Emotion Detection with Facial Expression Recognition

Introduction:

In this project, I developed an Emotion Detection system using deep learning techniques, specifically Convolutional Neural Networks (CNNs), to recognize facial expressions from images. The system is capable of identifying seven different emotions: anger, disgust, fear, happiness, neutral, sadness, and surprise.

Libraries Used:

Keras: For building and training the CNN model.

OpenCV (cv2): For image processing tasks, including webcam access and face detection.

Pandas: For data manipulation and organization.

NumPy: For numerical computations and array manipulations.

Matplotlib: For plotting and visualizing images and predictions.

scikit-learn (sklearn): For preprocessing tasks like label encoding.

Tqdm: For displaying progress bars during data processing.

Dataset:

The dataset used for training and testing the model consists of images categorized into different facial expressions. Each expression category contains grayscale images of faces expressing the corresponding emotion. The dataset is organized into training and testing sets, with each image associated with a label indicating the expressed emotion.

Data Preprocessing:

1. Loading Data: Images and their corresponding labels are loaded from the specified directories (`images/train` for training data and `images/test` for testing data).

2. Image Processing: Images are loaded and converted into grayscale using `load_img` from Keras. Grayscale images are then converted into arrays and normalized to a range of [0, 1] by dividing by 255.

3. Label Encoding: Emotion labels (e.g., 'angry', 'happy') are encoded using `LabelEncoder` from scikit-learn and then converted into categorical format using `to_categorical` from Keras.

Model Architecture:

- Input Layer: Accepts grayscale images of size 48x48 pixels.

-Convolutional Layers: Utilizes multiple convolutional layers with max pooling and dropout to extract spatial features from the input images.

- Flatten Layer: Flattens the output from convolutional layers into a 1D array.

- Fully Connected Layers: Consists of densely connected layers with dropout for feature classification.

- Output Layer: Utilizes a softmax activation function to predict the probability distribution over the seven emotion classes.

Model Training:

The model is trained using the Adam optimizer and categorical cross-entropy loss function. Training data (`x_train` and `y_train`) are used to fit the model over 100 epochs, with a batch size of 128. Validation data (`x_test` and `y_test`) are used to evaluate the model's performance during training.

Model Saving and Loading:

