



- **University:** JNU
- **Department:** Computer Science and Technology
- **Course:** Human-Computer Interaction
- **Project Title:** GitHub Project (Facial detection and emotion recognition)
- **Project Part:** Part 4
- **Author:** 蒋云翔 2022102330 (Yunxiang Jiang) (Accomplish the task by myself only)
- **Instructor:** 龙锦益 (Jinyi Long)
- **Date:** December 4, 2024

Catalogue

1. Brief Introduction to GitHub project	2
1.1 Website URL:	2
1.2 Project's brief introduction	2
2. Develop environment	5
3. Running result analysis	7

1. Brief Introduction to GitHub project

1.1 Website URL:

[serengil/deepface: A Lightweight Face Recognition and Facial Attribute Analysis \(Age, Gender, Emotion and Race\) Library for Python](https://github.com/serengil/deepface)

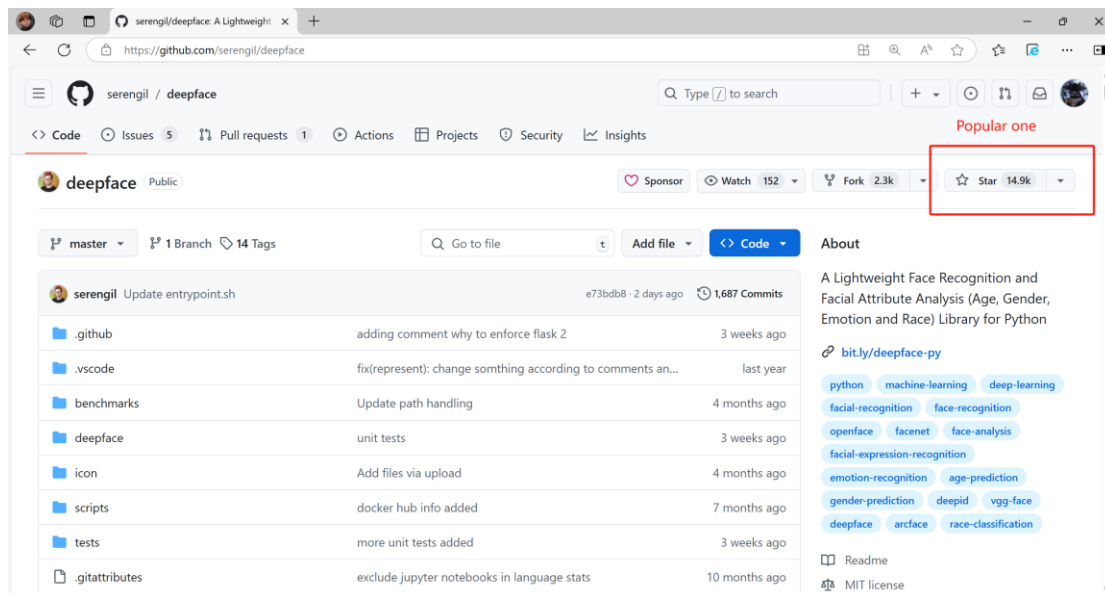


Figure 1: GitHub URL link

1.2 Project's brief introduction

DeepFace is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: **VGG-Face, FaceNet, OpenFace, DeepFace, DeepID, ArcFace, Dlib, SFace and GhostFaceNet**.

Experiments show that human beings have **97.53%** accuracy on facial recognition tasks whereas those models already reached and passed that accuracy level.

deepface

downloads 3M stars 15k license MIT Tests and Linting passing DOI 10.17671/gazibtd.1399077

blog sefiks.com youtube @sefiks X follow @serengil

patreon 41 patrons sponsors 1 buy me a coffee



DeepFace is a lightweight [face recognition](#) and facial attribute analysis ([age](#), [gender](#), [emotion](#) and [race](#)) framework for python. It is a hybrid face recognition framework wrapping [state-of-the-art](#) models: [VGG-Face](#), [FaceNet](#), [OpenFace](#), [DeepFace](#), [DeepID](#), [ArcFace](#), [Dlib](#), [SFace](#) and [GhostFaceNet](#).

[Experiments](#) show that **human beings** have 97.53% accuracy on facial recognition tasks whereas those models already reached and passed that accuracy level.

