Big Data Programming Project

By: Kim Wurster, a24kimwu

Overview

Emotion Recognition - CNN

It is difficult for autistic people to communicate and detect non verbal cues when interacting with people. Having a tool that can read different body languages could help autistic folks to improve their communication skills.

How can we create a web solution that is able to detect different emotions in real time using a Convolutional Neural Network?

Data Preparation

Test and Training Set

- Training: **60%**
- Testing: **25%**
- Validation: **15%**
- Final CNN trained with **400** pictures per class
- Random picking from folder

Preprocessing

- Set image size - **(224, 224)**
- preprocessing of all 3 sets using **ImageDataGenerator()**
- generating batches of tensor image data with real-time data augmentation
- improved training process and robustness

The Data

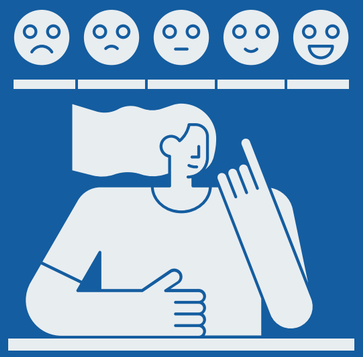
From where?

Kaggle Dataset (2024)
https://www.kaggle.com/datasets/sujaykapadnis/emotion-recognition-dataset

What data?

Pictures of 5 different emotions:

- Angry
- Happy
- Neutral
- Sad
- Surprise



Data Structure

- Testing and Training Folders, each folder:
- 5 different folders for 5 different emotions

The Plan & The Story

Plan

- Build CNN Model
  - using atleast 1000 pictures for training
- Take the CNN Model and use it in a Web Application
  - access web camera using cv2
- Web Application should be able to detect emotion in real time

ResNet50

- deep convolutional neural network (50 layers)
- Uses bottleneck layers (1x1, 3x3, 1x1 convolutions) -> reducing computation while maintaining performance
- Offers improved accuracy over shallower models like ResNet-34 and VGG-16 (Zafar, 2023)

OpenCV

- Infrastructure for Computer Vision applications
- Video Capturing from Webcams (Chauke, 2022)

