Face mask detection using PyTorch | Detecto

Nistor Robert\*

Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

\*Robert, V.S, Robert; [nistorcristian15@outlook.com](mailto:nistorcristian15@outlook.com)

**Abstract**

A Computer Vision application for detecting if a person is wearing a face mask using Pytorch and Detecto.

**Keywords:** PyTorch; face mask detection; object detection; Detecto

1 Context and motivation

* With the COVID-19 pandemic, face masks have become an important tool for preventing the spread of the virus. A face mask detection algorithm could be used to help enforce mask-wearing rules in public spaces, and thus help to keep people safe.
* A face mask detection algorithm could be used to automate the process of checking for mask-wearing in places where it is required, such as in stores, offices, or public transportation. This could save time and effort for both employees and customers.

2 PyTorch

**PyTorch** object detection is the process of detecting objects within an image or video by using deep learning models. PyTorch provides several pre-trained models and libraries for object detection, such as torchvision, which makes it easy to train and deploy object detection models.

2.1 Detecto

**Detecto** is a PyTorch-based package for object detection, providing easy training and evaluation of popular architectures like Faster R-CNN, Mask R-CNN, and RetinaNet. It has a simple interface for loading datasets, applying data augmentation, and visualizing results. It also supports multiple GPUs and transfer learning. It is easy to use and can be integrated with other libraries, making it a useful tool for quickly building object detection models.

2.2 Classes and methods

2.2.1 Dataloader

**detecto.core.DataLoader** is used for loading image data for object detection tasks. It is used to load a dataset of images and their corresponding object bounding boxes from a directory. It provides a convenient way to iterate through the dataset, loading images and bounding boxes as batches for training or testing. The class also provides methods for preprocessing the data, such as resizing or flipping images, as well as methods for splitting the data into train and test sets.

It works by loading image data and their corresponding object bounding boxes from a directory, and providing methods for preprocessing the data and splitting it into train and test sets.

2.2.2 Model

**detecto.core.Model** is a class in the detecto library that provides a simple and easy-to-use interface for creating and training object detection models. It wraps around PyTorch's built-in object detection models, such as Faster R-CNN and RetinaNet, and provides a consistent and simplified API for training, evaluating, and deploying object detection models.

When an instance of the class is created, it takes a PyTorch object detection model as input, such as Faster R-CNN or RetinaNet, and sets it as the backbone for the class. The user can then train the model by passing in a detecto.core.DataLoader object, which contains the image and annotation data, and the desired number of training epochs.

During training, the class uses the data from the detecto.core.DataLoader object to update the weights of the object detection model, in order to minimize the loss function. The loss function is defined as the difference between the predicted bounding boxes and the ground-truth bounding boxes in the dataset. The user can choose different loss functions depending on the use case.

2.2.3 Faster R-CNN ResNet-50 FPN (Faster R-CNN with ResNet-50 and Feature Pyramid Network)

**Faster R-CNN ResNet-50 FPN (Faster R-CNN with ResNet-50 and Feature Pyramid Network**) is a state-of-the-art object detection model developed by Facebook AI Research. It is built on top of the popular Faster R-CNN architecture, which uses a region proposal network (RPN) to generate a set of region proposals, and then applies a fully convolutional neural network (CNN) to classify and refine the object bounding boxes.

The ResNet-50 component of the model refers to the use of the ResNet-50 architecture as the base network for the CNN. ResNet-50 is a deep convolutional neural network that is trained on the ImageNet dataset and is known for its good performance on image classification tasks.

The FPN (Feature Pyramid Network) component of the model is a feature enhancement technique that is added on top of the base network. It aims to improve the feature representation of the image at multiple scales by creating a pyramid of features, where each level of the pyramid corresponds to a different scale of the image. This allows the model to detect objects at various scales and improves its performance on small objects.

3 State of the Art

3.1 Papers

3.1.1 [Detecting Face Masks Using Transfer Learning and PyTorch](https://www.analyticsvidhya.com/blog/2021/07/detecting-face-masks-using-transfer-learning-and-pytorch/)

The article uses transfer learning and PyTorch to build the model. The article describes how to fine-tune a pre-trained model, such as a ResNet-50, on a dataset of images of people wearing and not wearing face masks. The article also describes how to evaluate the model's performance and how to use the model to make predictions on new images.

3.1.2 [FASTERRCNN\_RESNET50\_FPN Documentation](https://pytorch.org/vision/main/models/generated/torchvision.models.detection.fasterrcnn_resnet50_fpn.html)

This is the official documentation about the Faster R-CNN ResNet-50 FPN.

3.1.3 [How I built a Face Mask Detector for COVID-19 using PyTorch Lightning (updated PL V.1.3.5)](https://towardsdatascience.com/how-i-built-a-face-mask-detector-for-covid-19-using-pytorch-lightning-67eb3752fd61)

This article describes the process of building a face mask detector using the PyTorch Lightning library (updated to version 1.3.5).

3.2 Application

3.2.1 [Face Mask Detection](https://play.google.com/store/apps/details?id=com.mmm.facemaskdetection&hl=en&gl=US)

Face Mask Detection is a free android app that takes input from front and back camera and make detection on the presence of a face mask.

4 Conclusion

In conclusion, a face mask detection algorithm using the detecto library can be a powerful tool for detecting face masks in real-time. The detecto library provides a simple and easy-to-use interface for creating and training object detection models, and it can be used in combination with PyTorch's built-in object detection models, such as Faster R-CNN and RetinaNet.

Acknowledgements

* Lect. PhD. Diana-Laura Borza [ dianaborza@cs.ubbcluj.com ]