CST Assignment #4

H3Art

Introduction

The Significance of Facial Emotion Recognition

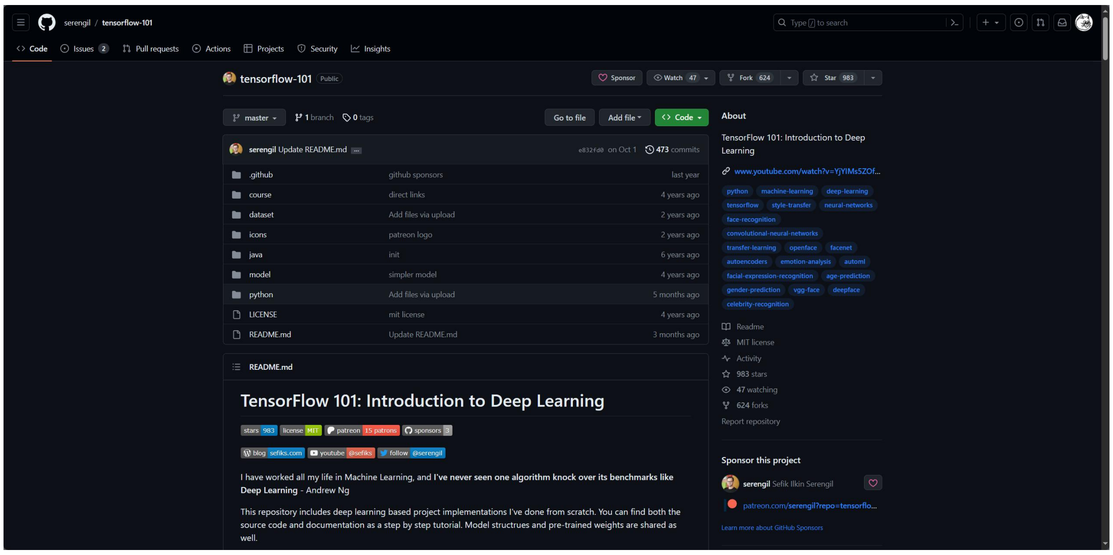
Facial emotion recognition technology represents a vital intersection in the domain of Human-Computer Interaction (HCI). This technology not only enables computers to understand and react to human emotions through facial expressions but also opens new avenues for creating more empathetic and intuitive user interfaces. Its significance lies in bridging the emotional gap between humans and machines, facilitating a more natural form of communication that goes beyond mere verbal or textual interaction.

Purpose of the Report

This report aims to delve deep into the realm of facial emotion recognition within HCI. It seeks to explore how this technology enhances user experience, making interactions with machines more humane and emotionally aware. I tried to obtain a facial emotion recognition project on the open source code website Github and deploy it, present its use effects in the report, and discuss the relationship between facial emotion recognition technology and human-computer interaction.

Project Overview

An available facial emotion recognition project I obtained is from tensorflow-101. This is an open source project that integrates multiple deep learning applications. The following is a partial introduction:



serengil Update README.md … e832fd0 on Oct 1 473 commits

.github	github sponsors	last year
course	direct links	4 years ago
dataset	Add files via upload	2 years ago
icons	patreon logo	2 years ago
java	init	6 years ago
model	simpler model	4 years ago
python	Add files via upload	5 months ago
LICENSE	mit license	4 years ago
README.md	Update README.md	3 months ago

README.md

TensorFlow 101: Introduction to Deep Learning

stars 983 license MIT patreon 15 patrons sponsors 3

blog sefiks.com youtube @sefiks follow @serengil

I have worked all my life in Machine Learning, and **I've never seen one algorithm knock over its benchmarks like Deep Learning** - Andrew Ng

This repository includes deep learning based project implementations I've done from scratch. You can find both the source code and documentation as a step by step tutorial. Model structrues and pre-trained weights are shared as well.