4. CNN Model

```
base_model = ResNet50(include_top=False, weights='imagenet')
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
x = Dropout(0.5) (x)
x = Dense(1024, activation='relu')(x)
x = Dropout(0.5) (x)
predictions = Dense(train_generator.num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)

#for layer in base_model.layers:
    #layer.trainable = False
```

```
#Change Learning Rate
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-5), loss='categorical_crossentropy', metrics=['accuracy'])

early_stopping = EarlyStopping(monitor='val_loss', patience=2, restore_best_weights=True, start_from_epoch=10)
```

```
#change parameters
model.fit(train_generator, epochs = 40, validation_data = valid_generator, callbacks=[early_stopping])
```

```python
# Function to preprocess live feed frames
def preprocess_live_frame(frame):
    # Resize the frame to match the input size of the trained model
    resized_frame = cv2.resize(frame, (224, 224))
    return np.expand_dims(resized_frame, axis=0)

# Access the live camera feed
cap = cv2.VideoCapture(0)  # Use 0 for the default camera, you may need to change this number

while True:
    ret, frame = cap.read()
    if not ret:
        break

    # Preprocess the live frame
    processed_frame = preprocess_live_frame(frame)

    # Perform inference using the trained model
    predictions = model.predict(processed_frame)
    confidence = np.max(predictions)
    emotion_label = emotions[np.argmax(predictions)]

    # Display recognized emotion and confidence level on the camera feed
    text = f"Emotion: {emotion_label}, Confidence: {confidence:.2f}"
    cv2.putText(frame, text, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2, cv2.LINE_AA)

    # Show the camera feed with emotion and confidence
    cv2.imshow('Emotion Recognition', frame)

    # Exit loop if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

The Plan & The Story

Long Computing Time

Training took a long time

Final version: ~ 3.5 hours

Not using a pretrained model

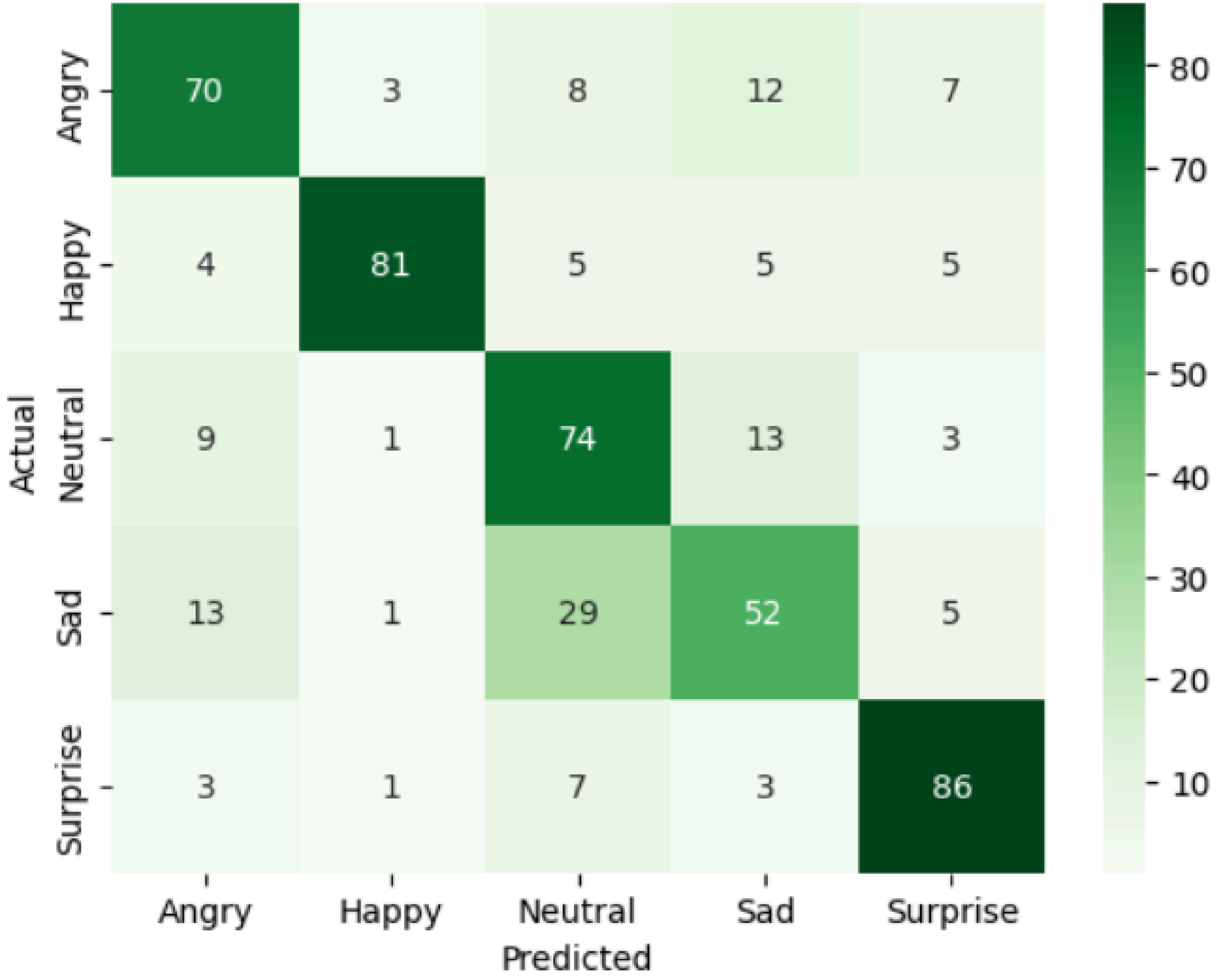
Low accuracy (<40%)

A lot of different parameters that could be changed to improve model

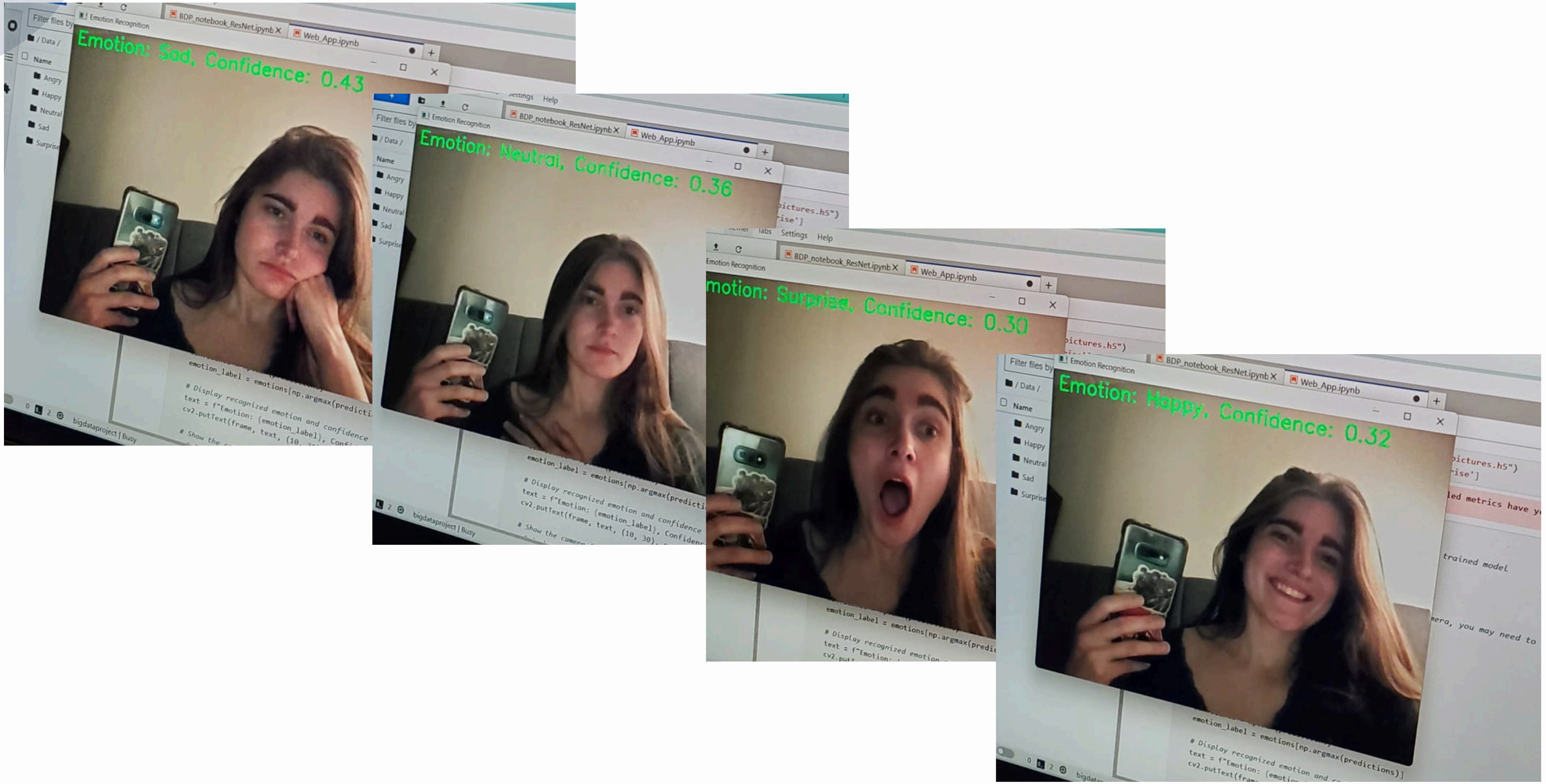
Low accuracy

Often classifying other emotions as neutral, especially sad