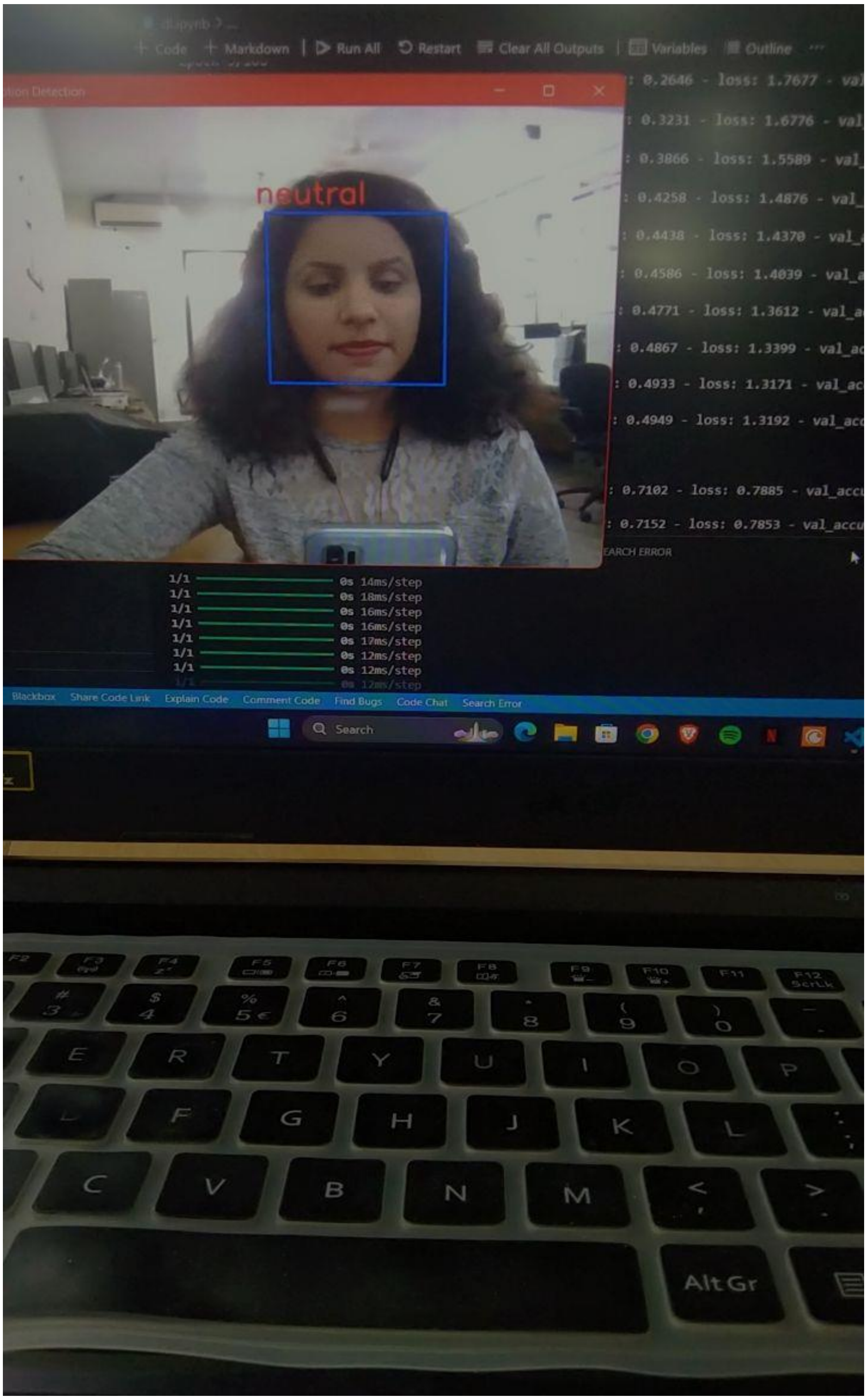
The trained model is saved as both a JSON file (`emotiondetector.json` for architecture) and an HDF5 file (`emotiondetector.h5` for weights). These files are later loaded to make predictions on new images.

Real-time Emotion Detection:

A separate Python script (`python file`) demonstrates real-time emotion detection using the trained model. It uses OpenCV to access the webcam, detects faces using Haar cascades, and then performs emotion recognition on detected faces. The predicted emotion label is displayed on the screen along with a bounding box around the detected face.

