ECE 510 Robotics 3 Final Report

DIM Robot

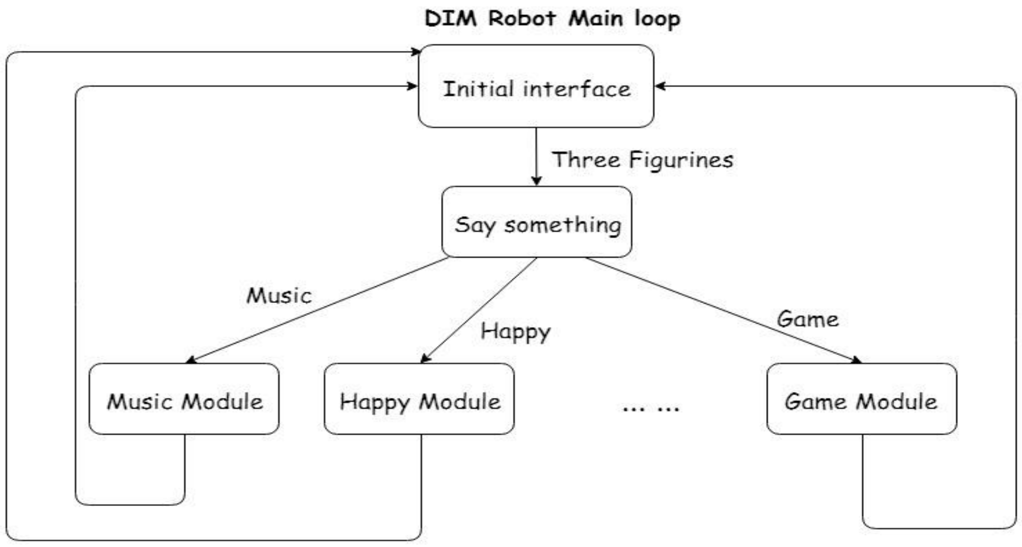
Ming Ma
Ting Wang
Xiaoqiao Mu

6/12/2019

Introduction:

For our final project, we work on the DIM robot which can interact with people. The original DIM robot can tell some jokes or say some facts about himself. Also, he can drum for people if people say “music” and he can show different emotions by changing his face. However, this is still not very interesting. So, we add a simple game: rock, paper, and scissors. So, people can say “game” to play this game with DIM.

How to play the game:



Our program is integrated into the previous program. So, DIM maintains the original functionalities. In order to play our game, the first thing you need to do is have your 3 fingers appear in the recognition box. Then, the led in the neck of the robot will become green. At this time, people can say “game” and our program is executed if you can hear “Let’s play a game!”. If DIM cannot recognize your voice, you can try more times to make it work.

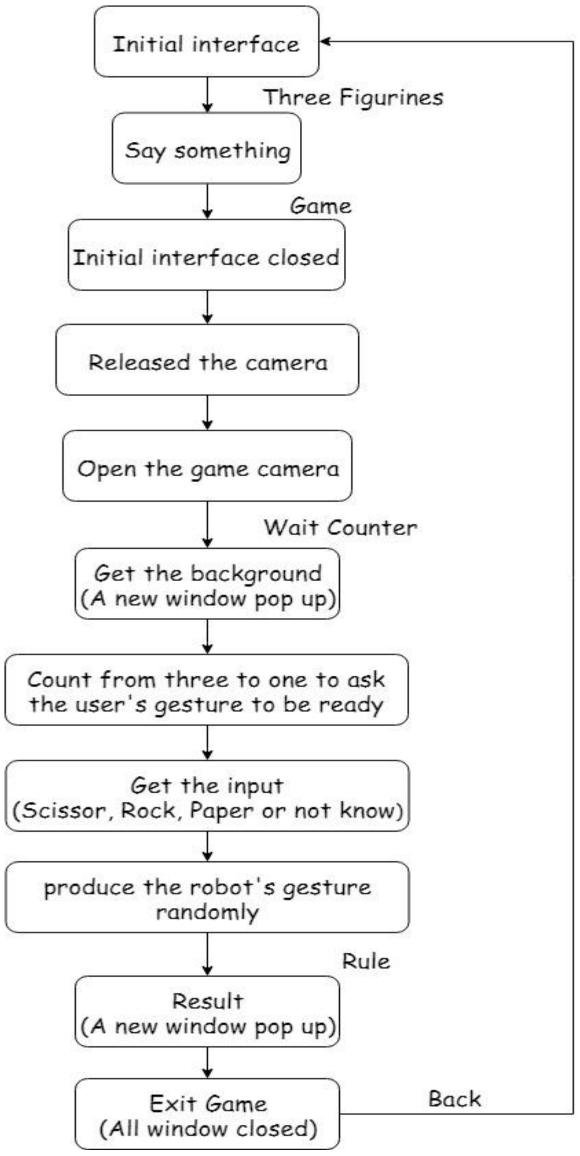
Then, DIM will draw a new recognition box and it will automatically capture the background after some seconds. Make sure there are no people show in the box. Otherwise, it will produce some additional noise which will affect the accuracy of recognition.