About

TensorFlow 101: Introduction to Deep Learning

www.youtube.com/watch?v=YjYIMs5ZOf...

python machine-learning deep-learning tensorflow style-transfer neural-networks face-recognition convolutional-neural-networks transfer-learning openface facenet autoencoders emotion-analysis automl facial-expression-recognition age-prediction gender-prediction vgg-face deepface celebrity-recognition

Readme
MIT license
Activity
983 stars
47 watching
624 forks
Report repository

Sponsor this project

serengil Sefik Ilkin Serengil

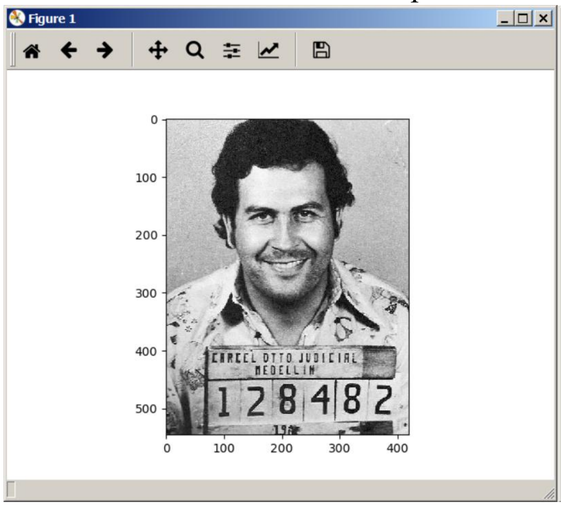
patreon.com/serengil?repo=tensorflo...

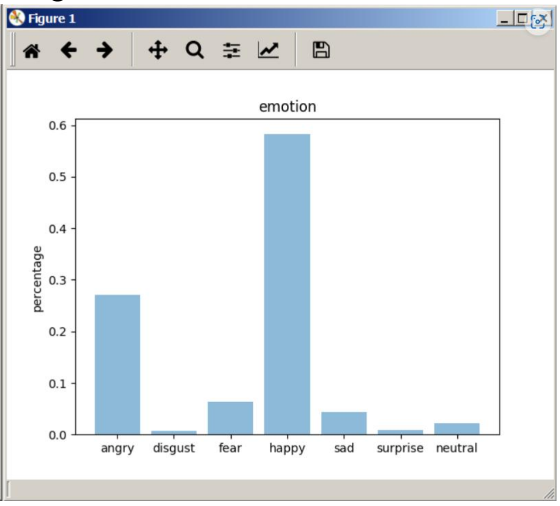
Learn more about GitHub Sponsors

Project Introduction

The TensorFlow 101 repository, titled "TensorFlow 101: Introduction to Deep

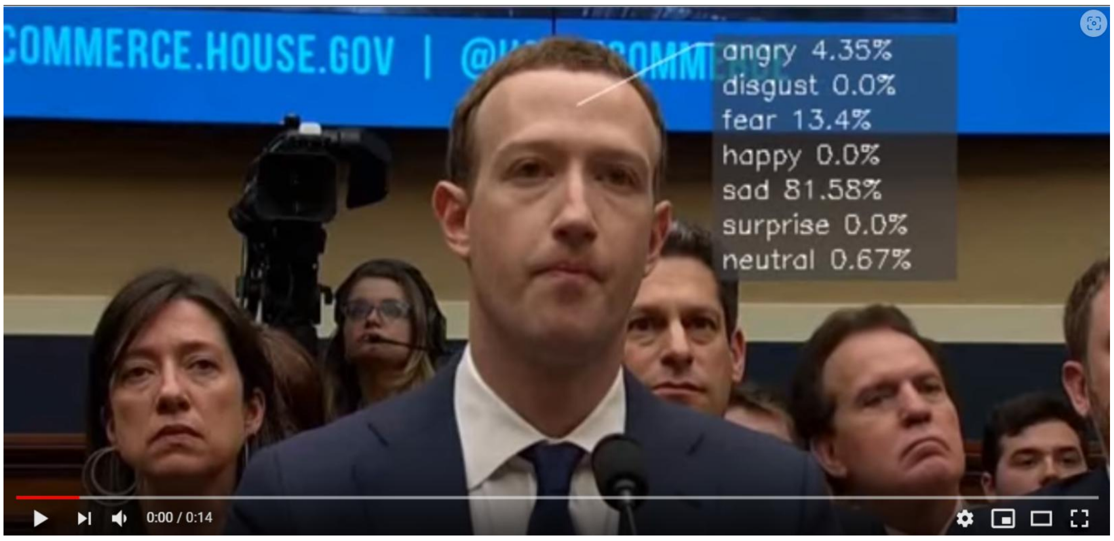
Learning", presents a collection of deep learning-based project implementations, offers a comprehensive resource for learning and applying deep learning techniques, with a special emphasis on facial expression recognition. This repository is a rich source of both source code and step-by-step tutorials, providing valuable insights into model structures and the use of pre-trained weights.