Figure 2: The README file of the project

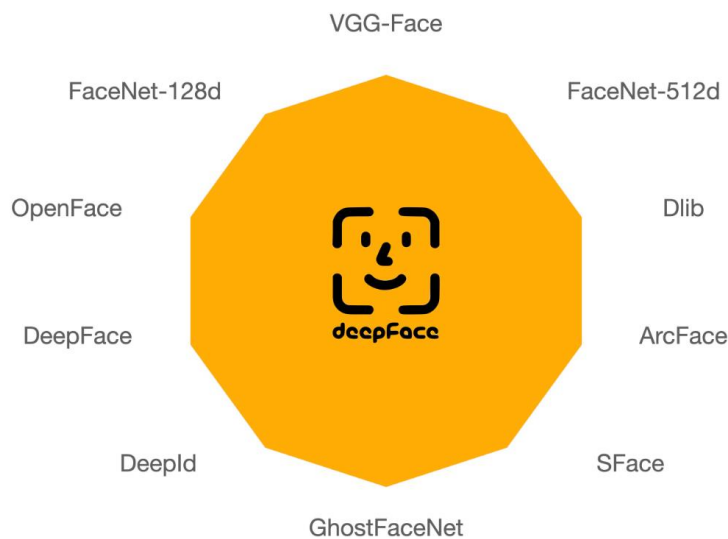


Figure 3: The related models that be used in deepFace

FaceNet, VGG-Face, ArcFace and Dlib are overperforming ones based on experiments - see **BENCHMARKS** for more details. You can find the **measured scores** of various models in DeepFace and the **reported scores** from their original studies in the following table.

Model	Measured Score	Declared Score
Facenet512	98.4%	99.6%
Human-beings	97.5%	97.5%
Facenet	97.4%	99.2%
Dlib	96.8%	99.3 %
VGG-Face	96.7%	98.9%
ArcFace	96.7%	99.5%
GhostFaceNet	93.3%	99.7%
SFace	93.0%	99.5%
OpenFace	78.7%	92.9%
DeepFace	69.0%	97.3%
DeepID	66.5%	97.4%

Figure 4: The measured and Declared score of different models

This can be used to detect and recognize a variety of information, such as age, gender, emotion, race, etc. Our project mainly applies its function of **identifying emotions**.

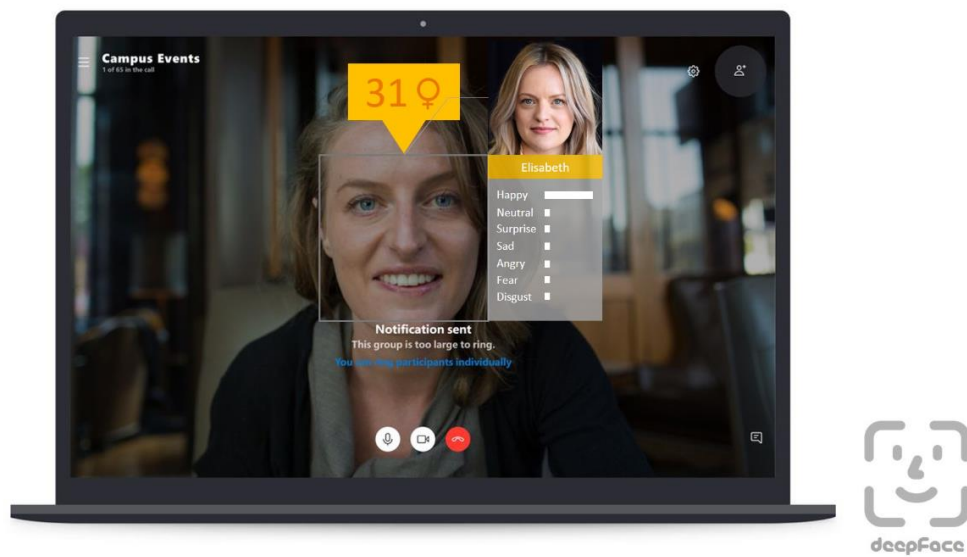


Figure 5: A using example of deepFace

2. Develop environment

This is an integrated package, which contains the we need such as TensorFlow, matplotlib, OpenCV python, NumPy, pandas and so on.

