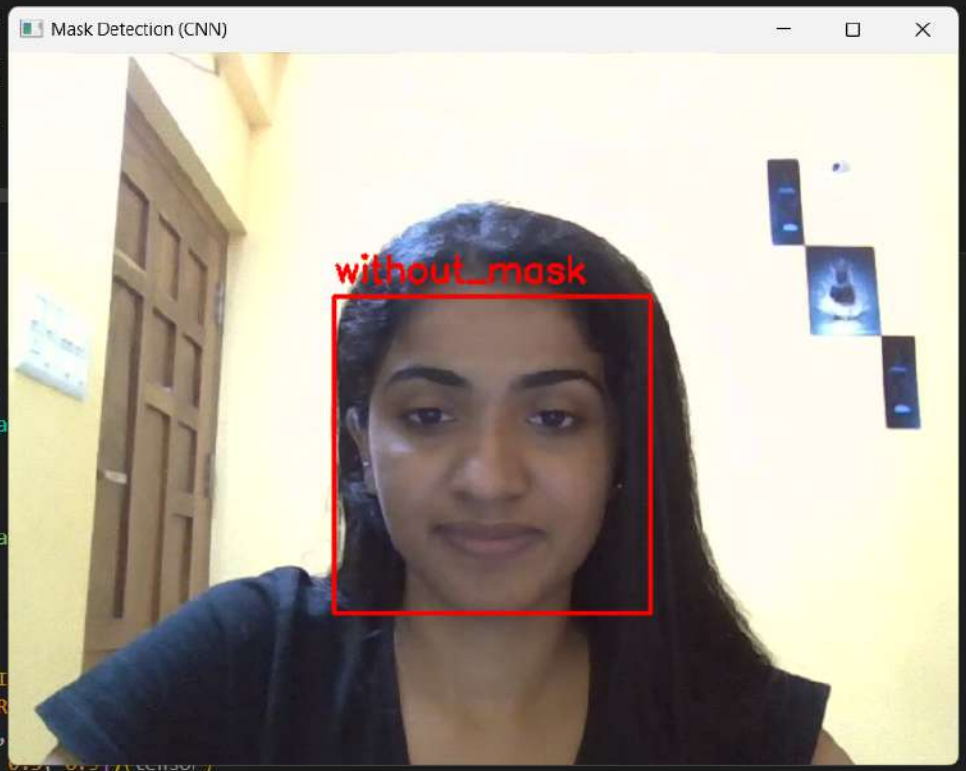
File Edit Selection View Go Run Terminal Help ICT

mask_detection.ipynb ●

mask_detection.ipynb > from tqdm import tqdm

Generate + Code + Markdown | Interrupt Restart Clear All Outputs Go To | Jupyter Variables Outline ··· Python 3.13.5

```
Epoch 12: Loss=0.0335, Val Acc=97.13%
Model saved
Epoch 13/15: 100%|██████████| 96/96 [00:49<00:00,  1.94it/s]
Epoch 13: Loss=0.0275, Val Acc=96.87%
Epoch 14/15: 100%|██████████| 96/96 [00:47<00:00,  2.04it/s]
Epoch 14: Loss=0.0266, Val Acc=94.52%
Epoch 15/15: 100%|██████████| 96/96 [00:48<00:00,  2.00it/s]
Epoch 15: Loss=0.0178, Val Acc=97.00%
```



