Detecção de uso de máscaras em ambientes fechados com MobileNetV2 e Single Shot Multibox Detector

Paulo Fernando Neres Ferreira[1], **Daniel Lima Gomes Junior, Alex Martins Santos**

[1]Departamento de Computação
Instituto Federal de Educação, Ciência e Tecnologia do Maranhão (IFMA)
Campus Monte Castelo – São Luís, MA – Brazil

paulolf@acad.ifma.edu.br, {daniellima, alexmartins}@ifma.edu.br

Abstract. *This paper presents a deep learning application for detecting people wearing facial masks in indoor environments. With the advent of Covid 19 pandemic, the necessity of taking precautions to slow and control the pandemic growth is essential, between those precautions, wearing facial masks is vital to keep contagion rates as low as possible. The system is capable of detecting the correct use of facial masks in indoor environments using MobileNetV2 as feature extractor and SSD as the object detector, along with TensorFlow, Keras, OpenCV and Python technologies. The results show reasonable performance with low training time, hardware and data density.*

Resumo. *Este artigo apresenta uma aplicação de aprendizado profundo para detecção de pessoas utilizando máscaras faciais em ambientes fechados. Com o advento da pandemia da Covid 19, a necessidade de tomar precauções para diminuir e controlar o avanço da pandemia é essencial. Entre essas precauções, o uso de máscaras tornou-se importante para reduzir as taxas de contágio. O sistema apresentado é capaz de detectar o uso correto de máscaras em ambientes fechados usando mobilenetV2 como extrator de características e SSD como detector de objetos, junto de tecnologias como TensorFlow, Keras, OpenCV e Python. Os resultados mostram performances razoáveis utilizando curto período de treinamento, baixo poder de computação e densidade de dados.*

1. Introdução

Ao longo da pandemia do novo coronavírus, diversas medidas de prevenção e contenção do vírus foram reforçadas, como o distanciamento social e a higiene, entre essas medidas está também o uso de máscaras de proteção, que podem reduzir a emissão e contato com gotículas que podem conter o vírus.

O uso de máscaras faz parte de um pacote de medidas de prevenção e controle para frear a propagação de determinadas doenças respiratórias virais, incluindo a COVID-19. As máscaras podem ser usadas para a proteção de pessoas saudáveis (quando em contato com alguém infectado) ou para controle da fonte (quando usadas por alguém infectado para prevenir transmissão subsequente). Sob recomendação da Organização Mundial da Saúde. [OMS 2020].

Muitos lugares de utilização pública passaram a indicar a utilização de máscaras como forma de proteção individual, até mesmo como requisito de acesso a esses ambientes. Em conjunto com a medição de temperatura corporal, surgiu a necessidade de empregar pessoal para fiscalizar e educar quanto à utilização correta do EPI.

A medição de temperatura corporal tem sido descontinuada, tendo em vista a baixa especificidade para determinação de possíveis contaminados, Entretanto a utilização da máscara tem sido mantida como recomendação básica de enfrentamento à pandemia.

Em espaços amplos, com grande fluxo de pessoas e também ambientes confinados como elevadores, tornam inviáveis manter colaboradores para a fiscalização e orientação sobre a utilização das máscaras, sendo de grande importancia a colaboração de sistemas automatizados para essa tarefa.

Motivado pela importância do uso de máscaras para minimizar as taxas de transmissão da COVID-19, em especial, dentro de ambientes fechados, este trabalho propõe abordar o desenvolvimento de um modelo de detecção de objetos levando em consideração a detecção de pessoas utilizando máscaras de proteção e o contexto de ambientes fechados, principalmente em salas de aula.

Este trabalho apresenta a aplicação de um sistema de detecção de objetos focado na detecção não só de indivíduos com e sem máscara, mas também pessoas usando máscara de forma incorreta.

O modelo utilizado é baseado em sistemas de detecção em um estágio, em que todo o processo é feito em uma etapa só, permitindo que o modelo seja treinado com apenas uma etapa de *backpropagation* por passo de treinamento. Modelos com essa característica são constituídos por um modelo de classificação, porém sem a última camada e geralmente pré-treinado, seguido de um conjunto de camadas específicas responsáveis pela detecção. A razão de utilizar um modelo de classificação pré treinado é aproveitar as camadas de convolução treinadas durante a classificação para melhorar a performance do modelo e reduzir o tempo de treinamento e a quantidade de dados utilizados necessários para convergir [Goodfellow et al. 2016].