Installation pyPI v0.0.93

The easiest way to install deepface is to download it from [PyPI](#). It's going to install the library itself and its prerequisites as well.

```
$ pip install deepface
```

Perform this command in cmd

Alternatively, you can also install deepface from its source code. Source code may have new features not published in pip release yet.

```
$ git clone https://github.com/serengil/deepface.git
$ cd deepface
$ pip install -e .
```

Once you installed the library, then you will be able to import it and use its functionalities.

```
from deepface import DeepFace
```

Figure 6: The instruction of environment building method

```
(demo) PS C:\Users\86136\Desktop\deepface-master> pip install deepface
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Requirement already satisfied: deepface in c:\anaconda\envs\demo\lib\site-packages (0.0.93)
Requirement already satisfied: requests>=2.27.1 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (2.32.3)
Requirement already satisfied: numpy>=1.14.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (2.0.2)
Requirement already satisfied: pandas>=0.23.4 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (2.2.3)
Requirement already satisfied: gdown>=3.10.1 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (5.2.0)
Requirement already satisfied: tqdm>=4.30.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (4.67.1)
Requirement already satisfied: Pillow>=5.2.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (11.0.0)
Requirement already satisfied: opencv-python>=4.5.5.64 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (4.10.0.84)
Requirement already satisfied: tensorflow>=1.9.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (2.18.0)
Requirement already satisfied: keras>=2.2.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (3.7.0)
Requirement already satisfied: Flask>=1.1.2 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (3.1.0)
Requirement already satisfied: flask-cors>=4.0.1 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (5.0.0)
Requirement already satisfied: mtcnn>=0.1.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (0.1.1)
Requirement already satisfied: retina-face>=0.0.1 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (0.0.17)
Requirement already satisfied: fire>=0.4.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (0.7.0)
Requirement already satisfied: gunicorn>=20.1.0 in c:\anaconda\envs\demo\lib\site-packages (from deepface) (23.0.0)
Requirement already satisfied: termcolor in c:\anaconda\envs\demo\lib\site-packages (from fire>=0.4.0->deepface) (2.5.0)
Requirement already satisfied: Werkzeug>=3.1 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (3.1.3)
Requirement already satisfied: Jinja2>=3.1.2 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (3.1.4)
Requirement already satisfied: itsdangerous>=2.2 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (2.2.0)
Requirement already satisfied: click>=8.1.3 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (8.1.7)
Requirement already satisfied: blinker>=1.9 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (1.9.0)
Requirement already satisfied: importlib-metadata>=3.6 in c:\anaconda\envs\demo\lib\site-packages (from Flask>=1.1.2->deepface) (8.5.0)
Requirement already satisfied: beautifulsoup4 in c:\anaconda\envs\demo\lib\site-packages (from gdown>=3.10.1->deepface) (4.12.3)
```

Perform this pip command as shown in the README.md and we can build our project environment.
I have configured the experimental environment before making this report, so it will show that all packages have been satisfied.

Figure 7: Configure our project environment

There were some minor incidents during the configuration experiment. Since this project was created several years ago, the configuration environment at that time was a little older than the current one, so I needed to configure some packages with older versions to meet this project. So I used **anaconda** to create a python 3.9 virtual environment to run the project

```
(base) C:\Users\86136>conda env list
# conda environments:
#
base                * C:\anaconda
demo                C:\anaconda\envs\demo
github              C:\anaconda\envs\github
limu                C:\anaconda\envs\limu

(base) C:\Users\86136>conda activate demo
(demo) C:\Users\86136>conda list
# packages in environment at C:\anaconda\envs\demo:
#
# Name                    Version            Build    Channel
absl-py                   2.1.0              pypi_0  pypi
astunparse                1.6.3              pypi_0  pypi
beautifulsoup4            4.12.3             pypi_0  pypi
blinker                   1.9.0              pypi_0  pypi
ca-certificates           2024.11.26         haaf9532.0
certifi                   2024.8.30          pypi_0  pypi
charset-normalizer        3.4.0              pypi_0  pypi
click                     8.1.7              pypi_0  pypi
colorama                  0.4.6              pypi_0  pypi
contourpy                 1.3.0              pypi_0  pypi
cycler                    0.12.1             pypi_0  pypi
deepface                  0.0.93             pypi_0  pypi
exceptiongroup            1.2.2              pypi_0  pypi
filelock                  3.16.1             pypi_0  pypi
fire                       0.7.0              pypi_0  pypi
flask                     3.1.0              pypi_0  pypi
flask-cors                5.0.0              pypi_0  pypi
flatbuffers               24.3.25            pypi_0  pypi
fonttools                 4.55.1             pypi_0  pypi
gast                      0.6.0              pypi_0  pypi
gdown                     5.2.0              pypi_0  pypi
```

