IITISoC'21

Team ML7 Presentation:

Domain- ML/AI

Project Topic- Rock, Paper, Scissor Game

Team Member: 1. Kshitij M Bhat (200003042)
2. Bhavya Dalal (200003020)
3. Tanishq Selot (200003076)

Tech Stack used- Python libraries like:

1. Tensorflow
2. Numpy
3. Scikit-learn
4. OpenCV
5. Pygame
6. Matplotlib

The Dataset:

We used the dataset from Tensorflow (https://www.tensorflow.org/datasets/catalog/rock_paper_scissors) for our project.

- This dataset contains images from various hands, from different races, ages, and genders, posed into Rock / Paper or Scissors and labeled as such.
- These images were created using CGI techniques as an experiment in determining if a model trained on a CGI-based dataset could classify real images.
- Each image is 300×300 pixels in 24-bit color. The dataset is licensed as a CC By 2.0, free to share and adapt for all uses, commercial or non-commercial.

Some Examples:

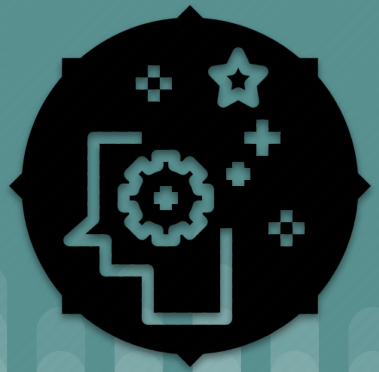
This dataset is very diverse as it contains images from almost all the races, genders and ages. Hence, our model is less biased.

Data Augmentation :

A technique to increase the diversity of the training set by applying random (but realistic) transformations such as image rotation is known as data augmentation.

- We assigned the input size of the images as 300x300x3 (3 for Red, Blue, Green).
- We also applied some random flipping and some random rotations to the images to expand our training set.
- Our motto behind performing this was to prevent overfitting of the model.

Normalization - It was done to change the values of numeric columns in the dataset to use a common scale. Eg. Each RGB channel contains 8 bits, hence we divide by 255 to convert [0,255] to [0,1] bits.



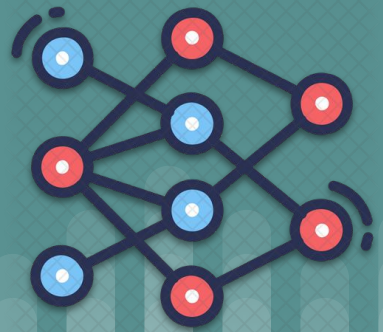
Creating the Model (Using CNNs):

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm.

- **The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction. The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction.**

- **Then we flatten the images before sending them to the multi-level perceptron (Neural Network).**

- **Our model used 5 each of ConvNet and MaxPooling layers.**



Training and testing the model:

- We trained the model using the model.fit() function and tested using model.evaluate() function.
- We had an 80%:20% training:validation data split.
- We performed just 3 epochs i.e; just 53 seconds to train the model and achieved the following accuracies and F1 score:

Training Accuracy - 97.52%

Validation Accuracy - 99.40%

Test Accuracy - 94.35%

F1 score - 94.95%

OpenCV:

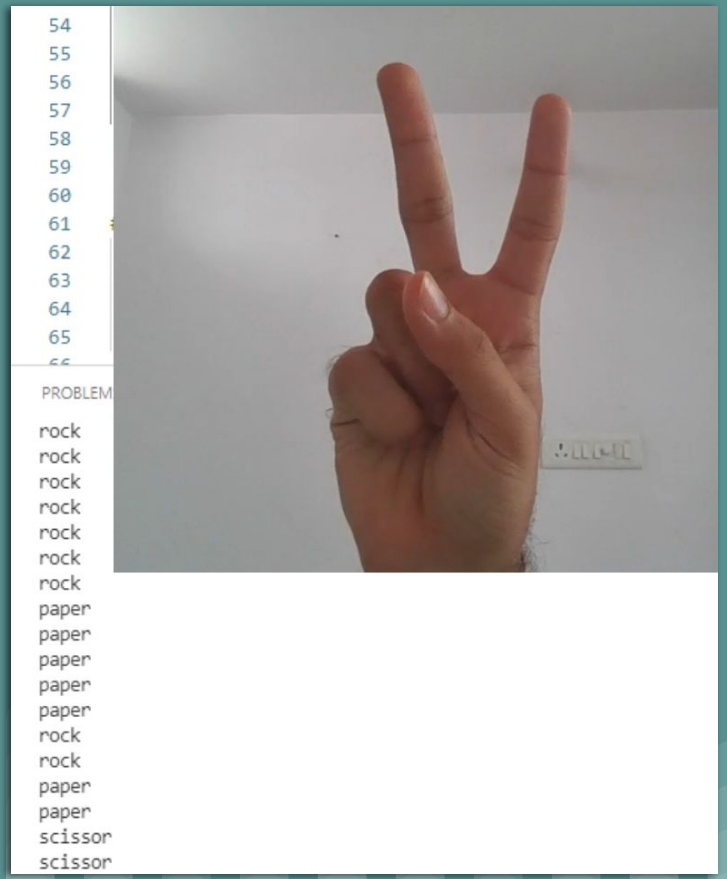
We take the image as an input from the Webcam and feed it our image classifier model for classification and display the result on our pygame interface. OpenCV captures the image and resizes it to feed to our classifier.

- The real-time hand movements' evaluation was introducing a lag in the processing of the image leading to a lag in prediction also.

- Bhavya came up with a solution in the form of the timer method. Here, the machine won't perform continuous evaluation (reason behind the lag).

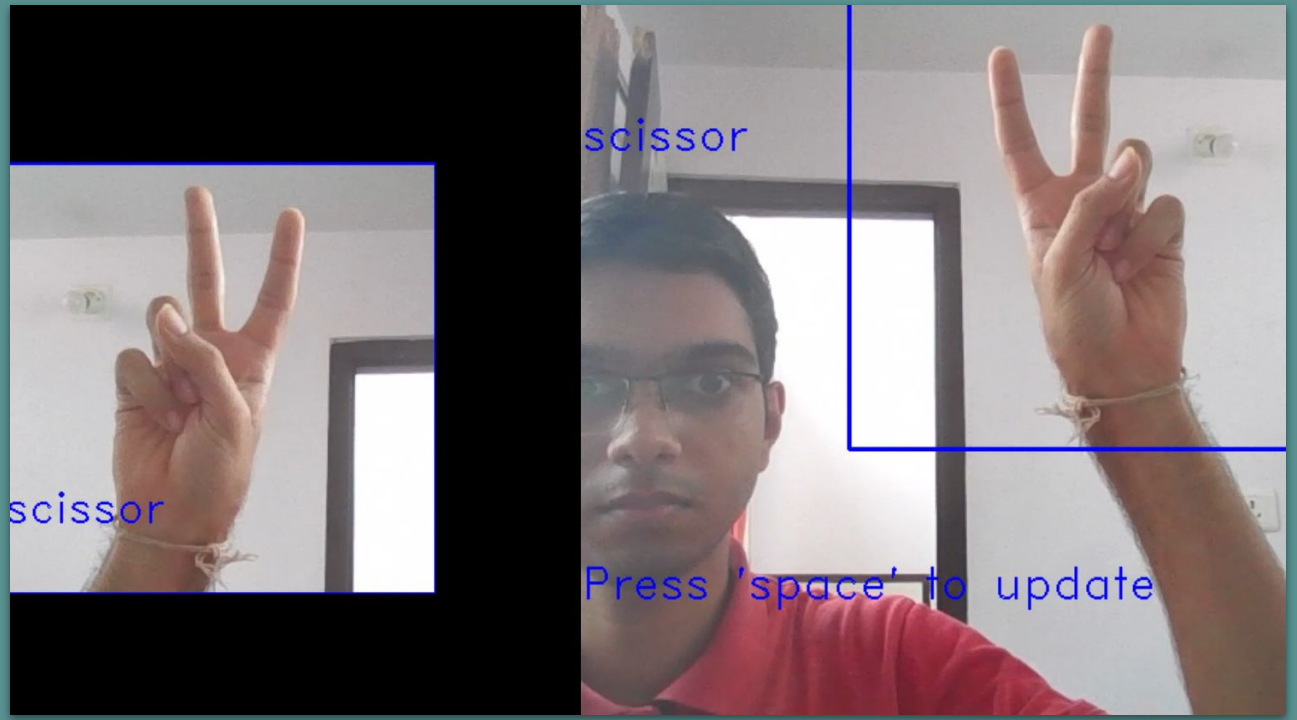
- Instead, when the user presses the spacebar, openCV instructs the webcam to capture an image which is then used for classification.