Real-Time Emotion Analysis Capability

One of the key features of this project is its ability to perform real-time emotion analysis. This capability demonstrates the practical application of the project in dynamic, real-world scenarios, where instant emotional recognition can be crucial, such as in interactive applications or user experience enhancements.



```
angry 4.35%
disgust 0.0%
fear 13.4%
happy 0.0%
sad 81.58%
surprise 0.0%
neutral 0.67%
```

Facial Recognition Technology

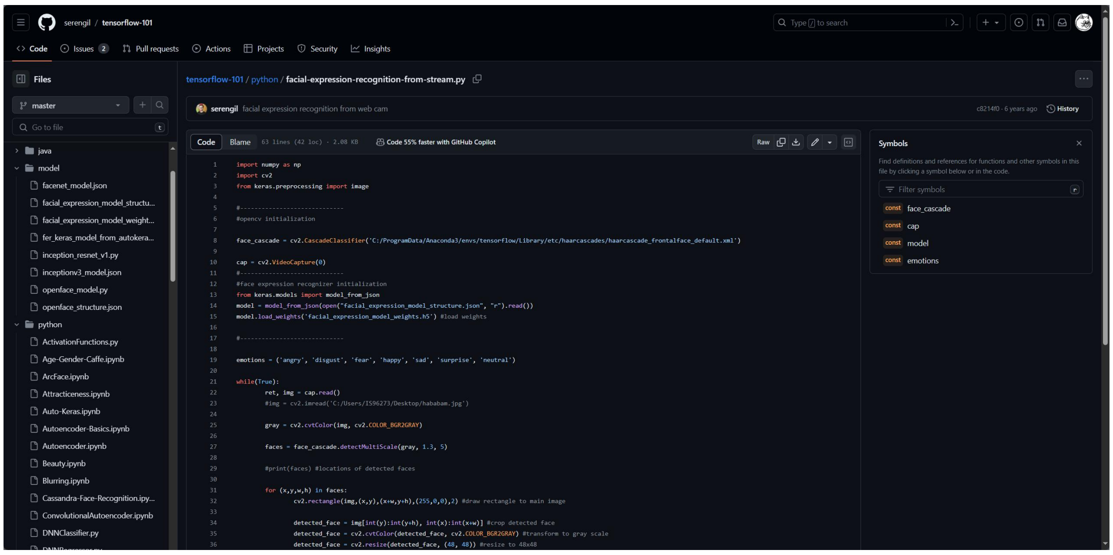
The project heavily relies on convolutional neural networks (CNNs), a cornerstone in modern facial recognition technology. It includes implementations where two facial images are fed into a CNN model, generating multi-dimensional vector representations. These representations are then compared to determine if the two images belong to the same person, showcasing the project's utility in identity verification and security applications.

Integration of Advanced Models

TensorFlow 101 incorporates various state-of-the-art facial recognition models, including VGG-Face, FaceNet, DeepFace, and others. Each of these models has been developed by leading institutions and companies, reflecting high levels of accuracy and reliability. These models are wrapped in the deepface library for Python, making it convenient to build and run sophisticated facial recognition systems with minimal coding. This integration emphasizes the project's focus on providing accessible yet powerful tools for facial emotion and identity recognition.

