
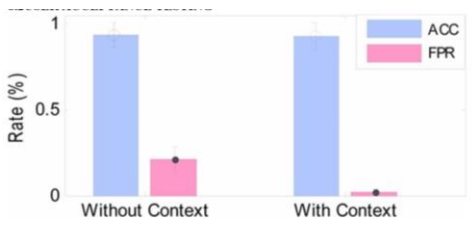



## Project Development Phase Model Performance Test

Date	10 November 2022
Team ID	PNT2022TMID28409
Project Name	A Gesture-Based Tool For Sterile Browsing Of Radiology Images
Maximum Marks	10 Marks

### Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Detects the hand gesture shown by the surgeons during the surgery	 <p>The screenshot shows a Jupyter Notebook cell with the following code: <code>import cv2</code>, <code>cat = cv2.imread('c:\Users\VERSHIVA\Dataset\data\five\hand1(984).jpg')</code>, and <code>plt.imshow(cv2.cvtColor(cat, cv2.COLOR_BGR2RGB))</code>. The output shows a grayscale image of a hand with fingers spread, overlaid on a black background.</p>
	Accuracy	Training Accuracy – 92%  Validation Accuracy -89%	 <p>The bar chart compares the Accuracy (ACC) and False Positive Rate (FPR) for two scenarios: 'Without Context' and 'With Context'. The Y-axis represents the Rate (%), ranging from 0 to 1. For 'Without Context', the ACC is approximately 0.92 and the FPR is approximately 0.25. For 'With Context', the ACC is approximately 0.89 and the FPR is approximately 0.05.</p> <pre> def getPredictedClass(model):     image = cv2.imread('temp.png')     gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)     gray_image = cv2.resize(gray_image, [100, 120])     gray_image = gray_image.reshape(1, 100, 120, 1)     prediction = model.predict_on_batch(gray_image)     predicted_class = np.argmax(prediction)     if predicted_class == 0:         return "Blank"     elif predicted_class == 1:         return "One"     elif predicted_class == 2:         return "Thumb Up"     elif predicted_class == 3:         return "Thumb Down"     elif predicted_class == 4:         return "Punch"     elif predicted_class == 5:         return "High Five"     return "High Five" if __name__ == "__main__":     accumulator = 0.5     camera = cv2.VideoCapture(0)     fps = int(camera.get(cv2.CAP_PROP_FPS))     top, right, bottom, left = 10, 350, 225, 500     num_frames = 0         </pre>

			<pre># first conv layer # input shape = (img_rows, img_cols, 1) model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(100,100, 1))) model.add(BatchNormalization()) model.add(MaxPooling2D(pool_size=(2, 2))) model.add(Dropout(0.25))  # second conv layer model.add(Conv2D(64, kernel_size=(3, 3), activation='relu')) model.add(BatchNormalization()) model.add(MaxPooling2D(pool_size=(2, 2))) model.add(Dropout(0.25))  # flatten and put a fully connected layer model.add(Flatten()) model.add(Dense(128, activation='relu')) # fully connected model.add(Dropout(0.5))  # softmax layer model.add(Dense(6, activation='softmax'))  # model summary optimizer = Adam() #write your optimizer model.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['categorical_accuracy']) model.summary()</pre> 
3.	Confidence Score	Class Detected -  Confidence Score -	NA