After this, DIM will say “Tree, Two, One, Start”. After saying start, people can show their gestures in the recognition box and DIM will give some random gestures among rock, paper, and scissors. The final result will be told by DIM. DIM will say different words and drum in different ways according to different results. Also, people can also see the final result in the terminal.

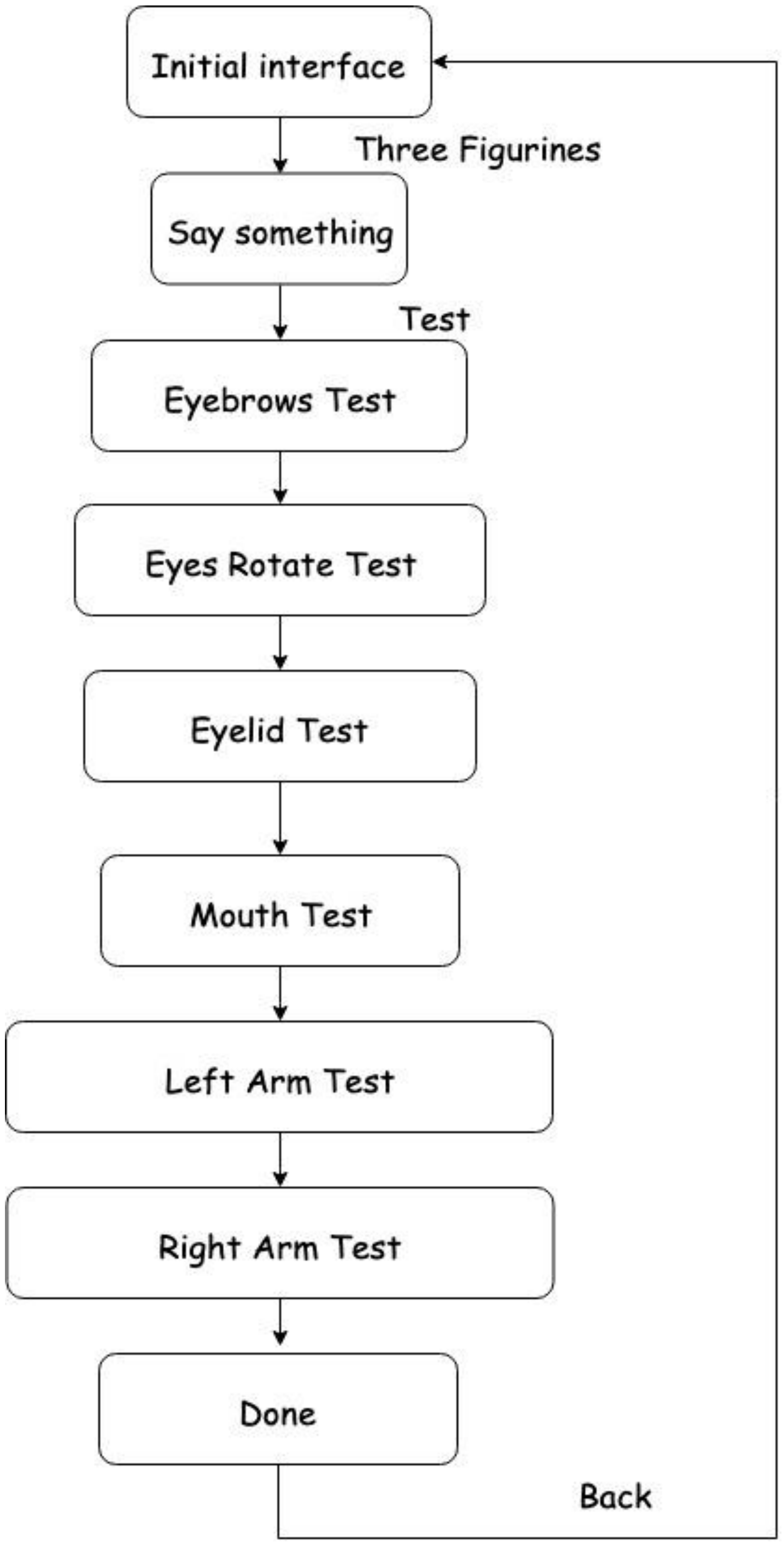
This is one round. After DIM shows the game result in the terminal, it will go back to the main program again. So, people can do the previous steps again to play more round. You can check the game flowchart for more details. Try your best to win more!