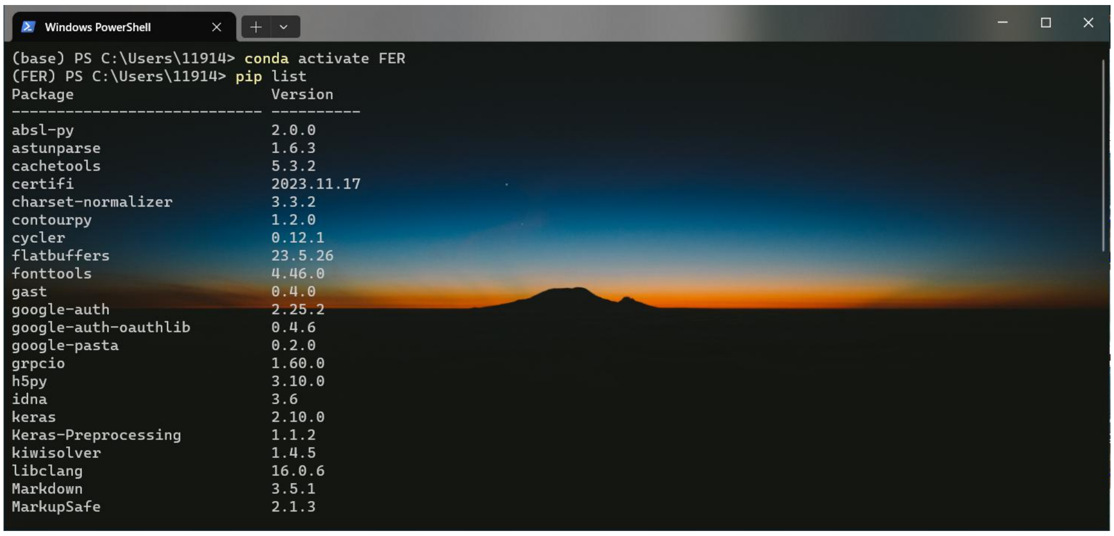
Deploying Project

The entire TensorFlow 101 repository contains a lot of content, including a large number of Python files and Jupyter notebook files, but in my final deployment process, I do not need a large number of other types of deep learning applications and training codes, I only need to deploy the applications , leaving only the model checkpoints to load to run the project content.



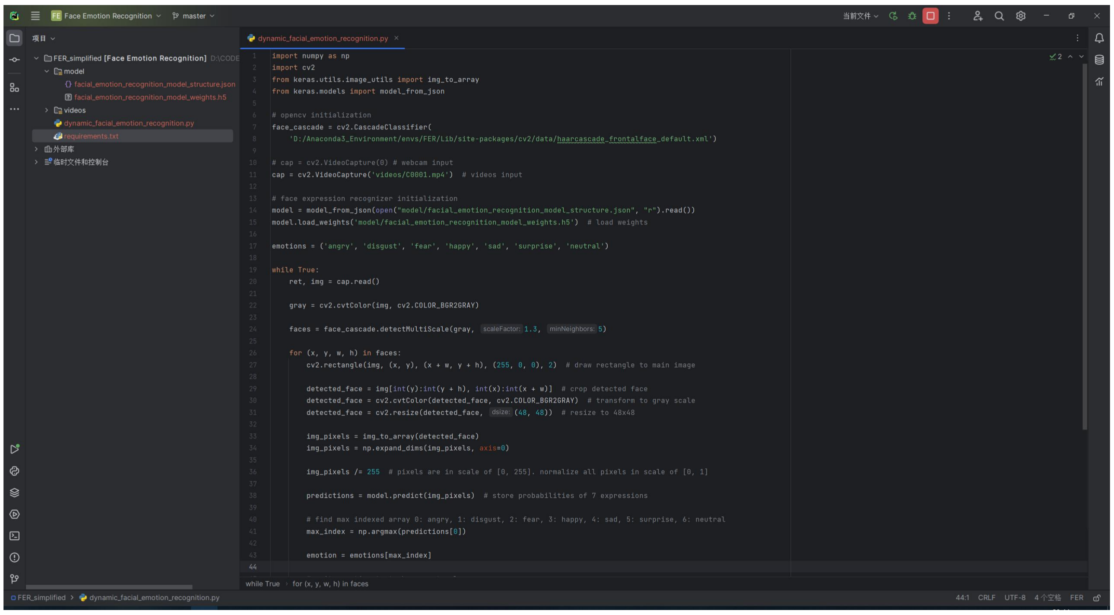
```python
import numpy as np
import cv2
from keras.preprocessing import image

#----------------------------
#opencv initialization

face_cascade = cv2.CascadeClassifier('C:/ProgramData/Anaconda3/envs/tensorflow/Library/etc/haarcascades/haarcascade_frontalface_default.xml')

cap = cv2.VideoCapture(0)
#----------------------------
#face expression recognizer initialization
from keras.models import model_from_json
model = model_from_json(open("facial_expression_model_structure.json", "r").read())
model.load_weights('facial_expression_model_weights.h5') #load weights

#----------------------------

emotions = ('angry', 'disgust', 'fear', 'happy', 'sad', 'surprise', 'neutral')

while(True):
	ret, img = cap.read()
	#img = cv2.imread('C:/Users/IS96273/Desktop/hababam.jpg')

	gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

	faces = face_cascade.detectMultiScale(gray, 1.3, 5)

	#print(faces) #locations of detected faces

	for (x,y,w,h) in faces:
		cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2) #draw rectangle to main image

		detected_face = img[int(y):int(y+h), int(x):int(x+w)] #crop detected face
		detected_face = cv2.cvtColor(detected_face, cv2.COLOR_BGR2GRAY) #transform to gray scale
		detected_face = cv2.resize(detected_face, (48, 48)) #resize to 48x48
```

