Intelligent Interactive Systems (1MD032) Ash: The Hospital Receptionist Report for Project by Team 23

Venkata Sai Teja Mogillapalle Dhanush Kumar Akunuri
Naveen Pedhiboiena Shreyas Shivakumara

June 1, 2020

1. Abstraction

During this pandemic situation(Covid-19), the need for continuing the hospital services by reducing the contact with humans is given top most importance. Applications like Furhat becomes increasingly useful in this case. Virtual Furhat application helps the humans to contact with our receptionist(Furhat) whitout even touching the screen just by using hand gestures for communication.

2. Introduction

We demonstrate an application system to book an appointment using the virtual version of Furhat. Furhat interacts with virtual patients through hand gestures in the video. We use an OpenCV model to detect the key points in the videos and recognize the hand gesture with the detected key points using a machine learning model multi-layer perceptron(MLP). We then describe how we can integrate these models with Furhat using Kotlin. At last, we propose an end-to-end integration learning system to implement this task.

3. Landmark Extraction

3.1. Methodology

This section describes about the methods used to extract landmarks from the hand. We have combined a custom CNN model and a model for detecting key points on the hand as presented in [1]

1. Design
   This project is divided into 2 sub-systems. First, we identify the hand-

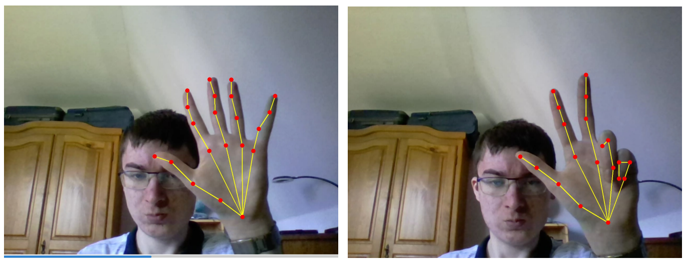
side(palm or dorsal).We developed a CNN model for achieving this.Next, we use the prediction and give it along with images to our next part, which will output the coordinates of landmarks in the image.

2. Materials
   For this subsystem, we used python programming language and implemented it in Jupyter Notebook which is an interactive web application for python. Learn-opencv provides source code for detecting the hand gestures from the hand image inputs.

3. Procedure
   The first sub-part of this subsystem detects the side of the hand(palm or dorsal). We train a CNN network to perform this task.

3.2. Results

The results obtained in the subsystem are seen in 1a and 1b . We got an accuracy of 100% on identifying the hand-side using the CNN Model.



(a) Result on Open-Palm gesture (b) Result on Three-Fingers Palm

Figure 1: Results of Landmark Extraction

3.3. Discussion

Our model is not successful when the image consists of coordinates of both palm and dorsal side in the same hand gesture pose. 1b (3 fingers palm). And also the algorithm sometimes detects the points which are not visible on the screen.

4. Gesture Recognition

4.1. Methodology

This section provides the design and result of detecting the hand gesture. The approach is done by implementing Multilayer Peceptron(MLP). The MLP is a class of feedforward artificial neural network (ANN). The inputs to the MLP are the coordinates that must have been generated from the first sub-system. But we have used the coordinate information provided to us for training our MLP which outputs the hand pose.

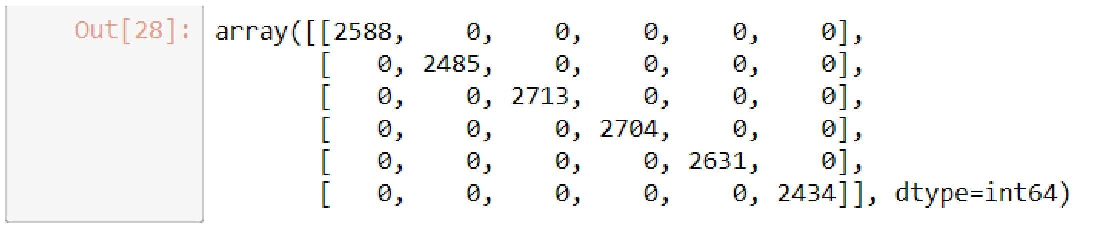
1. Design
   The MLP architecture we used for this sub-system consists of a input layer, hidden layer and an output layer. The size of the input layer is the number of input columns we pass to the network. By default, the size of the hidden layer is 100 and the in this case, we have output of size 6 as we would like to classify our inputs into 6 classes.

2. Materials
   To implement this subsystem, we implemented python code in Jupyter Notebook which is an interactive web application for python. We have used Scikit package in python for implementing MLP.

3. Procedure
   The input data for the network is the annotations of hand signals that are columns from 2 to 82 in the CSV file and its labels produced by combining two rows namely "camera face" and "gesture".The dataset is divided into training(46665 data points) and testing(15555 data points). We shuffled the data before training to remove any bias in it and to prevent the model from learning only from the order in which we train. The model outputs the hand gesture.

