidas são de extrema relevância.

O objetivo deste trabalho é a criação de um modelo para detectar pessoas que não estão usando máscaras, ou usando de forma incorreta. Para isso, foi utilizado o Face Mask dataset, treinado por duas versões do YOLO: o YOLOv5 ([JOCHER, 2021](#)) e o YOLOX ([GE et al., 2021](#)) para fins de comparação entre essas versões.

Com isso, sistemas podem ser criados utilizando tais modelos para monitoramento em tempo real a partir da análise de câmeras, podendo gerar alertas em um alto-falante, por exemplo.

3 Coleta e Preparação de Dados

O conjunto de imagens Face Mask, utilizado para o treinamento e validação do modelo desenvolvido nesse trabalho foi obtido no site Kaggle¹, que possui diversas bases de dados para os mais diversos fins de domínio público.

A base utilizada possui um total de 848 imagens com 3 classes distintas: com máscara, sem máscara e máscata usada de forma errada. Cada imagem também possui coordenadas das áreas de cada objeto de interesse presente na imagem em arquivos xml. Abaixo segue um exemplo de um arquivo xml e sua respectiva imagem.

```

<annotation>
    <folder>images</folder>
    <filename>maksssksksss0.png</filename>
    <size>
        <width>512</width>
        <height>366</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>without_mask</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>79</xmin>
            <ymin>105</ymin>
            <xmax>109</xmax>
            <ymax>142</ymax>
        </bndbox>
    </object>
    <object>
        <name>with_mask</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>

```

¹ <https://www.kaggle.com/andrewmvd/face-mask-detection>

```
<occluded>0</occluded>
<difficult>0</difficult>
<bndbox>
    <xmin>185</xmin>
    <ymin>100</ymin>
    <xmax>226</xmax>
    <ymax>144</ymax>
</bndbox>
</object>
<object>
    <name>without_mask</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <difficult>0</difficult>
    <bndbox>
        <xmin>325</xmin>
        <ymin>90</ymin>
        <xmax>360</xmax>
        <ymax>141</ymax>
    </bndbox>
</object>
</annotation>
}
```



Figura 1 – Exemplo de uma imagem utilizada no treinamento do modelo.

Como podemos observar na Figura 1, a imagem possui 3 objetos (pessoas) onde duas delas estão sem máscara e uma com máscara. É possível observar essas informações no arquivo XML citado anteriormente. Cada objeto possui a tag name que significa a qual classe tal objeto pertence, e as coordenadas de onde o objeto foi identificado.

Os dados estão no formato PASCAL VOC, utilizado pelo YOLOX. Já para o YOLOv5 os dados tiveram que estar no formato YOLOv5 Pytorch, portanto foi necessário converter os dados para esse formato. A distribuição das imagens está representada na Tabela 1.

Tabela 1 – Distribuição das imagens

Districuição dos Dados	Qtd. Imagens	(%)
Treinamento	594	70%
Validação	169	20%
Teste	85	10%
Total	848	100%

4 Desenvolvimento

Com o constante desenvolvimento em detecção de objetos, YOLO e suas versões sempre buscaram a melhor compensação entre acurácia e velocidade em aplicações em tempo real. YOLOv2 e YOLOv3 utilizaram as mais avançadas tecnologias de detecção disponíveis quando foram criados. Hoje, YOLOv5 possui a melhor compensação com acurácia de 48,2% nos dados da base COCO ([GE et al., 2021](#)).

Porém, com o tempo técnicas mais avançadas de otimização foram desenvolvidas e ainda não foram integradas com a família e modelos YOLO ainda. O principal objetivo do YOLOX é justamente integrar esses avanços mais recentes ao modelo.

Como base, YOLOX tem a implementação do YOLOv3 com Darknet53 com algumas mudanças em sua arquitetura como mostra a Figura 2. A principal mudança foi substituir uma "coupled head" por uma desacoplada, que contém uma camada de convolução 1x1, seguida de duas ramificações paralelas com 3 camadas de convolução 3x3 respectivamente. Essa mudança teve um ganho de 1.1 ms utilizando uma Tesla V100 ([GE et al., 2021](#)).