Comparing the two methods:



- **Continuous evaluation** was the reason behind the lag between input generation-processing and prediction.
- Here, the machine was evaluating even when the hand gesture was changing.
- Here, when the gesture changed from rock to scissor, the model guessed the transition as paper.
- Also, the player was required to keep the hand at the centre for correct predictions.

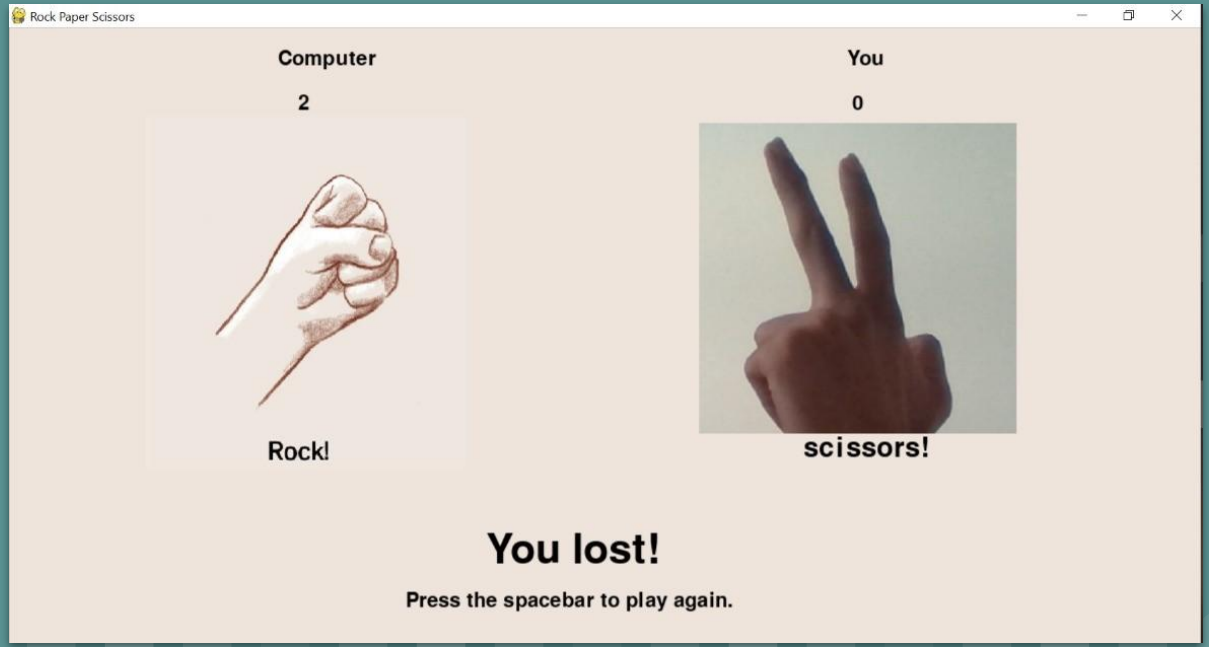
- Almost instantaneous prediction was possible upon pressing the spacebar with the timer method.
- Bounding region fixes the area where the user needs to put his hand.
- Performance and accuracy are enhanced.

