

Real-Time Machine Learning

...

Facial and Mask Recognition in Real-Time



The Team



Nelusha Dias

Rutgers School of Engineering

ECE 2024

David Banyamin

Rutgers School of Engineering

ECE 2023

Overview

- Use Machine Learning for Real Time Tasks
 - Facial Recognition
 - ID Face in frame
 - ID Mask
 - Applications
 - Building security camera
 - Photo ID



Timeline

- Learning
 - ML
 - Python
 - Pytorch
 - Hardware
- Algorithms
 - Facial Recognition
 - Put names to faces
 - Saving unknown & masked faces
 - Check for mask using CNN



Program Overview

- Combined useful aspects of two codes
 - Doorcam
 - Facial Recognition
- Known Face Identification
- Saves Unknown Faces
- Mask Identification

```
39 # Main Loop
40 def main_loop():
41     unknown_number = 1
42     mask_number = 1
43     while True:
44         # Grab a single frame of video
45         ret, frame = video_capture.read()
46
47         # Resize frame of video to 1/4 size for faster face recognition processing
48         small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
49
50         # Convert the image from BGR color (which OpenCV uses) to RGB color (which
51         # face_recognition uses)
52         rgb_small_frame = small_frame[:, :, ::-1]
53
54         # Find all the face locations and face encodings in the current frame of the video
55         face_locations = face_recognition.face_locations(rgb_small_frame)
56         face_encodings = face_recognition.face_encodings(rgb_small_frame, face_locations)
57
58         # Loop through each detected face and see if it is one we have seen before
59         # If so, we'll give it a label that we'll draw on top of the video
60         if process_this_frame:
61             mask = False
62
63         # Find all the faces and face encodings in the current frame of video
64         face_locations = face_recognition.face_locations(rgb_small_frame)
65         face_encodings = face_recognition.face_encodings(rgb_small_frame,
66         face_locations)
67         face_names = []
68         for face_encoding in face_encodings:
69
70             # See if the face is a match for the known face(s)
71             matches = face_recognition.compare_faces(known_face_encodings,
72             face_encoding)
73             name = "Unknown"
74
75
76             # If matches use the known face with the smallest distance to the
77             # new face
78             face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
79             best_match_index = np.argmin(face_distances)
80             if matches[best_match_index] and mask == False:
81                 name = known_face_names[best_match_index]
82
83             if name == "Unknown" and mask == False:
84                 image = frame
85                 unknown_number_string = str(unknown_number)
86                 image_name = "unknown" + unknown_number_string + ".jpg"
```

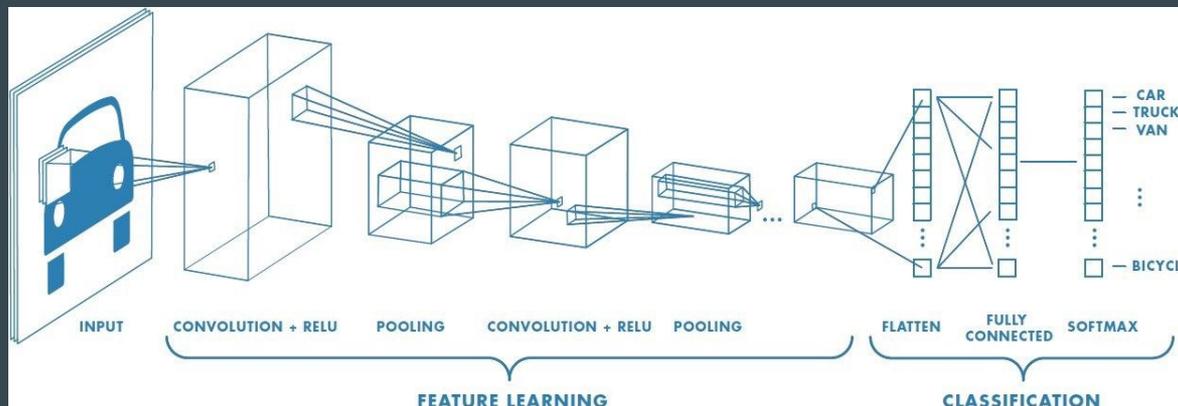
Convolutional Neural Network (CNN)