Com base nos trabalhos descritos, o modelo desenvolvimento nesse trabalho se preocupa não apenas na detecção do uso e do não uso de máscaras, mas também do uso incorreto de máscaras de proteção. Outro ponto visado, é atingir esses objetivos utilizando condições de hardware e quantidade de dados disponíveis inferiores aos mostrados nos trabalhos relacionados.

2. Trabalhos relacionados

Durante a pandemia, foram apresentados diversos trabalhos relacionados à visão computacional com objetivo de estudar e enfrentar determinadas necessidades ocasionadas por ela. No problema de detecção de objetos, muitos trabalhos envolvem a tarefa de identificação de pessoas utilizando máscaras de proteção.

O trabalho desenvolvido por Islam et al. (2020) introduz uma abordagem para um sistema de detecção e alerta do uso de máscaras de proteção utilizando redes convolucionais. Nesse sistema é adotado uma arquitetura simples de rede convolucional com a saída sendo dois valores representando respectivamente as probabilidades das classes *with_mask* e *without_mask*. Ao passo que a utilização de redes convolucionais é sem dúvidas eficiente, o trabalho possui dois grandes problemas, a não utilização de um *backbone* (uma rede de classificação servindo como *feature extractor*), o que acarreta na necessidade de utilizar grandes volumes de dados e/ou um longo período de treinamento, e a detecção apenas de uma pessoa por imagem.[Islam et al. 2020]

Jiang e Fan (2020) propõem o que consideram a primeira arquitetura de alta performance para um detector de faces no contexto de uso de máscaras faciais. O trabalho apresentado descreve caracteríscas não vistas em trabalhos anteriores como distinguir máscaras usadas incorretamente e corretamente. A preocupação em distinguir diferentes estados para o uso de máscaras também é levado em consideração por este trabalho.[Jiang and Fan 2020]

Balaji et al. (2021) apresentam um modelo baseado em dispositivos Raspberry Pi utilizando a linguagem de programação Python em conjunto de frameworks como TensorFlow e OpenCV para executar um modelo de detecção de objetos que detecta dois tipos de objetos, pessoas utilizando e não utilizando máscaras de proteção. O objetivo é utilizar o dispositivos para auxilar o governo e público a monitorar o uso de máscaras e cumprimento de outras regras como distânciamento em locais públicos. [Balaji et al. 2021]

Este trabalho utiliza técnicas de transferência de aprendizado e ajuste fino para adaptar um modelo de classicação pré treinado como extrator de caracteríscas, como base para o modelo de detecção SSD (*Single Shot Multibox Detector*), com objetivo de detectar não só o uso ou ausência de máscaras de proteção, mas também se a utilização de máscara é feita de forma correta. A utilização de um extrator de características pré treinado permite treinar um modelo robusto de forma rápida e utilizando poucos recursos como poder computacional e baixo volume de dados.

3. Metodologia

Neste trabalho é utilizado a técnica de detecção em um estágio de combinação de modelos de classificação pré-treinados e modelos de detecção baseado em redes neurais convolucionais. É utilizado o modelo de classificação MobileNetV2 para extração de características, a arquitetura do mobilenetv2 pode ser vista na Tabela 1, cada linha representa uma camada ou bloco de camadas do modelo, as duas últimas camadas ilustradas são descartadas para o problema de detecção. [Sandler et al. 2018]

O modelo pré-treinado com o dataset ImageNet (um dataset para classificação de imagens com mais de 1 milhão de exemplos divididos em um total de 1000 classes) teve suas últimas camadas descartadas e o restante do modelo usado para transferência de aprendizado para o problema de detecção de máscaras. [Deng et al. 2009]

O modelo de detecção utilizado é o Single Shot Multibox Detector (SSD), baseado no trabalho feito por Liu et al. (2016), originalmente utilizando a arquitetura de classificação VGG16 Simonyan e Zisserman (2014), como ilustrada na Figura 2. O SSD utiliza diferentes camadas do extrator de características combinado a novas camadas para gerar detecções otimizadas em detectar objetos de diferentes escalas, aumentando a eficiência das detecções, principalmente considerando objetos com escalas pequenas na imagem. [Liu et al. 2016] [Simonyan and Zisserman 2014]