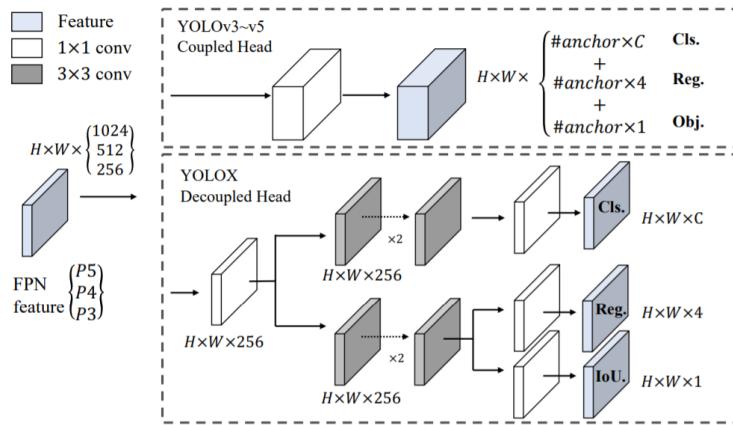


Figura 2 – Figura que mostra as diferenças entre YOLOX e YOLOv3-5 ([GE et al., 2021](#))

Tendo essas informações como base, foram criados 5 modelos do YOLOX variando do mais leve: YOLOX-s até o YOLOX- que é um modelo mais denso.

Para o desenvolvimento deste trabalho utilizamos o YOLOX-s para treinamento, e o YOLOv5-n para comparação. Em ambos os modelos, o treinamento foi feito por 100 épocas e batch size de 16. O tamanho das imagens foram de 640x640 para o YOLOX e de 416x416 para o YOLOv5. Utilizamos os pesos obtidos com o treinamento da base COCO como pesos iniciais.

5 Discussão dos Resultados

Para analisar o desempenho entre YOLOX e YOLOv5, foram utilizadas as métricas AP (average precision), map_5095 e map_50 para ambas as arquiteturas. As métricas map_5095 e map_50 são métricas padrão COCO e PASCAL VOC respectivamente que consistem em avaliar a média da precisão para limiares de IoU entre 0,5 e 0,95 no caso da métrica map_5095. Já a métrica map_50 é a média da precisão calculada com o valor IoU de 0,5.

IoU é a intersecção sobre união, que em detecção de objetos é um valor usado para medir a sobreposição das caixas de detecção de um objeto. Quanto mais próximos os valores previstos para uma caixa forem dos valores reais dessa mesma caixa, maior será a intersecção e maior será o valor de IoU. Dessa forma quanto maior forem os valores dessas duas métricas, melhor é o modelo.

Os resultados obtidos se encontram na Tabela 2. O YOLOX obteve uma AP total de 95,52% enquanto o YOLOv5 de 79,6%. É notável que o YOLOX obteve melhores resultados como um todo, e também analisando a AP de cada classe individualmente. O mesmo acontece quando comparamos as métricas map_5095 e map_50, YOLOX obteve um map_5095 de 29% maior quando comparado ao YOLOv5. Já para o map_50, essa diferença foi de 13%.

Esse resultados ficam evidentes nos exemplos de inferência abaixo (Exemplos de 1 a 4).

Tabela 2 – Métricas obtidas

Classes	X AP(%)	v5 AP(%)	Modelos	map_5095	map_50
with_mask	96,68	91,9	YOLOX	76,77%	95,51%
without_mask	99,50	84,4	YOLOv5	47,70%	82,11%
mask_weared_incorrect	90,37	62,5			
Total	95,52	79,6			

5.0.0.1 Exemplo 1

Neste exemplo, pode-se perceber que o YOLOX conseguiu detectar um número maior de objetos, principalmente objetos mais distantes da câmera.