Another thing we want to mention is that we have the “test” command. User can say “test” when the led in the neck of DIM becomes green. DIM will check all servos included in his body sequentially. DIM will say which parts of the body is testing. At this time, some parts are broken if you cannot see any changes in the corresponding part. He will say “I am done!” when the testing process is finished. And it will return to the main program also.

Game Flowchart:


Initial interface
Three Figurines
Say something
Game
Initial interface closed
Released the camera
Open the game camera
Wait Counter
Get the background (A new window pop up)
Count from three to one to ask the user's gesture to be ready
Get the input (Scissor, Rock, Paper or not know)
produce the robot's gesture randomly
Rule
Result (A new window pop up)
Exit Game (All window closed)
Back

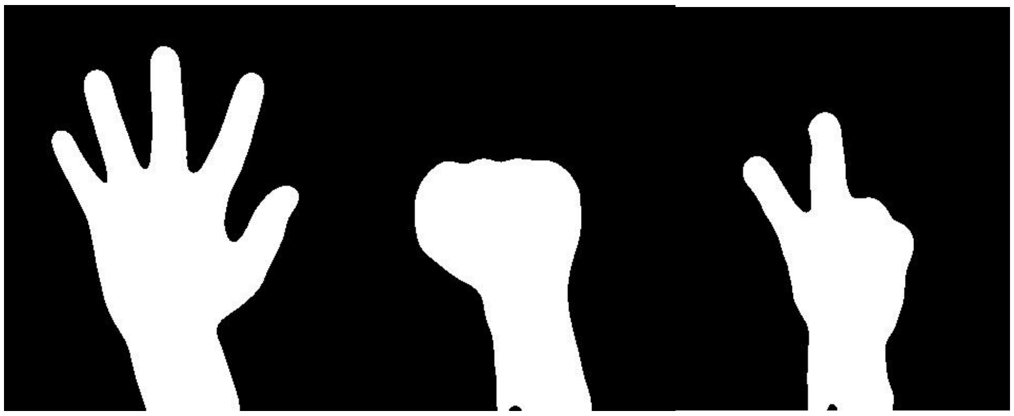
Test Flowchart:



Algorithm Introduction:

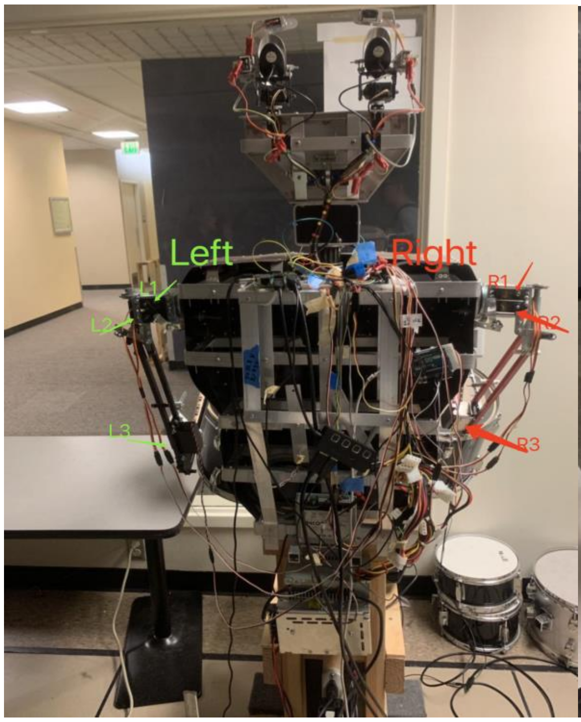
The camera in the DIM is controlled by using openCV. It's easy to draw a capture box which is the area we want to focus on. Our program will capture the background in the capture box and then do the pixel xor to eliminate the background. So, we can neglect the influence of different backgrounds. Anything other than captured background will show in the capture box (our gestures). The processing images are binary images which only contain our gestures. Thus, our input images are ready to be recognized.

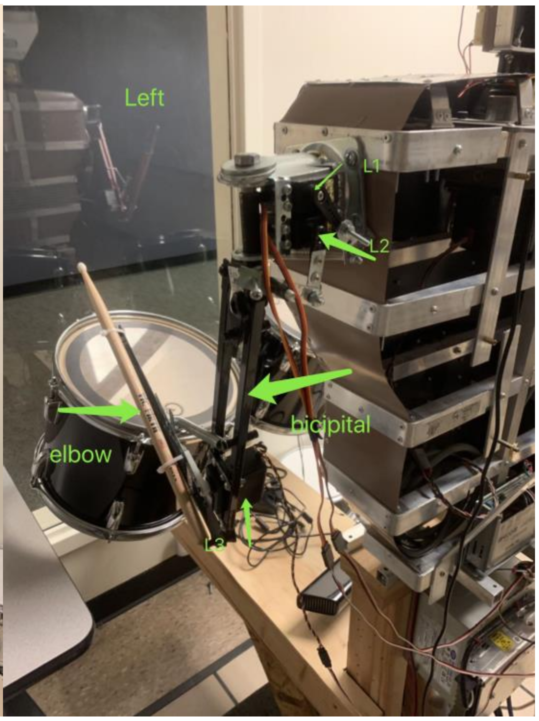
To train our model, we use tensorflow which contains some built-in machine learning models. We choose to use a sequential model to build a convolutional neural network. Our model only contains two layers. The activation function for the first layer is relu. The **relu**(Rectified Linear Unit) is a piecewise linear function that will output the input directly if is positive, otherwise, it will output zero. The activation function for the second layer is softmax. The softmax function is often used in the final layer of a neural network-based classifier. Such networks are commonly trained under a log loss (or cross-entropy) regime, giving a non-linear variant of multinomial logistic regression (from Wikipedia).