O modelo de detecção foi pré treinado utilizando o COCO Dataset [Lin et al. 2014] e utilizado para ajuste fino com o dataset Face Mask Dataset [Mask_Dataset 2020]. No momento de desenvolvimento do projeto, o dataset contava com cerca de 853 imagens anotadas com 3 categorias de objetos no formato PASCAL VOC [Everingham et al. 2010], faces com máscaras, sem máscaras e utilizando máscaras incorretamente. [Lin et al. 2014] [Mask_Dataset 2020] [Everingham et al. 2010]

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

Figura 1. Arquitetura MobilenetV2. [Sandler et al. 2018]

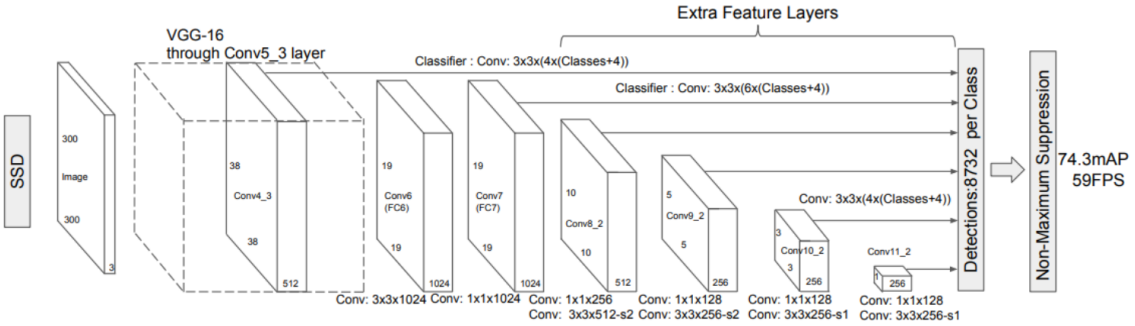

Figura 2. Arquitetura SSD original. (Adaptado. Liu *et al.*, 2016)

Os dados de entrada seguiram a normalização utilizada nos modelos de classificação MobileNet, com valores de pixels variando entre (-1, 1). Todas as técnicas de pré processamento empregadas podem ser vistas na Tabela 1 abaixo.

A taxa de divisão da base de dados foi de 80/20, 80% dos dados reservados para o treinamento e 20% para testes. Já o ambiente de treinamento é composto por:

a) Framework Tensorflow 2.7 com suporte à GPU;
b) Python 3.8;
c) Windows 10 64 bits;
d) Processador Intel Core i7 4GHz;
e) Memória RAM total de 16GB;
f) GPU NVidia GTX 1050 ti 4GB VRAM;
g) Base de dados pré-processada e convertida para formato binário *tf record*.

O treinamento foi realizado com imagens redimensionadas para um resolução fixa de 320x320 pixels, essa resolução permite realizar o treinamento com *batch size* de 16 utilizando os recursos de hardware mecionados anteriormente, aproveitando o máximo de recursos disponíveis com a maior resolução e *batch size* possíveis.

A taxa de aprendizado inicial escolhida foi de 0,007, considerando trabalhos como [Li et al. 2018] que uma taxa de 0,1 com decaimento exponencial e [Zaki et al. 2020] com

Tabela 1. Técnicas de pré processamento empregadas. As técnicas de augmentação e normalização são realizadas em tempo de execução.

Normalização	Valores dos pixels normalizados para a faixa (-1, 1) em float32.
Random crop	Cada imagem cortada aleatoriamente para escala de até 90%.
Random Translate	Cada imagem é descolada aleatoriamente na vertical e horizontal em fator variando de -25% e +25%.
Random Horizontal Flip	Cada imagem é invertida ou não (aplicada aleatoriamente com uma taxa de 50%) de forma horizontal.

variações de 0,1 a 0,001. É empregando o algoritmo de decaimento de cosseno na taxa de aprendizado, com 40 mil passos para decaimento.

A duração total do treinamento foi de aproximadamente quatro horas com *batch size* de tamanho 16 (16 exemplos por passo de treinamento) e o otimizador utilizado foi o *stochastic gradient descent* (SGD).

4. Resultados

Tendo em vista as condições de treinamento impostas devido a limitação de tempo e recursos, foi possível atingir resultados satisfatórios, tanto em velocidade de processamento utilizando baixo poder computacional, quanto em questão de visualição das detecções.