General Knowledge

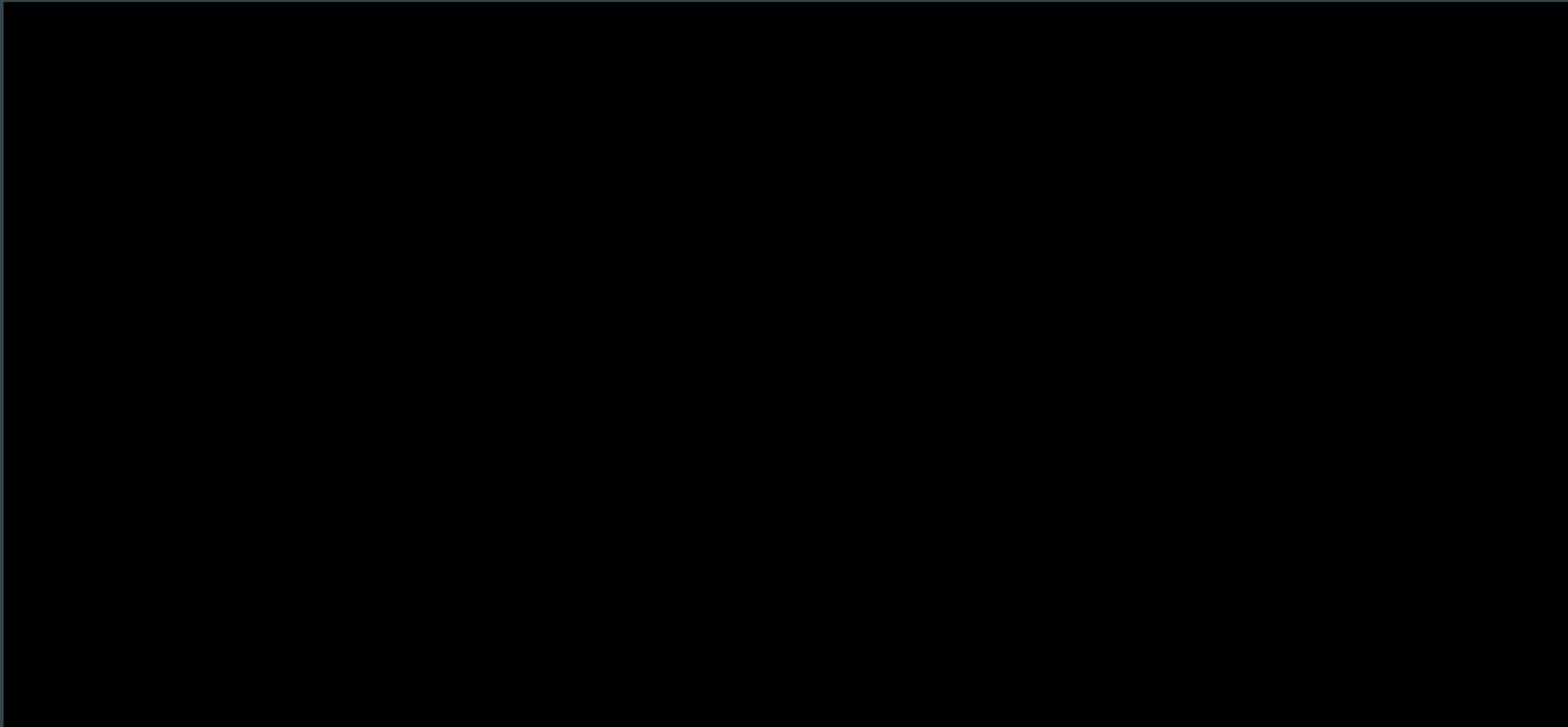
- Algorithm takes input image and identifies what it is
- Image goes through layers which apply filters onto the image

Our Program

- Two 2D convolutional layers
- Two pooling layers
- Three fully-connected layers
- Learning rate 0.001
- 4 epochs and batch size of 4



Video Demonstration



Website

The screenshot shows a Wix website with a navigation bar at the top containing 'WIX', 'Upgrade your website to remove Wix ads', and 'Upgrade Now'. The main content area is titled 'REAL TIME FACIAL RECOGNITION' and features a section 'THE PROJECT' with two paragraphs of text. To the right of the text is an image of a computer circuit board. Below the text is a button labeled 'The Code'. The website also includes a footer with navigation links: 'Home', 'Project', 'How The Code Works', and 'The Team'.