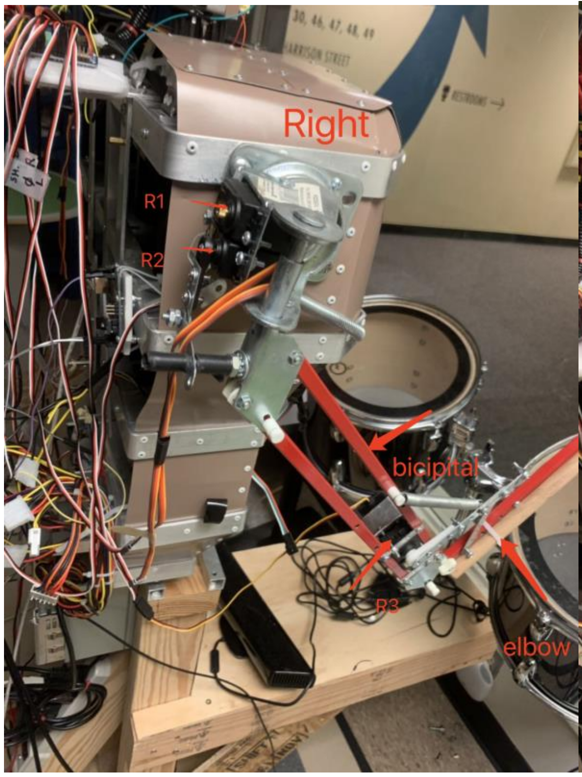


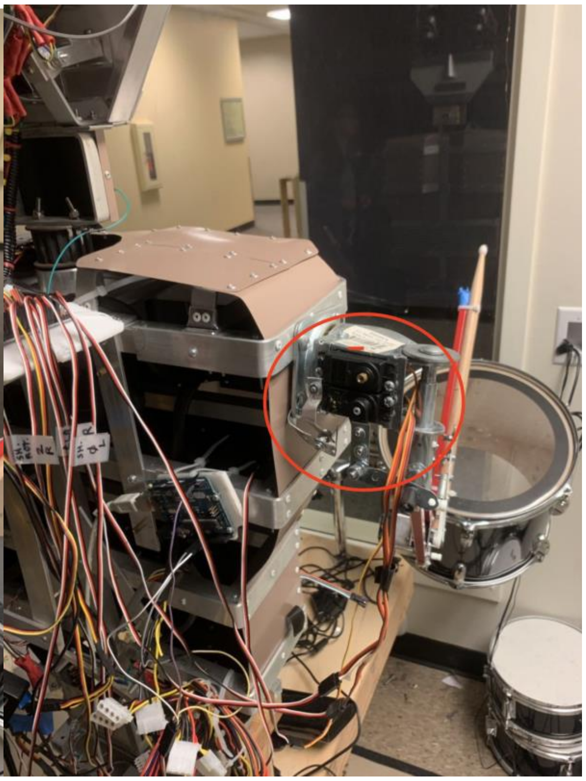
The training and testing data are similar pictures as shown above. For each gesture, we include 1000 pictures for training and 100 pictures for testing. The trained model is saved as an h5 file. Our program will load the model and make some predictions on the captured images. The recognized gestures, game result, and accuracy of recognition are drawn in a new window by using openCV. For more details about our model and training process, you can refer to our homework 2 report.

Servo and Test:



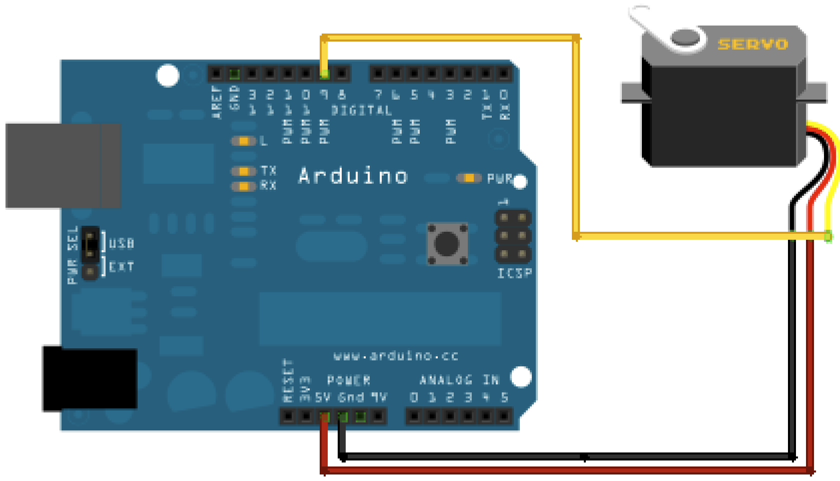






When we test the movements of the robot, we found that some movements cannot be realized. We think that some servos might be broken. In order to find the broken servos, we test all the servos which control the robot's arms.

There are six servos to control the robot's two arms. In the robot's left arm, L1 is controlling the robot to rotate his left bicipital up and down. L2 is controlling the robot to rotate his left arm outside or inside. L3 is controlling the robot to rotate his left elbow up and down. R1 is controlling the robot to rotate his right bicipital up and down. R2 is controlling the robot to rotate his right arm outside or inside. R3 is controlling the robot to rotate his right elbow up and down.



We use Arduino to test our servos. The signal wire of the servo is connected to pin 9. We use the power source to provide power (5V, 1.5A) for the servo. One thing we want to mention is that we need to connect Arduino ground and servo ground to the ground of power source as common ground.