I created this named 'demo' python interpreter to run the whole project

```

Anaconda Prompt (anac...
pip 24.2 py39haa95532_0
pluggy 1.5.0 pypi_0 pypi
protobuf 5.29.0 pypi_0 pypi
pygments 2.18.0 pypi_0 pypi
pyparsing 3.2.0 pypi_0 pypi
pysocks 1.7.1 pypi_0 pypi
pytest 8.3.4 pypi_0 pypi
python 3.9.20 h8205438_1
python-dateutil 2.9.0.post0 pypi_0 pypi
pytz 2024.2 pypi_0 pypi
requests 2.32.3 pypi_0 pypi
retina-face 0.0.17 pypi_0 pypi
rich 13.9.4 pypi_0 pypi
setuptools 75.1.0 py39haa95532_0
six 1.16.0 pypi_0 pypi
soupsieve 2.6 pypi_0 pypi
sqlite 3.45.3 h2bbff1b_0
tensorboard 2.18.0 pypi_0 pypi
tensorboard-data-server 0.7.2 pypi_0 pypi
tensorflow 2.18.0 pypi_0 pypi
tensorflow-intel 2.18.0 pypi_0 pypi
tensorflow-io-gcs-filesystem 0.31.0 pypi_0 pypi
termcolor 2.5.0 pypi_0 pypi
tf-keras 2.18.0 pypi_0 pypi
tomli 2.2.1 pypi_0 pypi
tqdm 4.67.1 pypi_0 pypi
typing-extensions 4.12.2 pypi_0 pypi
tzdata 2024.2 pypi_0 pypi
urllib3 2.2.3 pypi_0 pypi
vc 14.40 h2eaa2aa_1
vs2015_runtime 14.40.33807 h98bbidd_1
werkzeug 3.1.3 pypi_0 pypi
wheel 0.44.0 py39haa95532_0
wrapt 1.17.0 pypi_0 pypi
zipp 3.21.0 pypi_0 pypi

```

Figure 8: Using anaconda to create virtual env

3. Running result analysis

In the analyze module of deepFace, emotions can be recognized as follows: **angry, disgust, fear, happy, sad, surprise and neutral**. This model will weigh and score the recognized emotions in the image, and finally get a **dominant emotion** as the final judgment result

```

from deepface import DeepFace
import cv2
import matplotlib.pyplot as plt

img_path = './dataset/Happy.jpg'
img = cv2.imread(img_path)
plt.imshow(img[:, :, ::-1])
demography = DeepFace.analyze(img_path)
print(demography)

```

```

{'angry': np.float32(3.3538613e-11), 'disgust': np.float32(3.3538613e-11), 'fear': np.float32(3.5967543e-07), 'happy': np.float32(99.93684), 'sad': np.float32(1.4474946e-06), 'surprise': np.float32(1.4474946e-06)}

```

We can indicate that this picture's emotion is happy since its weight is very close to 100

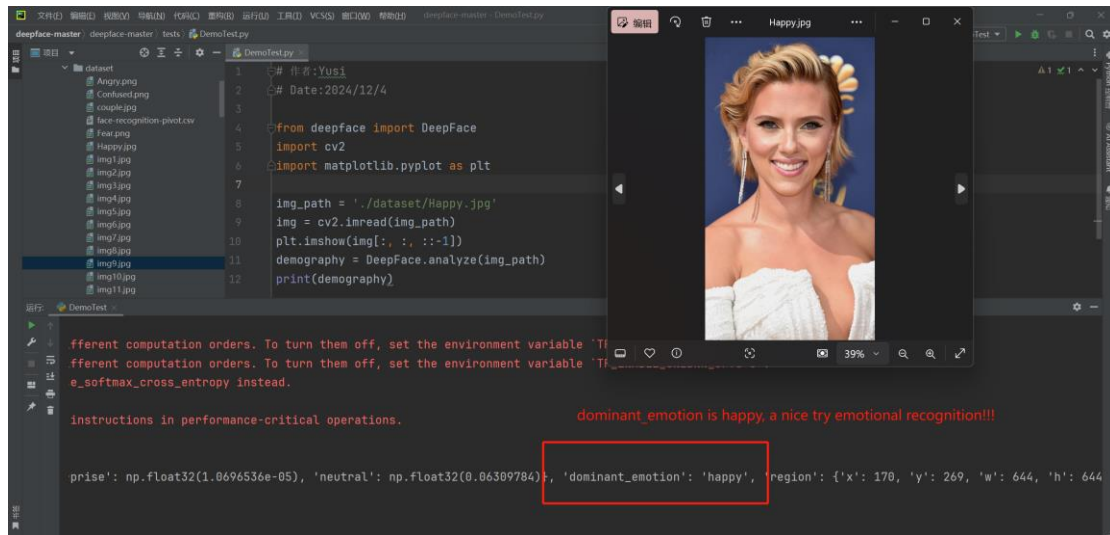


Figure 9: Test example (Happy emotion recognition)