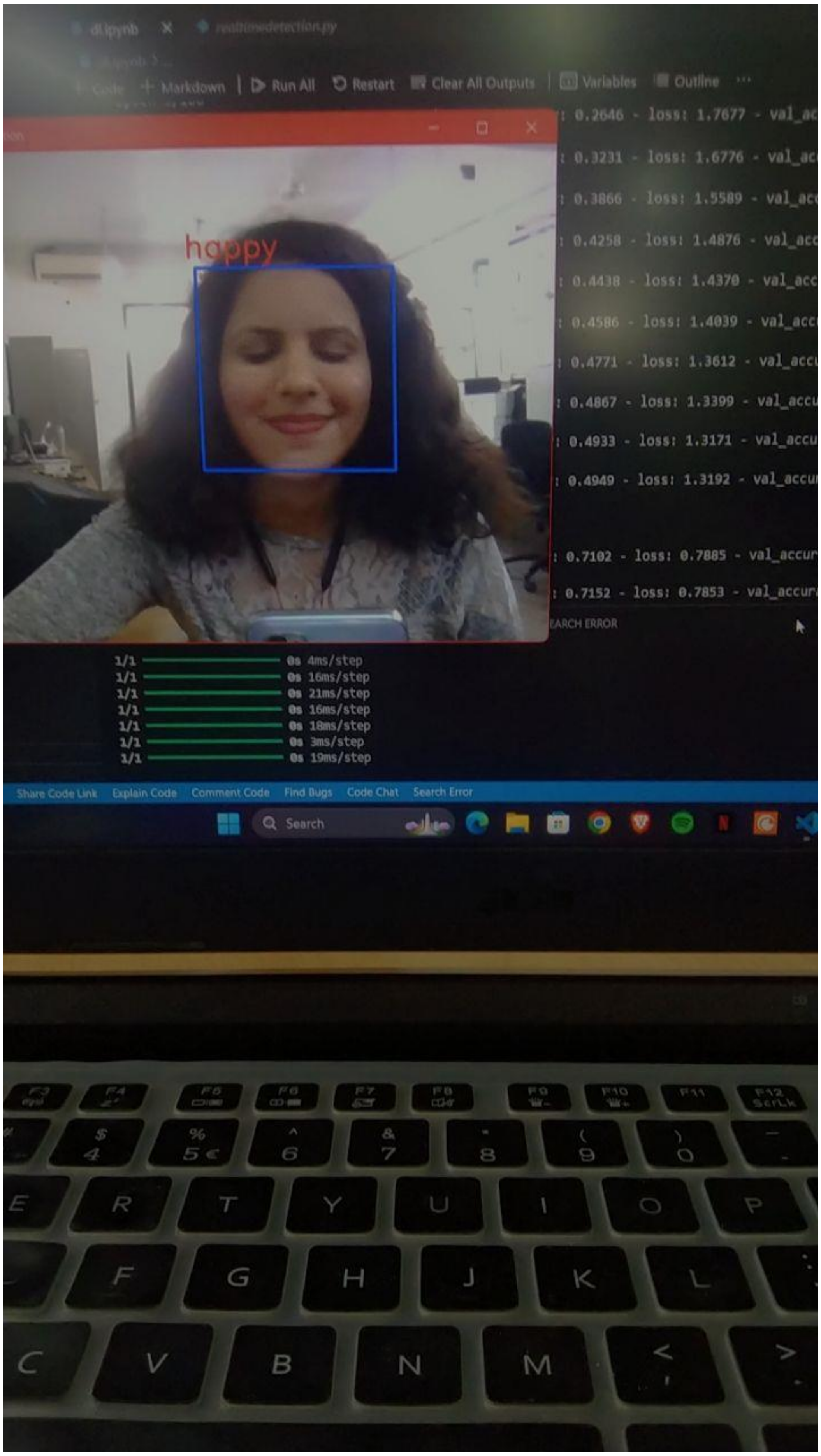
Conclusion:

This project illustrates the process of building an emotion detection system using deep learning and computer vision techniques. The trained model can effectively classify facial expressions in real-time.