A performance do modelo foi testada em diferentes ambientes e situações, primeiramente avaliando a latência do modelo em detecções utilizando 1 amostra, com uso de GPU e CPU, com visualização e sem visualização das imagens. Também foi avaliado seu uso aplicado ao processamento em tempo real, considerando os mesmos cenários citados anteriormente.

As imagens capturadas para inferência, que possuem uma resolução de 1920x1080 pixels, são redimensionadas para resolução 640x640 pixels. Originalmente, as imagens estão em formato de cores BGR (com o canal de cores sendo respectivamente azul, verde e vermelho), necessitando assim, serem convertidas para o formato RGB, em seguida normalizadas para seus valores de pixels variarem de -1 a 1 e, só então, estão aptas para serem processadas pelo modelo de detecção. A Tabela 2 mostra os resultados obtidos nas resoluções 640x640 e 320x320 utilizando CPU e GPU.

Os resultados de avaliações das métricas de precisão, recall e *mean average precision* (uma métrica que calcula a média da precisão em cada classe de objeto) são mostrados na Tabela 3. A Tabela mostra o ganho nas métricas, principalmente em *mAP*, quando a resolução das imagens avaliadas aumentam. Considerando a aplicação do modelo em ambientes fechados, em especial, salas de aula, as métricas usadas com objetos largos (com dimensões mínimas de 96x96 pixels) e resolução de 640x640 pixels mostra-se a mais recomendada para avaliar o modelo nesse cenário de ambientes fechados.

Tabela 2. Tempo de resposta do modelo utilizando diferentes resoluções e dispositivos.

Cenários		Tempo de processamento (ms)	Com visualização de imagens (ms)
640x640	CPU	219	227
	GPU	46	50
320x320	CPU	63	71
	GPU	22	26

Tabela 3. Métricas do modelo em diferentes resoluções.

Resolução	Métricas (%)		
	Average Precision (AP)	Average Recall (AR)	Mean Average Precision (mAP)
320x320	78,1	80,2	44,9
640x640	78,7	70,3	54,1
640x640 objetos largos (96x96)	83,3	85,4	83,3

A Figura 3 apresenta o resultado de inferência utilizando uma sala de aula como ambiente. Houve a preocupação de analisar o tamanho mínimo dos objetos considerando as dimensões da sala, confirmando o motivação de utilizar as métricas mencionadas anteriormente, o tamanho dos objetos variam de 60x125 a 140x178 pixels (considerando as distâncias mínimas e máximas possíveis dos objetos). As imagens foram capturadas em 1920x1080 pixels de resolução para visualização e redimensionadas para 640x640 para inferência. O dispositivo utilizado para inferência foi um Huawei IdeaHub S, com sistema operacional Windows 10 x64 integrado e com as seguintes especificações:

a) Processador Intel Core i5-8500 @ 3.0GHz;
b) Memória RAM 8GB.

5. Conclusão

Este trabalho apresenta a aplicação um modelo de detecção de objetos para detecção de pessoas no contexto do uso de máscaras de proteção em três categorias diferentes: com máscara, sem máscara e vestindo máscara incorretamente. É utilizado um modelo de detecção SSD com a arquitera MobilenetV2 como extrator de características pré treinados nos datasets ImageNet e COCO.

Os resultados apresentados em tempo real são razoáveis, considerando o contexto de ambientes fechados e a análise das imagens de inferência a serem utilizadas nesses ambientes, obtendo uma precisão de 83,3% e *mAP* de 54,1, realizando inferência com