4.2. Results

Overall, we got a result of 100% accuracy on the testing data as shown in the figure 2 using the Multilayer Perceptron Classifier.



```
Out[28]: array([[2588,    0,    0,    0,    0,    0],
               [   0, 2485,    0,    0,    0,    0],
               [   0,    0, 2713,    0,    0,    0],
               [   0,    0,    0, 2704,    0,    0],
               [   0,    0,    0,    0, 2631,    0],
               [   0,    0,    0,    0,    0, 2434]], dtype=int64)
```

Figure 2: Confusion Matrix Result

4.3. Discussion

It is interesting to know that we obtain a 100% accuracy by training a simple MLP classifier. A possible explanation for this result might be due to absense of noise and variations in the dataset.

5. Responsive Behavior

5.1. Methodology

This section describes the human-robot social application that collects COVID-19 patient's information and books an appointment based on this information in the hospital."Ash" is the name of the virtual Furhat robot, a receptionist in the Apollo hospital.

1. Design
   (a) A virtual user enters the interaction space to start the interaction.
   (b) The interaction begins with Furhat attending to and greeting the user and asks the user if the user has come for a coronavirus check-up.
   (c) Furhat displays a list of symptoms of COVID-19 and lets the user select from the list.
   (d) The user must select the severity for each of the symptoms between severe and mild.
   (e) Based on the symptoms information and its severity, the Furhat creates an appointment between the emergency ward or the general ward.
   (f) The interaction ends with Furhat thanking the user and then waiting for another user.
2. Materials
   This subsystem is programmed using Kotlin. It is compiled and run on IntelliJ IDEA software, that is an environment to work with Kotlin.
3. Procedure
   The main.kt contains the main class of the application containing the skills. We created ten different handles for a basic user management during an interaction allowing the robot to behave in a socially appropriate manner as users come and go from the interaction state. Table 1 refers to the gestures along with their names.

Gesture	Meaning
Open palm	Yes
Open dorsal	No
Fist dorsal	Mild
Fist palm	Severe
Three fingers palm	Next
Three fingers dorsal	Back

Table 1: Table illustrating the gestures with their meaning

5.2. Results

Below is an example dialogue between Ash(A) and a virtual patient(P) in the system.
A: *Welcome to Apollo Hospital! My name is Ash and I am here to assist you. Are you here for Corona Virus check up?*
P: *Open palm*
A: *We are going to do a symptoms check up. Please answer the following question. Do you have any of the following symptoms? Fever,Cough,Body Ache*
A: *You are currently pointed at fever. Say yes for selecting or next to select other symptoms from the list.*
P: *Open palm*
A: *How is the severity of Fever between Severe and mild?*
P: *Fist Palm*
A: *Are you facing any other symptoms from the list?*
P: *Open Palm*
A: *You are currently pointed at Cough. Say yes for selecting or next to select other symptoms*
P: *Three Fingers palm*
A: *You are currently pointed at Body ache. Say yes for selecting or next for the next step*
P: *Open palm*
A: *How is the severity of body ache between Severe and mild?*
P: *Fist Palm*
A: *Are you facing any other symptoms from the list?*
P: *Open dorsal*
A: *Please go to the emergency ward for treatment.Thank you!*

5.3. Discussion

This subsystem provided us the opportunity to build an application for a Furhat robot that records the patient symptoms and books an appointment in a hospital. In accordance to present input, It would be very challenging to build a Furhat robot that takes as input the gesture from the videos and shows the response of a virtual agent to that gesture.

6. Specialisation

6.1. Methodology

This section provides an end-to-end learning model for detecting the hand gestures instead of two subsystems as we described above. We have used a CNN model for creating this end-to-end learning model.

1. Design
   We have designed a CNN model for this problem. The input is image which is resized according the input size and it has three conv 2D layers , three max-pooling layers, one flattened function which is fully connected . We have using "adam" optimizer and "cross-entropy" loss function.

2. Materials
   For this subsystem, we used python programming language and implemented it in Jupyter Notebook which is an interactive web application for python. Tensorflow and Keras packages are used for developing the CNN network.

3. Procedure
   The video is breakdown into images from the video frames and given as input to the CNN network after resizing. Our CNN network is trained with the images and the corresponding hand gesture labels. The output of the model is the predicted hand-gesture for the input image.

6.2. Results

We have around 12000 images collected from the video frames. After training the model with 9000 images, our model was able to predict all the images in the test dataset correctly. We got 100% accuracy. Discussion As there is no much noise in the data. all the videos were recorded with almost no background disturbances and there is not much variation in the data, our model was able to run with 100% accuracy even after just few epochs(5 in our case).

6.3. Challenges faced (what has worked and what has not)

Our model fails to categorise the landmarks of a single hand pose into both palm and dorsal.For example: Three fingers palm that consist of both palm and dorsal key points. The integrating the three and two subsystem was also challenging for us as we had to connect the Furhat application with the TCP connection and send the output of subsystem 2 to subsystem 3 which we couldn't accomplish.

7. Conclusions and Discussions

We developed the solutions for each sub system separately along with an end-to-end learning model. Our end-to-end learning model was showing excellent results with 100% accuracy. We faced a lot some issues with the sub-system 1 and also we didn't connect each sub-system to make it a complete application. This entire application with the sub-systems pipelined would be a very useful application especially in the pandemic situations where human interactions should be limited.

References

[1] Tomas Simon et al. *Hand Keypoint Detection in Single Images using Multiview Bootstrapping*. 2017. arXiv: `1704.07809 [cs.CV]`.