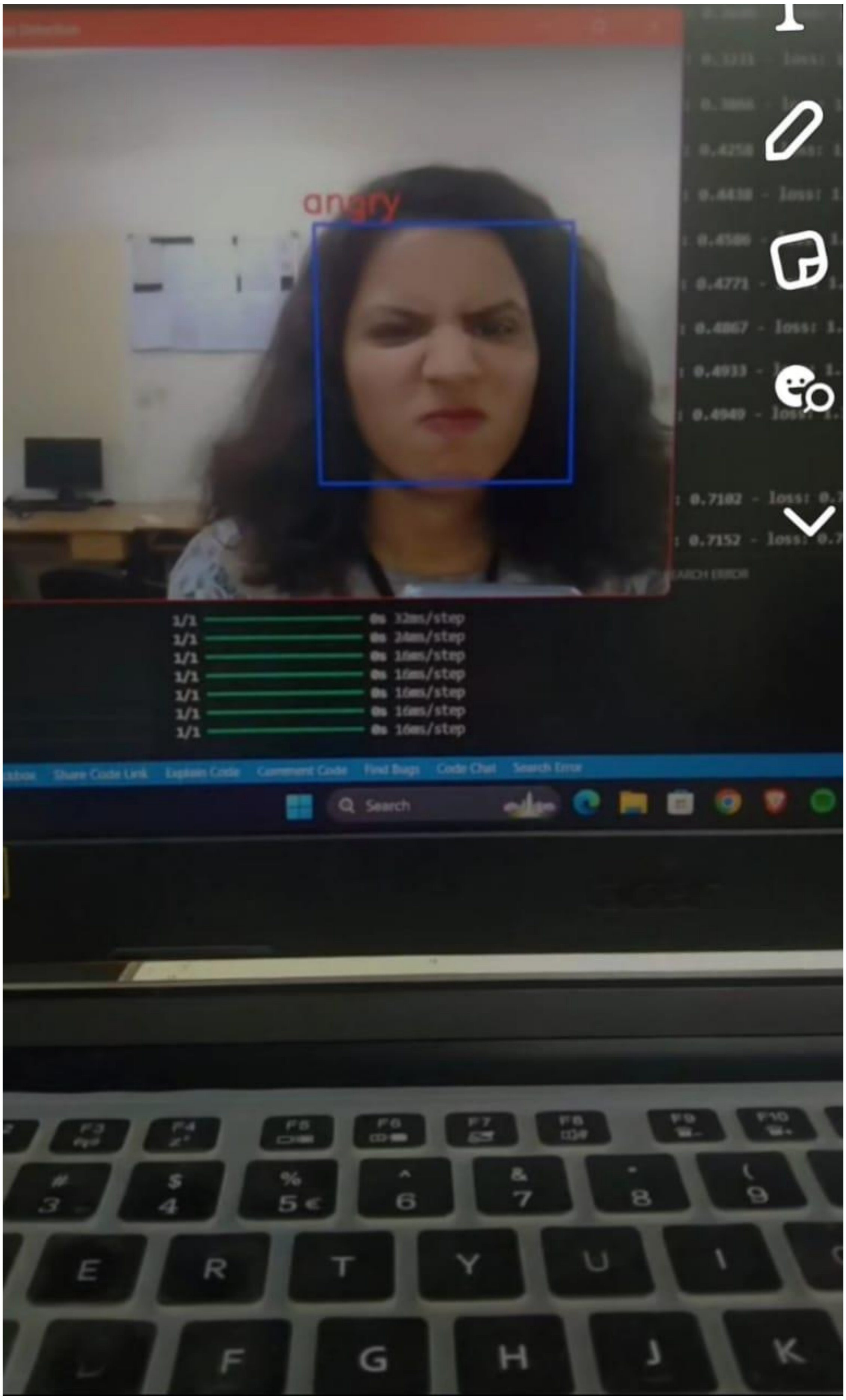
```
: 0.2646 - loss: 1.7677 - val
: 0.3231 - loss: 1.6776 - val_
: 0.3866 - loss: 1.5589 - val_
: 0.4258 - loss: 1.4876 - val_
: 0.4438 - loss: 1.4370 - val_
: 0.4586 - loss: 1.4039 - val_a
: 0.4771 - loss: 1.3612 - val_a
: 0.4867 - loss: 1.3399 - val_ac
: 0.4933 - loss: 1.3171 - val_ac
: 0.4949 - loss: 1.3192 - val_acc

: 0.7102 - loss: 0.7885 - val_accu
: 0.7152 - loss: 0.7853 - val_accu
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step
```