Difficult to display/detect angry in life stream video



test accuracy: 72.6


Emotion: Sad, Confidence: 0.43
Emotion: Neutral, Confidence: 0.36
motion: Surprise, Confidence: 0.30
Emotion: Happy, Confidence: 0.32
Emotion Recognition
BDP_notebook_ResNet.ipynb
Web_App.ipynb

Results of your investigation

- Achieved higher accuracy when using a pretrained model like ResNet50
- The CNN works well for detecting the emotions it is supposed to classify but has difficulty with micro emotions
- Training on more pictures -> better accuracy
- Smaller learning rate -> better accuracy
- Greater image width and height -> better accuracy

Conclusion

How can we create a web solution that is able to detect different emotions in real time using a Convolutional Neural Network?

01 Programming in Python using Keras, Tensorflow and OpenCV

02 Usage of a pretrained Model (in this case ResNet50) for the CNN

03 Using Streamlit to display the cv2 project in a web app using Streamlit Cloud (Sagar, 2021)

Discussion

01

Make the CNN more accurate (> 95%)

- improve parameters
- train on more pictures
- Use GridSearchCV

02

Create an actual web Application in form of a website using Streamlit

03

Train on a larger set of emotions (disgust, fear, etc.) and on micro emotions

04

End goal: Glasses that can detect emotions

- implemement feature makes recommendations on how to react to specific emotions

References

Chauke, Z. (01.09.2022). *Build a Computer Vision Web App — Flask, OpenCV, and MongoDB.* Medium. https://medium.com/better-programming/build-a-computer-vision-webapp-flask-opencv-and-mongodb-62a52d38738a

Kovenko, Volodymyr; Shevchuk, Vitalii (2021). *OAHEGA : EMOTION RECOGNITION DATASET*, Mendeley Data, V2, doi: 10.17632/5ck5zz6f2c.2

Sagar, D. (10.11.2021). How to Build an OpenCV Web App with Streamlit. loginradius. https://www.loginradius.com/blog/engineering/guest-post/opencv-web-app-with-streamlit/

Zafar, S. (12.11.2023). *Choosing the Right Pre-Trained Model: A Guide to VGGNet, ResNet, GoogleNet, AlexNet, and Inception*. Medium. https://medium.com/@sohaib.zafar522/choosing-the-right-pre-trained-model-a-guide-to-vggnet-resnet-googlenet-alexnet-and-inception-db7a8c918510

QUESTIONS?