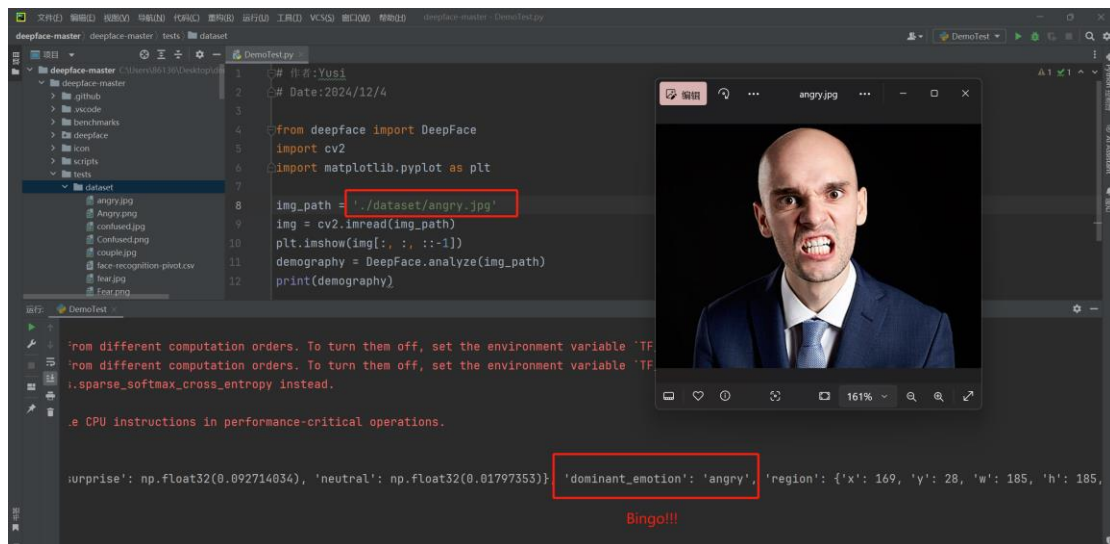


Figure 10: Test example (Angry emotion recognition)

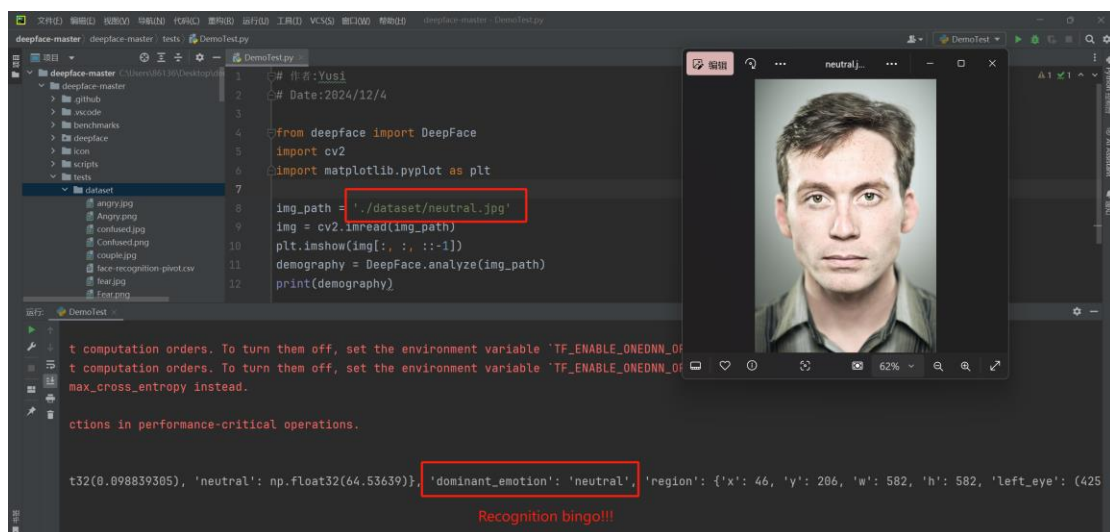


Figure 11: Test example (Neutral emotion recognition)