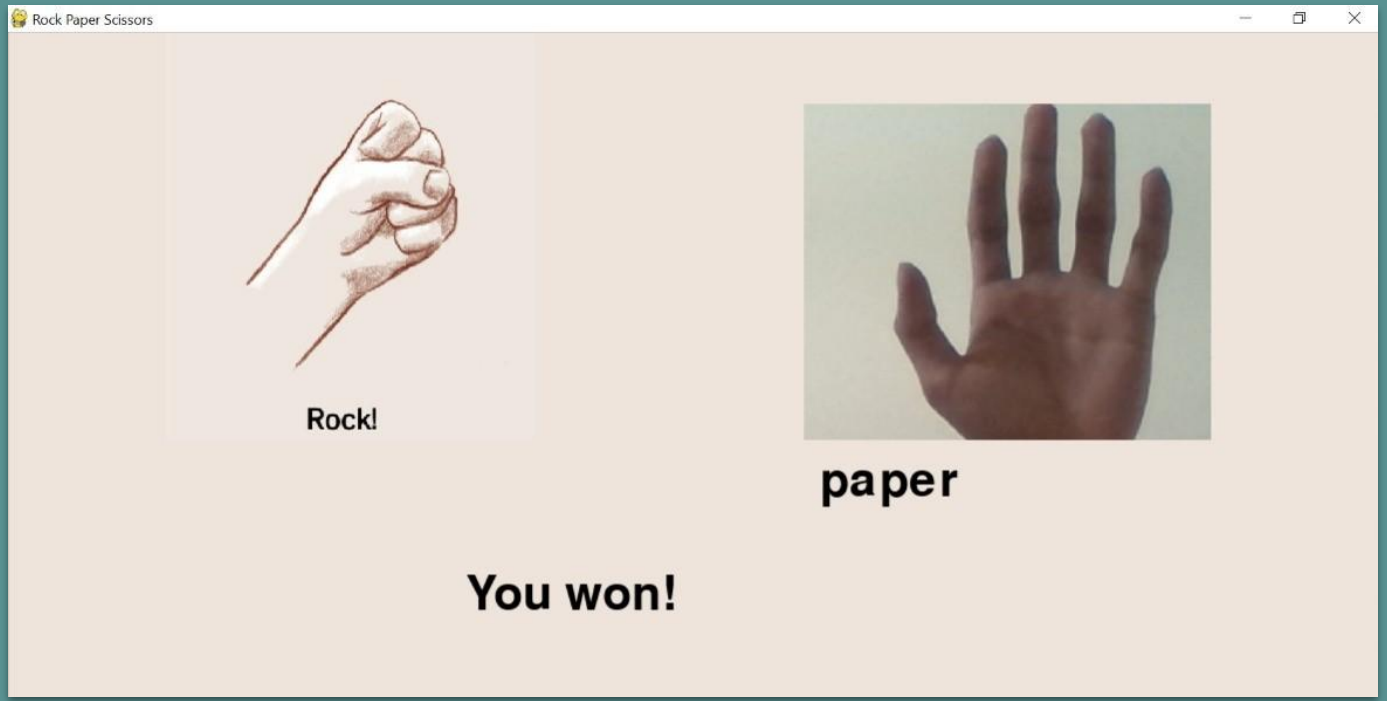


Pygame Interface:

- Pygame is used as the gaming interface for this project.
- It is used to display the responses from the computer and the user.
- The user's responses is taken from the webcam via openCV and is displayed at the interface.
- The computer generates randomly selected image and then the conditional statement is used to predict that who should get the point.
- The interface labels the user and computer responses and also displays the scoreboard at real-time so the user is not required to toggle to another tab for seeing the scores.

- The interface is also re-sizeable according to the user's utility.
- The random image selector for the computer is not at all biased and there is an equal probability of both the sides winning.





Throughout our project, from the dataset to the model, and from the model to the interface, we have tried our best to minimize any bias and to maximize performance and accuracy.

Thank you