Figura 3 – Imagem original (exemplo 1)

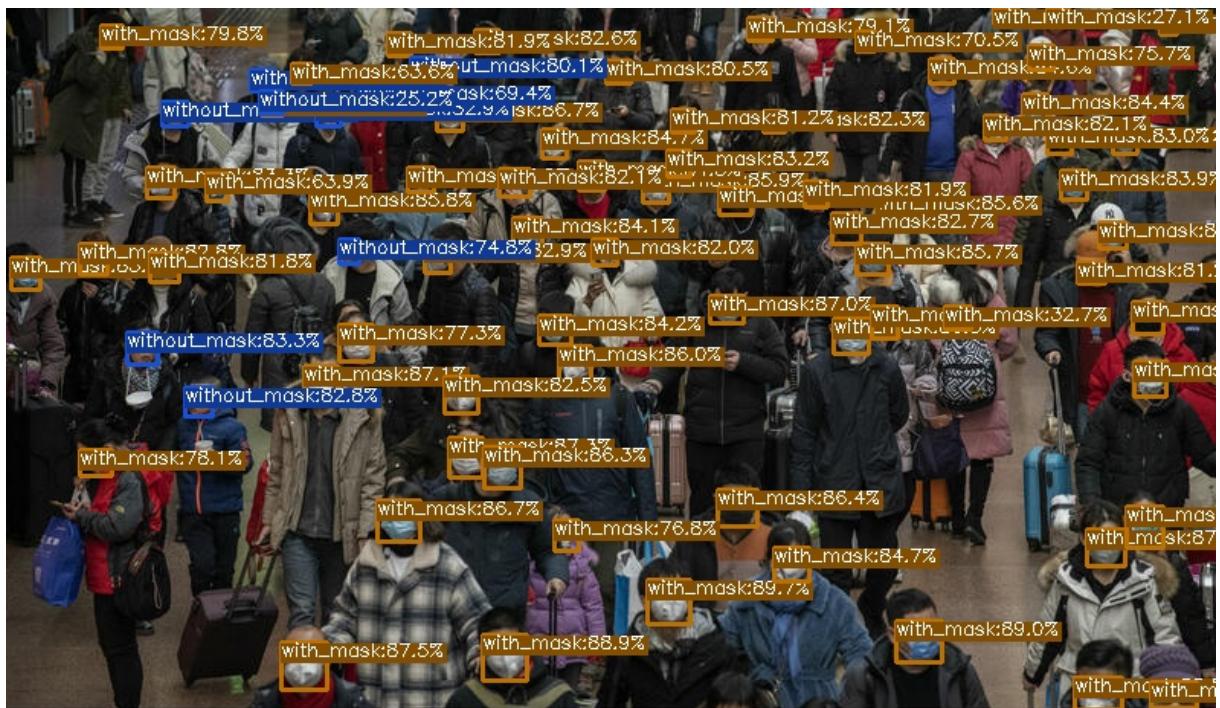


Figura 4 – Inferência com YOLOX (exemplo 1)



Figura 5 – Inferência com YOLOv5 (exemplo 1)

5.0.0.2 Exemplo 2

O mesmo acontece no exemplo 2, porém aqui YOLOX identificou um objeto como `mask_weared_incorrect`, quando deveria ser `with_mask`, que teve sua classificação correta pelo YOLOv5.



Figura 6 – Imagem original (exemplo 2)



Figura 7 – Inferência com YOLOX (exemplo 2)



Figura 8 – Inferência com YOLOv5 (exemplo 2)

5.0.0.3 Exemplo 3

Nesse exemplo com uma imagem muito mais complexa, podemos notar facilmente a diferença de quantidade de objetos detectados pelo YOLOX e pelo YOLOv5.



Figura 9 – Imagem original (exemplo 3)



Figura 10 – Inferência com YOLOX (exemplo 3)



Figura 11 – Inferência com YOLOv5 (exemplo 3)

5.0.0.4 Exemplo 4

Nesse caso, utilizando uma imagem com menos pessoas, ambos os modelos detectaram a mesma quantidade de objetos porém YOLOX fez uma classificação errada `without_mask` quando deveria ser `with_mask`. Nesse caso o YOLOv5 acertou todas as detecções.