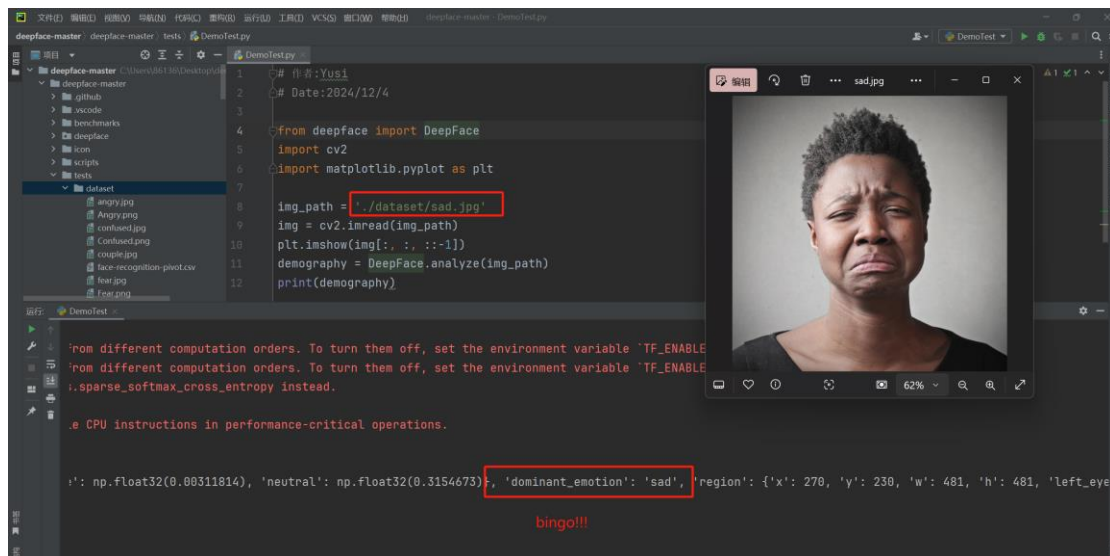


Figure 12: Test example (Sad emotion recognition)

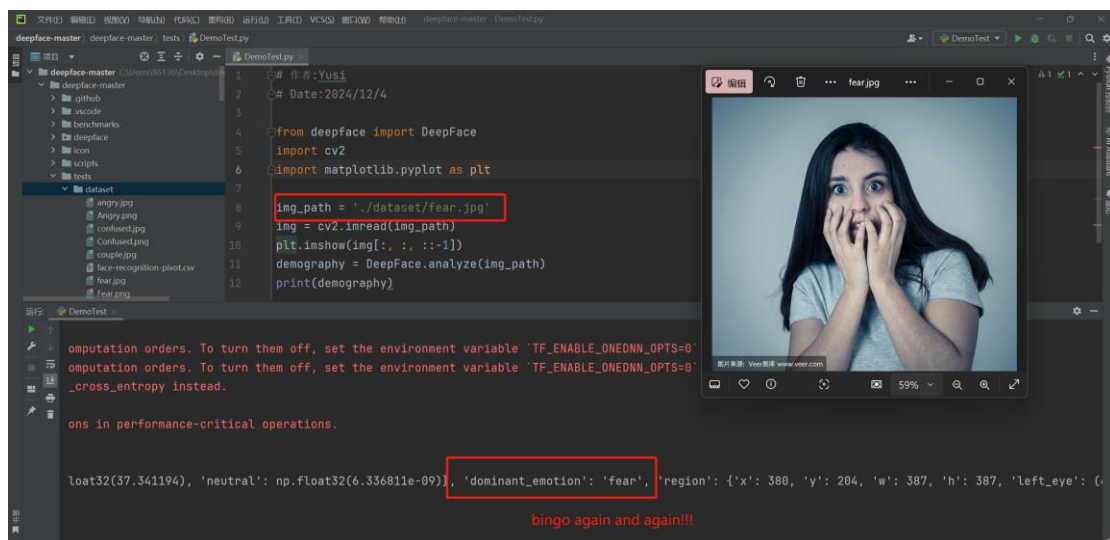


Figure 13: Test example (Fear emotion recognition)

As you can see, all our test cases produced correct mood predictions!!