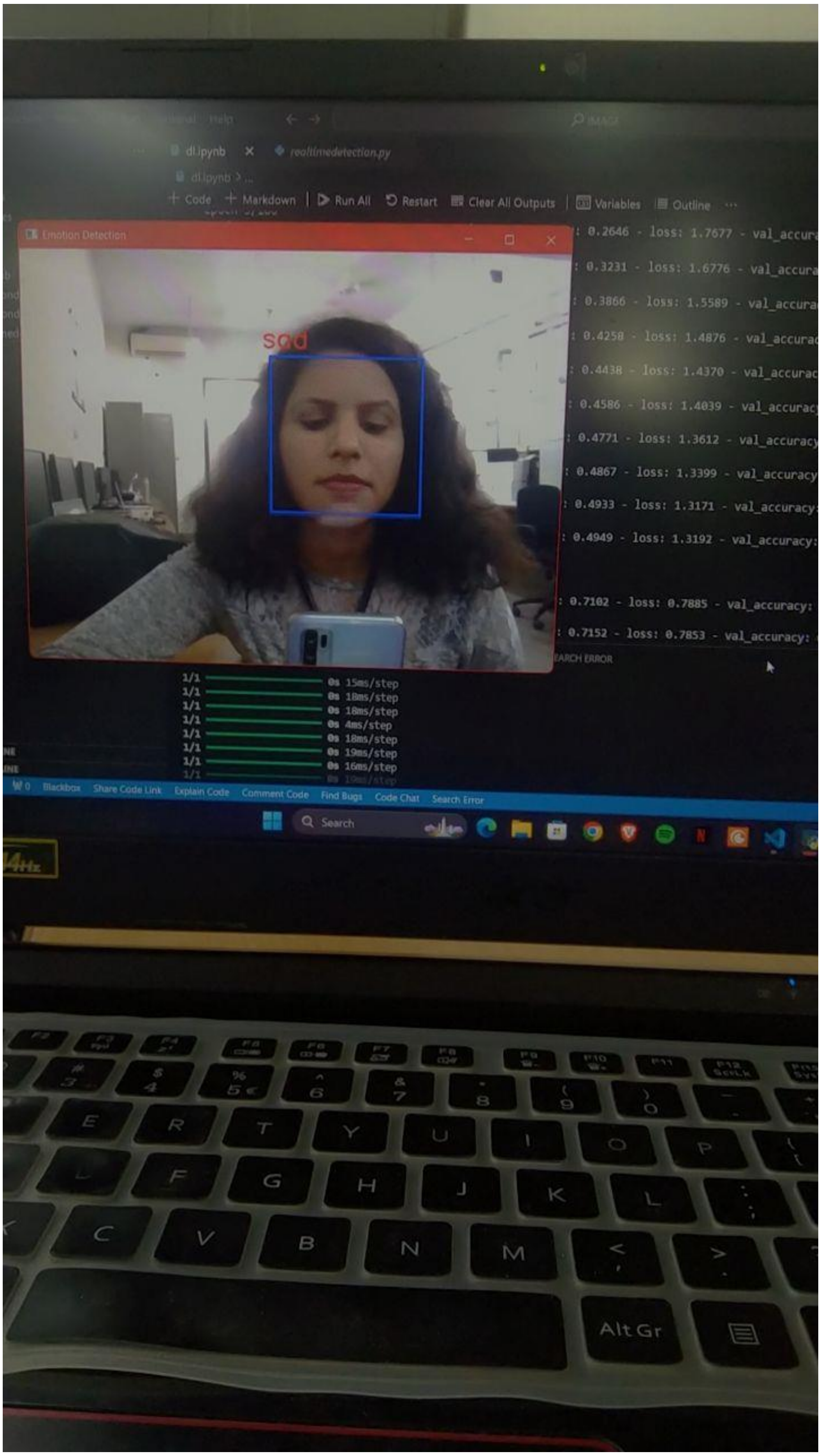
```
: 0.2646 - loss: 1.7677 - val_ac
: 0.3231 - loss: 1.6776 - val_ac
: 0.3866 - loss: 1.5589 - val_acc
: 0.4258 - loss: 1.4876 - val_acc
: 0.4438 - loss: 1.4370 - val_acc
: 0.4586 - loss: 1.4039 - val_acc
: 0.4771 - loss: 1.3612 - val_accu
: 0.4867 - loss: 1.3399 - val_accu
: 0.4933 - loss: 1.3171 - val_accu
: 0.4949 - loss: 1.3192 - val_accu

: 0.7102 - loss: 0.7885 - val_accur
: 0.7152 - loss: 0.7853 - val_accur
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
```