Figura 12 – Imagem original (exemplo 4)

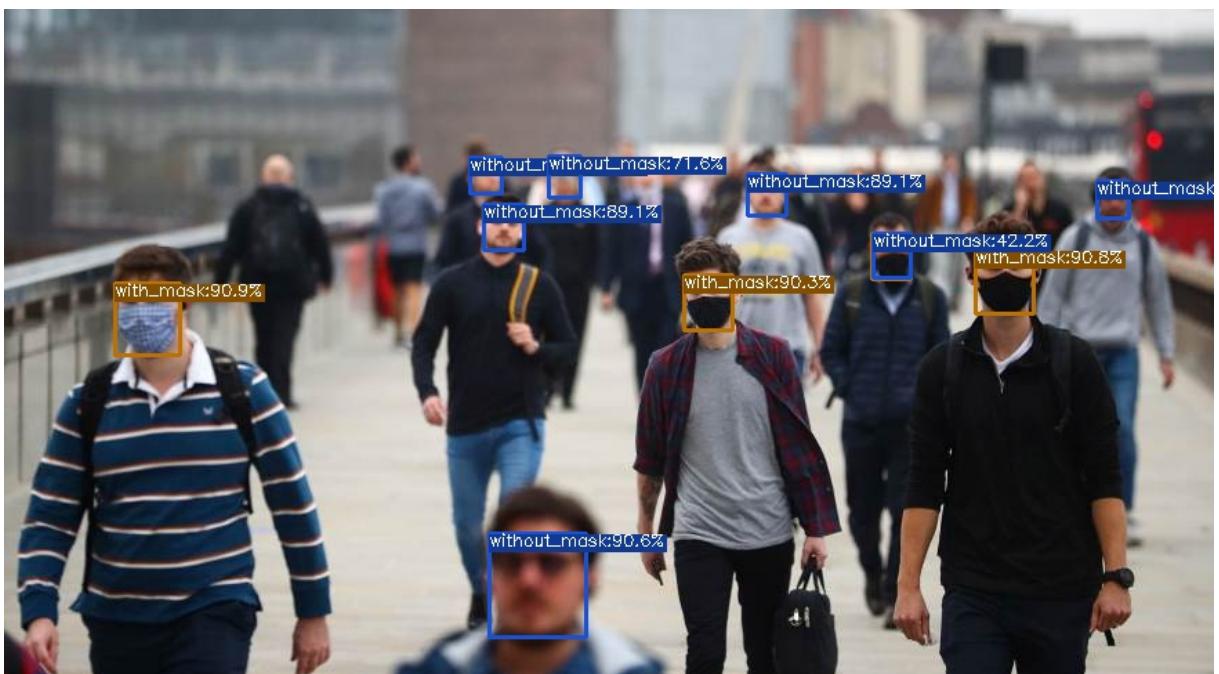


Figura 13 – Inferência com YOLOX (exemplo 4)



Figura 14 – Inferência com YOLOv5 (exemplo 4)

6 Conclusão

A partir dos resultados obtidos, o YOLOX obteve melhor desempenho com relação as métricas obtidas. Porém, duas observações bem importantes foram feitas durante a fase de treinamento e testes de performance. A primeira é que o YOLOv5 possui um tempo de treinamento duas vezes menor do que o YOLOX. Enquanto o YOLOv5 levou cerca de 1 hora para fazer o treinamento na base de dados utilizada nesse trabalho, o YOLOX já levou cerca de 2 horas para completar o treinamento. A segunda é o FPS, enquanto o YOLOv5 obteve um FPS entre 155-214, o YOLOX atingiu um FPS de 56-59.

Portanto, ambos os modelos são de extrema relevância na área de visão computacional e sua utilização ou não vai depender da aplicação e dos recursos disponíveis. Por exemplo, uma aplicação mais sensível, onde erros não podem ser frequentes, o uso de modelos mais robustos como o YOLOX é uma boa alternativa. O YOLOv5, por exemplo, seria uma boa opção para uma aplicação móvel.

Os resultados mostraram a capacidade da criação de um modelo para detectar o uso correto de máscara em lugares monitorados, auxiliando na fiscalização e colaborando assim para a diminuição da transmissão de COVID em ambientes controlados.

Referências

GAO, J. et al. *Computer vision in healthcare applications*. [S.l.]: Hindawi, 2018. Citado na página 6.

GE, Z. et al. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. Citado 3 vezes nas páginas 3, 8 e 12.

JOCHER, G. *YOLOv5*. [S.l.]: GitHub, 2021. <<https://github.com/ultralytics/yolov5>>. Citado na página 8.

ULHAQ, A. et al. Computer vision for covid-19 control: A survey. *arXiv preprint arXiv:2004.09420*, 2020. Citado na página 6.