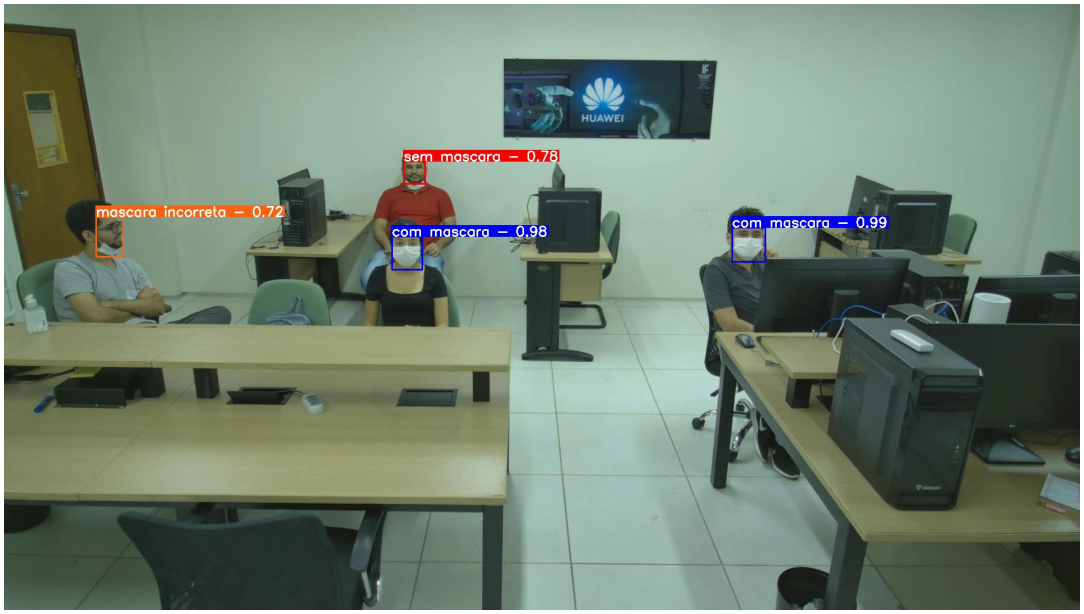

Figura 3. Exemplo de classificação realizada em sala de aula.

imagens na resolução 640x640 pixel para ganho em precisão ao passo que mantém uma performance em tempo real de 4,5 quadros por segundos utilizando CPU e 19,6 utilizando GPU.

Os testes foram realizados em uma sala de aula utilizando o Huawei IdeaHub para captura, visualização e processamento das imagens. Durante a verificação em tempo real, não foi possível notar erros grosseiros de detecção do modelo, pois as condições de iluminação normais para o cenário considerado, proporcionam um ambiente favorável para a detecção de objetos.

O sistema apresentado pode ser facilmente aprimorado, utilizando treinamentos mais longos, e com um maior volume de dados. Como pontos de melhoria podem ser destacados, testes utilizando outros tipos de ambientes, utilizado como base para ajuste fino em outros trabalhos e otimização utilizando técnicas de quantização.

Referências

Balaji, S., Balamurugan, B., Kumar, T. A., Rajmohan, R., e Kumar, P. P. (2021). A brief survey on ai based face mask detection system for public places (march 28, 2021). *Irish Interdisciplinary Journal of Science & Research*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., e Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., e Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Goodfellow, I., Bengio, Y., e Courville, A. (2016). *Deep Learning*. MIT Press. Disponível em: <http://www.deeplearningbook.org>. Acecsso em: Jun, 2022.

Islam, M. S., Haque Moon, E., Shaikat, M. A., e Jahangir Alam, M. (2020). A novel approach to detect face mask using cnn. In *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, pages 800–806.

Jiang, M. e Fan, X. (2020). Retinamask: A face mask detector. *CoRR*, abs/2005.03950.

Li, Y., Huang, H., Xie, Q., Yao, L., e Chen, Q. (2018). Research on a surface defect detection algorithm based on mobilenet-ssd. *Applied Sciences*, 8(9).

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., e Dollár, P. (2014). Microsoft coco: Common objects in context.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., e Berg, A. C. (2016). SSD: Single shot MultiBox detector. In *Computer Vision – ECCV 2016*, pages 21–37. Springer International Publishing.

Mask_Dataset (2020). Mask dataset. `https://makeml.app/datasets/mask`.

OMS (2020). Recomendações sobre o uso de máscaras no contexto da covid-19. Disponível em: <`https://apps.who.int/iris/bitstream/handle/10665/332293/WHO-2019-nCov-IPC_Masks-2020.4-por.pdf`>. Acesso em: Ago, 2021.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., e Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520.

Simonyan, K. e Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition.

Zaki, S. Z. M., Asyraf Zulkifley, M., Mohd Stofa, M., Kamari, N. A. M., e Ayuni Mohamed, N. (2020). Classification of tomato leaf diseases using MobileNet v2. *IAES Int. J. Artif. Intell. (IJ-AI)*, 9(2):290.