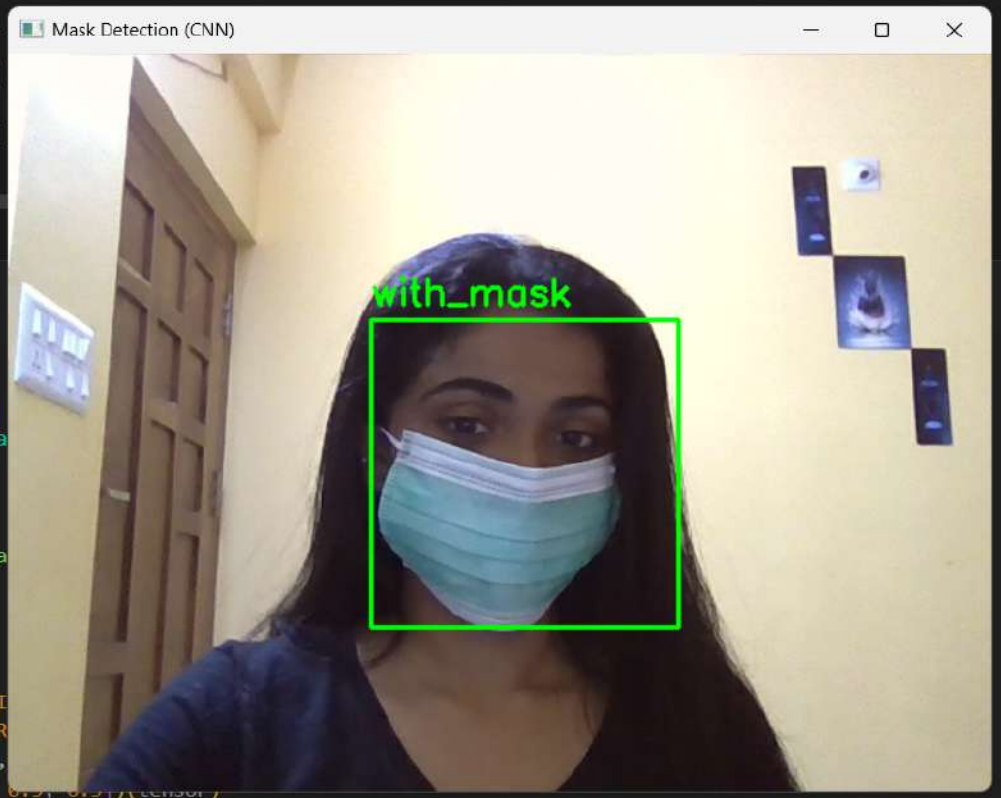

```
import cv2
import numpy as np

# Load CNN model
model = MaskCNN(num_classes=len(class_names)).to(device)
model.load_state_dict(torch.load("mask_cnn.pth", map_loca
model.eval()

face_cascade = cv2.CascadeClassifier(
    cv2.data.haarcascades + "haarcascade_frontalface_defa
)


def preprocess_face(face_bgr):
    face_resized = cv2.resize(face_bgr, (IMG_SIZE, IMG_SI
    face_rgb = cv2.cvtColor(face_resized, cv2.COLOR_BGR2R
    tensor = torch.tensor(face_rgb / 255.0).permute(2, 0,
    tensor = transforms.Normalize([0.5, 0.5, 0.5], [0.5,
    return tensor.unsqueeze(0).to(device)


cap = cv2.VideoCapture(0)
```

mask_detection.ipynb ●

mask_detection.ipynb > from tqdm import tqdm

Generate | Code | Markdown | Interrupt | Restart | Clear All Outputs | Go To | Jupyter Variables | Outline | ... | Python 3.13.5

```
Epoch 12: Loss=0.0335, Val Acc=97.13%
Model saved
Epoch 13/15: 100%|██████████| 96/96 [00:49<00:00,  1.94it/s]
Epoch 13: Loss=0.0275, Val Acc=96.87%
Epoch 14/15: 100%|██████████| 96/96 [00:47<00:00,  2.04it/s]
Epoch 14: Loss=0.0266, Val Acc=94.52%
Epoch 15/15: 100%|██████████| 96/96 [00:48<00:00,  2.00it/s]
Epoch 15: Loss=0.0178, Val Acc=97.00%
```




```
import cv2
import numpy as np

# Load CNN model
model = MaskCNN(num_classes=len(class_names)).to(device)
model.load_state_dict(torch.load("mask_cnn.pth", map_loca
model.eval()

face_cascade = cv2.CascadeClassifier(
    cv2.data.haarcascades + "haarcascade_frontalface_defa
)


def preprocess_face(face_bgr):
    face_resized = cv2.resize(face_bgr, (IMG_SIZE, IMG_SI
    face_rgb = cv2.cvtColor(face_resized, cv2.COLOR_BGR2R
    tensor = torch.tensor(face_rgb / 255.0).permute(2, 0,
    tensor = transforms.Normalize([0.5, 0.5, 0.5], [0.5,
    return tensor.unsqueeze(0).to(device)


cap = cv2.VideoCapture(0)

while True:
```

⚠ 3 | Spaces: 4 | {} | Cell 4 of 5 | Prettier