After we tested all the servos, we found that there are three servos broken. In the robot's left part, the L1, L2 servos are totally broken. The L3 servo is half broken. It means that the L3 servo is able to up without connecting to its left elbow. It seems that the L3 servo doesn't have enough power. In the robot's right part, the R1, R2 servos are working. The R3 servo is broken. Even though the R1 servo works, the robot's right bicipital cannot rotate up and down. The reason is that there is a mechanical part missing (red circle). The R3 servo is totally broken. The robot's right elbow cannot rotate up and down. We have replaced the R3 servo. After we replaced the R3 servo, the robot's right elbow could rotate up and down.

Finally, here is the table:

Left Arm		Right Arm	
L1	Not work	R1	Work
L2	Not work	R2	Work
L3	Work but doesn't have enough power	R3	Work

We also have learned the difference between servos and motors.

The motor has two wires to control the rotating. These two wires are power and ground. The motor will start rotating when the motor connects to the power. The motor will stop until that power is detached. Most of the motors run at high revolutions per minute (RPM). The speed of the motor can be controlled by using PWM (pulse width modulation) technique, a technique of fast pulsing the power ON & OFF.

The servo includes a DC motor, a control circuit, a gearing set, and a position sensor. The motor has three wires to control the rotating. These three wires are control, power, and ground. Comparing with the DC motors, the servos could be controlled more precisely. The servo could move the   arm within a certain range.

Mechanical Control

After changing any available servos, we test every servo's working degree. When the frequency of the system is 50Hz, the working degrees for every servo is as the below table.

Body	Channel	Position	Degree
Right Eyebrow	Channel 0	\	80
		--	120
		/	140
Left Eyebrow	Channel 1	\	140
		--	120

		/	80
Right Eyelid	Channel 2	Open	90
		Close	150
Left Eyelid	Channel 3	Open	120
		Close	60
Eye Horizontal	Channel 4	Left	80
		Center	110
		Right	160
Eye Vertical	Channel 5	Up	80
		Center	100
		Down	120
Mouth	Channel 6	Open	60
		Close	0
Right Shoulder joint	Channel 7	Rotate	The servo work but it can not make the shoulder move
Left Shoulder joint	Channel 8	Rotate	Broken
Right Arm_side ways	Channel 9	Outside (Right)	90

		Inside (Left)	120
Left Arm_side ways	Channel 10	Outside (Left)	Broken
		Inside (Right)	
Right Elbow	Channel 11	High	150
		Low	170
Left Elbow	Channel 12	High	Broken
		Low	

At the end of the game, the robot should do some action when he speaks out his gesture. Our setting is below:

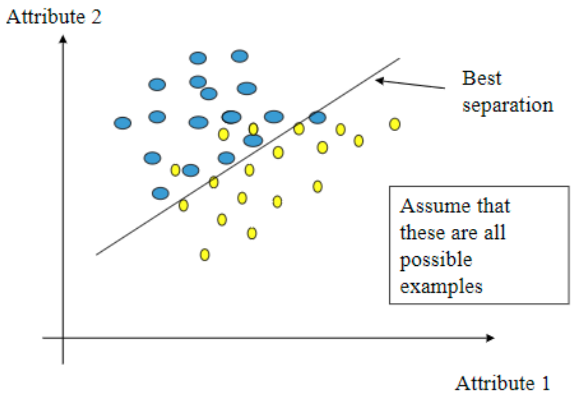
Paper	Beat the upside of drum once
Scissor	Beat the upside of drum twice
Rock	Beat the right side of drum once