Before running, I had to ensure that my environment was configured correctly. After I used conda to create a new Python 3.10 environment named FER, I installed some libraries needed for this project, including numpy, tensorflow, keras, opencv-python, etc.



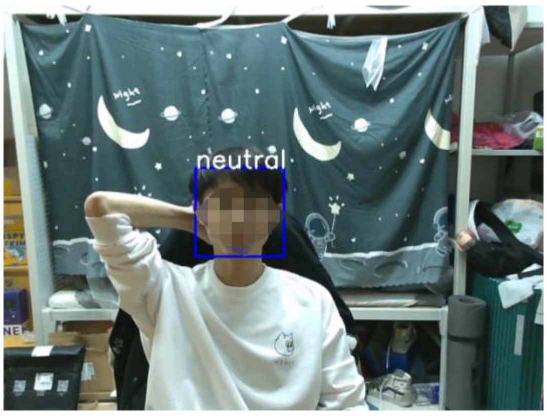
```
(base) PS C:\Users\11914> conda activate FER
(FER) PS C:\Users\11914> pip list
Package                      Version
---------------------------- ----------
absl-py                      2.0.0
astunparse                   1.6.3
cachetools                   5.3.2
certifi                      2023.11.17
charset-normalizer           3.3.2
contourpy                    1.2.0
cycler                       0.12.1
flatbuffers                  23.5.26
fonttools                    4.46.0
gast                         0.4.0
google-auth                  2.25.2
google-auth-oauthlib         0.4.6
google-pasta                 0.2.0
grpcio                       1.60.0
h5py                         3.10.0
idna                         3.6
keras                        2.10.0
Keras-Preprocessing          1.1.2
kiwisolver                   1.4.5
libclang                     16.0.6
Markdown                     3.5.1
MarkupSafe                   2.1.3
```

After configuring the environment, I streamlined the entire project content. In the end, only two model checkpoint files and a Python script code to start dynamic facial emotion recognition were left. At the same time, I briefly modified the logic of this script code so that In addition to calling my computer's camera for facial emotion recognition, it can also read video files and perform emotion recognition on faces in the video.