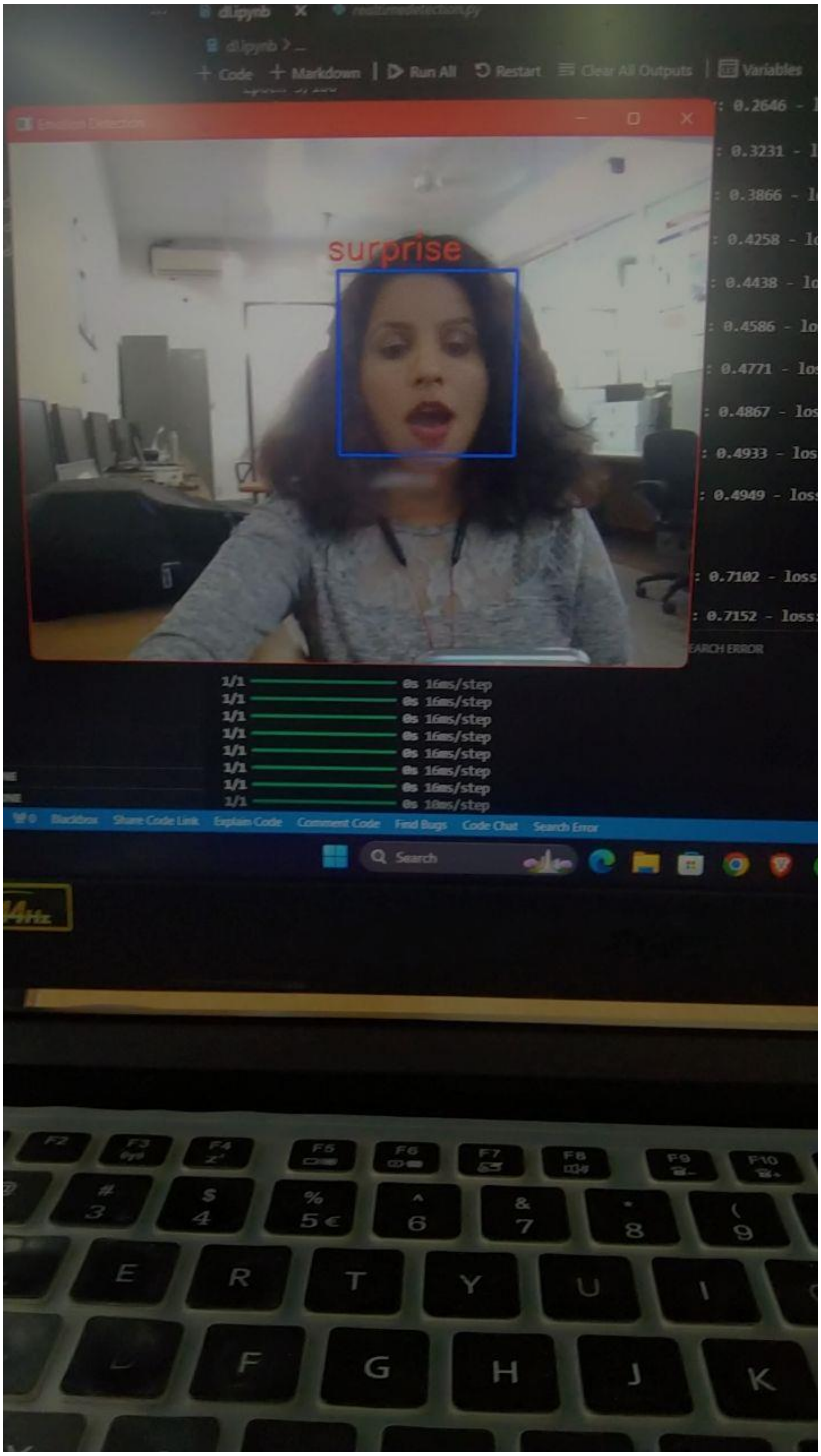
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
```

```
: 0.3211 - loss:
: 0.3866 - loss:
: 0.4258 - loss:
: 0.4438 - loss: 1.
: 0.4586 - loss: 1.
: 0.4771 - loss: 1.
: 0.4867 - loss: 1.
: 0.4933 - loss: 1.
: 0.4949 - loss: 1.

: 0.7102 - loss: 0.
: 0.7152 - loss: 0.
```



```
: 0.2646 - loss: 1.7677 - val_accura
: 0.3231 - loss: 1.6776 - val_accura
: 0.3866 - loss: 1.5589 - val_accura
: 0.4258 - loss: 1.4876 - val_accurac
: 0.4438 - loss: 1.4370 - val_accurac
: 0.4586 - loss: 1.4039 - val_accuracy
: 0.4771 - loss: 1.3612 - val_accuracy
: 0.4867 - loss: 1.3399 - val_accuracy
: 0.4933 - loss: 1.3171 - val_accuracy:
: 0.4949 - loss: 1.3192 - val_accuracy:

: 0.7102 - loss: 0.7885 - val_accuracy:
: 0.7152 - loss: 0.7853 - val_accuracy:
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
```



```
: 0.2646 - l
: 0.3231 - l
: 0.3866 - l
: 0.4258 - l
: 0.4438 - l
: 0.4586 - lo
: 0.4771 - los
: 0.4867 - los
: 0.4933 - los
: 0.4949 - loss
: 0.7102 - loss
: 0.7152 - loss
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
```