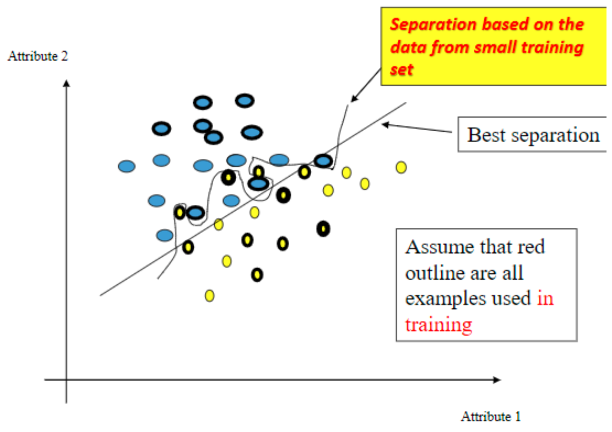
One improvement in the expression aspect of Robot is that the robot can speak more like a human. In the past, the Robot opens his mouth before he speeches words and close his mouth after he completes speaking. We use a library named 'threading', which can make two tasks proceed at the same time, to achieve that robot can repeat opening his mouth and closing his mouth while he is speaking.

Knowledge Review

1. CNNs are regularized versions of multilayer perceptions. Multilayer perceptron usually refer to fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. However, CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme (from Wiki).

2. Overfitting is a modeling error which occurs when a function is too closely fit a limited set of data points. Overfitting the model generally takes the form of making an overly complex model to explain idiosyncrasies in the data under study (from Wiki).

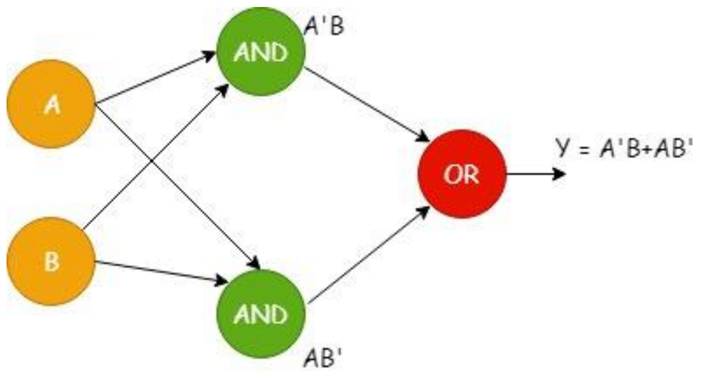




Separation based on the data from small training set
Attribute 2
Best separation
Assume that red outline are all examples used in training
Attribute 1

3. The xor function cannot be achieved by using one layer neural network.

$$A \oplus B = A'B + AB'$$



Troubles and Solutions

1. The first issue with our project is that the accuracy of the prediction is always 100%. This accuracy only changes when there are no objects in the background. The possible reason for this is we train our model by using binary images. There is no background in binary images. So, the prediction is accurate for some degree. Another possible reason is that our model is simple. So, the recognition is not very complex and the prediction is certain in some ways. For our project, we don't plan to reorganize or re-train our model because the prediction is good. Next group can decide to train a more complex model to make the prediction accuracy work.
2. The second issue is that there is a long processing time between receiving game command from users (speech) and open a new window to play the game. DIM does nothing during this time. That may make users doubt whether the robot is working. Therefore, we make the robot wink his both eyes during the conversion process to let users know he is processing.
3. We test the servos by using Arduino UNO, but the whole robot system is controlled by the Raspberry Pi. The frequency is different. Therefore, we tested the movement of robot with small degree.

Improvement and Future Work for Next Team

1. Fixed the mechanical missing part.
2. Replace all the broken or half broken servos
3. Add more modules such as speech recognition
4. Try to integrate another board to increase the speed of processing.
5. Make the robot can rotate his head.
6. Fix the display issue with the prediction accuracy or train more complex model.
7. Add more interesting games.

Summary

In this project, we have learned:

1. Integrate Python code
2. The different between servos and motors. Also, how to test servo by using Arduino.
3. How to do some simple image processing and configure cameras by using openCV.
4. How to train a convolutional neural network by using Tensorflow and Keras.
5. Some basic machine learning algorithms such as CNN, SVM, Tree and so on.

Reference:

1. https://www.arduino.cc/en/tutorial/sweep
2. https://en.wikipedia.org/wiki/Convolutional_neural_network
3. https://en.wikipedia.org/wiki/Overfitting
4. https://www.modmypi.com/blog/whats-the-difference-between-dc-servo-stepper-motors
5. https://www.elprocus.com/difference-dc-motor-servo-motor-stepper-motor/