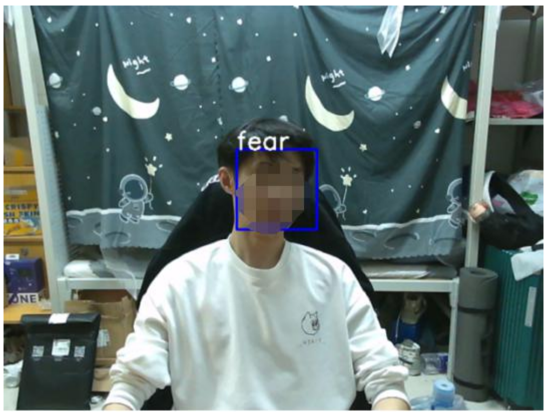

Face Emotion Recognition
master
项目
FER_simplified [Face Emotion Recognition] D:\CODE
model
facial_emotion_recognition_model_structure.json
facial_emotion_recognition_model_weights.h5
videos
dynamic_facial_emotion_recognition.py
requirements.txt
外部库
临时文件和控制台
dynamic_facial_emotion_recognition.py
while True › for (x, y, w, h) in faces
FER_simplified › dynamic_facial_emotion_recognition.py

```python
import numpy as np
import cv2
from keras.utils.image_utils import img_to_array
from keras.models import model_from_json

# opencv initialization
face_cascade = cv2.CascadeClassifier(
    'D:/Anaconda3_Environment/envs/FER/Lib/site-packages/cv2/data/haarcascade_frontalface_default.xml')

# cap = cv2.VideoCapture(0) # webcam input
cap = cv2.VideoCapture('videos/C0001.mp4')  # videos input

# face expression recognizer initialization
model = model_from_json(open("model/facial_emotion_recognition_model_structure.json", "r").read())
model.load_weights('model/facial_emotion_recognition_model_weights.h5')  # load weights

emotions = ('angry', 'disgust', 'fear', 'happy', 'sad', 'surprise', 'neutral')

while True:
    ret, img = cap.read()

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(gray, scaleFactor: 1.3, minNeighbors: 5)

    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)  # draw rectangle to main image

        detected_face = img[int(y):int(y + h), int(x):int(x + w)]  # crop detected face
        detected_face = cv2.cvtColor(detected_face, cv2.COLOR_BGR2GRAY)  # transform to gray scale
        detected_face = cv2.resize(detected_face, dsize: (48, 48))  # resize to 48x48

        img_pixels = img_to_array(detected_face)
        img_pixels = np.expand_dims(img_pixels, axis=0)

        img_pixels /= 255  # pixels are in scale of [0, 255]. normalize all pixels in scale of [0, 1]

        predictions = model.predict(img_pixels)  # store probabilities of 7 expressions

        # find max indexed array 0: angry, 1: disgust, 2: fear, 3: happy, 4: sad, 5: surprise, 6: neutral
        max_index = np.argmax(predictions[0])

        emotion = emotions[max_index]

```

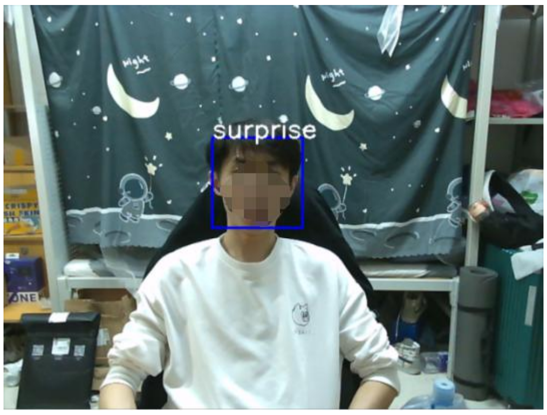
I personally tested all six expressions and emotions in this project, but the recognition accuracy was not high. The recognition response of this system was still relatively slow. To get accurate results, I needed to pause my expression.