Poster

The poster is titled 'Real-Time Machine Learning' and is attributed to Nelusha Dias and David Banyamin. It is divided into several sections: 'Summary', 'Mask Recognition', 'Facial Recognition', 'Future Work', and 'References'. The 'Summary' section lists the project's goals and potential uses. 'Mask Recognition' describes the CNN model used. 'Facial Recognition' details the scanning process and includes a flowchart. 'Future Work' outlines improvements and adjustments. 'References' lists academic and technical sources. The poster also features a small image of a circuit board and a code snippet.

Summary

- Worked with algorithms and a Convolutional Neural Network to run a facial recognition program on NVIDIA Jetson Nano to identify known, unknown, and masked individuals
- Facial Recognition Program
 - Able to tell who is in the frame
 - If unknown save for later identification
 - Trained to recognize masks and person under it
- Potential Use
 - ID people
 - Building Security System
 - Checking for masks

Mask Recognition

- Used a Convolutional Neural Network (CNN) as a model to identify whether a person is masked or not
- Returns True if masked, False if not
- If the code returns True
 - Real-time video display shows a list of top 10 people that the individual may be
 - Video frame is captured and saved into a file
- If the code returns False
 - Run through the regular facial recognition

Facial Recognition

- Scan every frame for faces
- Find face landmarks on the detected face and create an encoding
- Iterate through the saved known faces to match name to face
- If person is unknown:
 - Save image of person & count number of visits
- If a mask is detected:
 - Saves the image with a list of possible people

Future Work

- Improvement on the Convolutional Neural Network for higher accuracy
- Adjust numerical parameters
- Add and remove layers
- Change the loss function
- Adjust ability to read a masked face as a face
- If there is a larger mask (not tight fitting) the code currently does not identify it as a face
- Ability to read multiple masked faces
- The program is only able to recognize and register one out of all the people in the frame when someone is wearing a mask

References

Chelvey, A. (2020, October 05). Build a face recognition system for \$60 with the new Nvidia Jetson Nano SGB and Python. Medium. <https://medium.com/@agstjey/build-a-face-recognition-system-for-60-with-the-new-nvidia-jetson-nano-sgb-and-python-18e4b0d52764>

Face-recognition (2020, February 20). PyPi. <https://pypi.org/project/face-recognition/>

Calabrese, M. (2020, August 28). MaskDetection is a dataset of human faces with a correctly and incorrectly worn mask based on the dataset flickr-faces-HQ. PFFHQ. (2021, April 28). GitHub. <https://github.com/sabam1/MaskedFaceNet>

Pfeiffer, A. (2020, August 28). Implementing CNNs in PyTorch with custom dataset and transfer learning. Medium. <https://medium.com/@arad/face-ai-how-to-implementing-cnn-in-pytorch-with-custom-dataset-and-transfer-learning-19652ee111ec>

Training a classifier — PyTorch tutorials 1.9.0-cv1.02 documentation. (2021). PyTorch. https://pytorch.org/tutorials/beginner/tutorials10_tutorial.html

WINLAB

<https://realtimeml.wixsite.com/facial-recognition>

Conclusion & Future

Results:

- Got a functioning program
 - Identify the person in front of the camera
 - Saves unknown and masked individuals

```
[root@node21-32:~/proj# python3 cnn.py
Epoch [1/4], Step [2000/23223], Loss: 0.0003
Epoch [1/4], Step [4000/23223], Loss: 0.0003
Epoch [1/4], Step [6000/23223], Loss: 0.0000
Epoch [1/4], Step [8000/23223], Loss: 0.0002
Epoch [1/4], Step [10000/23223], Loss: 0.0032
Epoch [1/4], Step [12000/23223], Loss: 0.0012
Epoch [1/4], Step [14000/23223], Loss: 0.0000
Epoch [1/4], Step [16000/23223], Loss: 0.0014
Epoch [1/4], Step [18000/23223], Loss: 0.0001
Epoch [1/4], Step [20000/23223], Loss: 0.0002
Epoch [1/4], Step [22000/23223], Loss: 0.0001
Epoch [2/4], Step [2000/23223], Loss: 0.0000
Epoch [2/4], Step [4000/23223], Loss: 0.0006
Epoch [2/4], Step [6000/23223], Loss: 0.0002
Epoch [2/4], Step [8000/23223], Loss: 0.0007
Epoch [2/4], Step [10000/23223], Loss: 0.0033
Epoch [2/4], Step [12000/23223], Loss: 0.0001
Epoch [2/4], Step [14000/23223], Loss: 0.0021
Epoch [2/4], Step [16000/23223], Loss: 0.0000
Epoch [2/4], Step [18000/23223], Loss: 0.0000
Epoch [2/4], Step [20000/23223], Loss: 0.0002
Epoch [2/4], Step [22000/23223], Loss: 0.0012
Epoch [2/4], Step [24000/23223], Loss: 0.0000
Epoch [2/4], Step [26000/23223], Loss: 0.0000
Epoch [3/4], Step [2000/23223], Loss: 0.0001
Epoch [3/4], Step [4000/23223], Loss: 0.0000
Epoch [3/4], Step [6000/23223], Loss: 0.0002
Epoch [3/4], Step [8000/23223], Loss: 0.0011
Epoch [3/4], Step [10000/23223], Loss: 0.0016
Epoch [3/4], Step [12000/23223], Loss: 0.0015
Epoch [3/4], Step [14000/23223], Loss: 0.0004
Epoch [3/4], Step [16000/23223], Loss: 0.0003
Epoch [3/4], Step [18000/23223], Loss: 0.0001
Epoch [3/4], Step [20000/23223], Loss: 0.0004
Epoch [3/4], Step [22000/23223], Loss: 0.0002
Epoch [4/4], Step [2000/23223], Loss: 0.0000
Epoch [4/4], Step [4000/23223], Loss: 0.0002
Epoch [4/4], Step [6000/23223], Loss: 0.0012
Epoch [4/4], Step [8000/23223], Loss: 0.0002
Epoch [4/4], Step [10000/23223], Loss: 0.0002
Epoch [4/4], Step [12000/23223], Loss: 0.0002
Epoch [4/4], Step [14000/23223], Loss: 0.0000
Epoch [4/4], Step [16000/23223], Loss: 0.0002
Epoch [4/4], Step [18000/23223], Loss: 0.0010
Epoch [4/4], Step [20000/23223], Loss: 0.0007
Epoch [4/4], Step [22000/23223], Loss: 0.0000
Finished Training
```

Improvement:

- CNN
- Adjust ability to read a masked face
- Ability to read multiple masked faces

Potential Use:

- Security camera
- Ensure people are wearing masks before entrance

References

- Geitgey, A. (2020, October 5). Build a face recognition system for \$60 with the new Nvidia Jetson Nano 2GB and Python. Medium.
<https://medium.com/@ageitgey/build-a-face-recognition-system-for-60-with-the-new-nvidia-jetson-nano-2gb-and-python-46edbddd7264>
- Face-recognition. (2020, February 20). PyPI. <https://pypi.org/project/face-recognition/>
- Cabani/maskedface-net: Maskedface-net is a dataset of human faces with a correctly and incorrectly worn mask based on the dataset flickr-faces-HQ (FFHQ). (2021, April 28). GitHub. <https://github.com/cabani/MaskedFace-Net>
- Pahinkar, A. (2020, August 25). Implementing CNN in PyTorch with custom dataset and transfer learning. Medium.
<https://medium.com/analytics-vidhya/implementing-cnn-in-pytorch-with-custom-dataset-and-transfer-learning-1864daac14cc>
- Training a classifier — PyTorch tutorials 1.9.0+cu102 documentation. (2021). PyTorch.
https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

Thank You!
Any Questions?