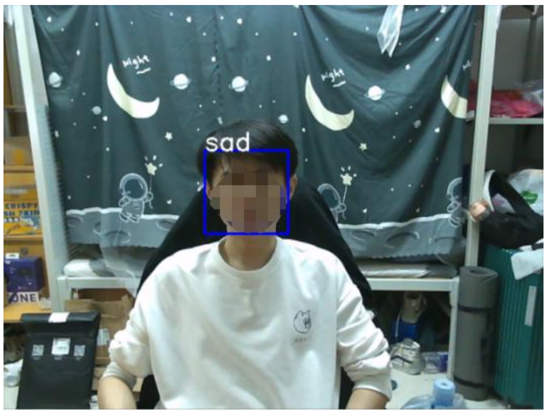
Emotion: neutral



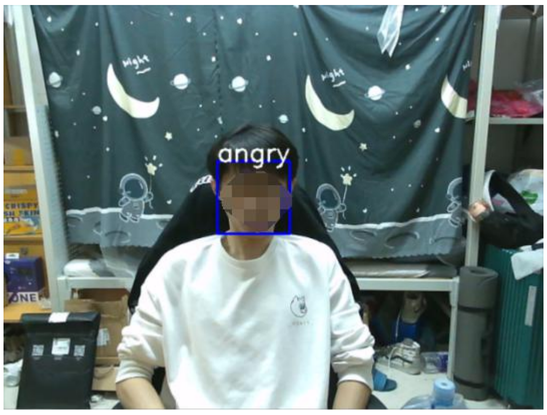
Emotion: fear



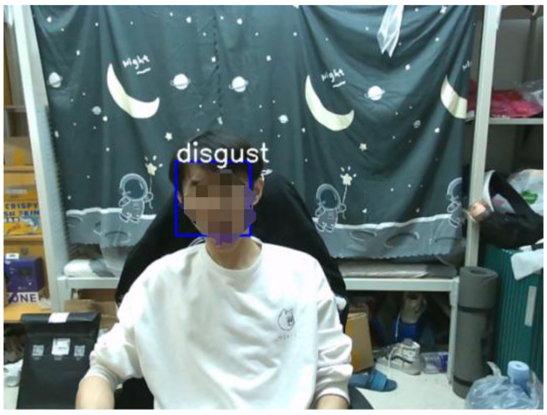
Emotion: surprise



Emotion: sad



Emotion: angry



Emotion: disgust

In addition, I also used some videos for testing, and some results can be obtained as follows:

Emotion recognition in video clip

Emotion recognition in video clip

Reflections and Conclusions

Technical Challenges and Solutions

During the deployment of the TensorFlow 101 project, I encountered several technical challenges, particularly regarding library compatibility. A notable issue was the absence of the `img_to_array` method in `keras.preprocessing.image`, a result of using an updated version of the Keras library where this method had been moved to `keras.utils.image_utils`. This experience underscores the importance of thorough documentation review and the need to stay updated with library changes in fast-evolving fields like deep learning.

Furthermore, I observed that the project's face recognition sampling rate was relatively low. To enhance user experience in real-time applications, increasing this rate is crucial. However, this improvement comes with a trade-off, potentially increasing the computational load and demanding more from the GPU. This challenge highlights the delicate balance between performance and resource efficiency in facial emotion recognition systems.

Personal Insights and Future Outlook

Working on this project has been an enlightening journey, deepening my understanding of the intricate relationship between facial emotion recognition and human-computer interaction. This technology not only makes interactions more natural and empathetic but also opens doors to innovative applications in various fields.

In the future outlook of human-computer interaction (HCI), the integration of facial emotion recognition is set to deepen the interaction between humans and machines. This technology enables a more intuitive and empathetic mode of interaction, where machines can understand and respond to human emotions, transcending traditional command-based interactions.

Facial emotion recognition can revolutionize various fields. In healthcare, it can provide personalized mental health support by analyzing emotional states. In education, adapting teaching methods based on students' emotional responses can enhance learning experiences. The entertainment and gaming industry can utilize this technology to alter game narratives based on the player's emotions, creating more immersive experiences. Customer service can benefit by understanding and responding to client emotions, improving service quality. Additionally, this technology has significant potential in assistive technologies, helping individuals with communication challenges to interact more effectively.

These applications highlight the transformative potential of facial emotion recognition in enhancing HCI, pointing towards a future where technology not only understands our commands but also our emotions and intentions, paving the way for a more intelligent and empathetic interactive world.