

Article

IoT and AI-Based Application for Automatic Interpretation of the Affective State of Children Diagnosed with Autism

Aura-Loredana Popescu , Nirvana Popescu , Ciprian Dobre , Elena-Simona Apostol  and Decebal Popescu

Computer Science Department, University POLITEHNICA of Bucharest, RO-060042 Bucharest, Romania; nirvana.popescu@upb.ro (N.P.); ciprian.dobre@upb.ro (C.D.); elena.apostol@upb.ro (E.-S.A.); decebal.popescu@upb.ro (D.P.)

* Correspondence: popescuauraloredana@gmail.com; Tel.: +40-734205891

Abstract: In the context in which it was demonstrated that humanoid robots are efficient in helping children diagnosed with autism in exploring their affective state, this paper underlines and proves the efficiency of a previously developed machine learning-based mobile application called PandaSays, which was improved and integrated with an Alpha 1 Pro robot, and discusses performance evaluations using deep convolutional neural networks and residual neural networks. The model trained with MobileNet convolutional neural network had an accuracy of 56.25%, performing better than ResNet50 and VGG16. A strategy for commanding the Alpha 1 Pro robot without its native application was also established and a robot module was developed that includes the communication protocols with the application PandaSays. The output of the machine learning algorithm involved in PandaSays is sent to the humanoid robot to execute some actions as singing, dancing, and so on. Alpha 1 Pro has its own programming language—Blockly—and, in order to give the robot specific commands, Bluetooth programming is used, with the help of a Raspberry Pi. Therefore, the robot motions can be controlled based on the corresponding protocols. The tests have proved the robustness of the whole solution.

Keywords: neural networks; autism spectrum disorder; humanoid robots; drawing interpretation



Citation: Popescu, A.-L.; Popescu, N.; Dobre, C.; Apostol, E.-S.; Popescu, D. IoT and AI-Based Application for Automatic Interpretation of the Affective State of Children Diagnosed with Autism. *Sensors* **2022**, *22*, 2528. <https://doi.org/10.3390/s22072528>

Academic Editors:
Subhas Mukhopadhyay and
Mario Munoz-Organero

Received: 25 February 2022

Accepted: 21 March 2022

Published: 25 March 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Autism spectrum disorder is a condition of brain development that can appear before the age of 3 years [1]. Children diagnosed with autism have difficulties in expressing their feelings and communicating nonverbally. Through drawings, children can express emotions, and this is where the PandaSays mobile application is helpful for children diagnosed with autism; in order to improve the communication between parents or tutors and their children and to understand better their children's emotional state, the application will offer them some indicators based on the machine learning algorithms. Currently, the model trained with MobileNet has an accuracy of 56.25%, and the database of drawings is constantly increasing.

The main feature of the application is the use of machine learning algorithms to predict the affective state of the child from his/her drawings. Analysis of the drawings could provide parents/tutors or therapists a profound understanding of the child's psychological or mental state. In addition, drawings can aid the comprehension of the child's personality. This is a gap that will be filled by the PandaSays application.

Other features contained by the application are label recognition for identifying objects, activities, products, and so on, an augmented reality module, a drawing module, a sign language module, and Google Text-To-Speech. The Sign language and Google Text-To-Speech modules might be helpful in the case of aphasia or different types of speech impairments.

The latest studies have demonstrated that humanoid robots are efficient in helping children diagnosed with autism. Recent studies have demonstrated that children are more

motivated and interested in the interaction with a robot rather than with parents or a tutor. The robots can play an important role in children's education, as articles [2–5] describe; teachers might use them in the classroom to plan lessons. The roles played by the robots encompass educational planning, teaching social skills to children, personalizing robots for every child, accompanying teachers in classroom, and using robots to increase the awareness of the child about how their behavior can influence other children.

The website "robokind.com" [6] is dedicated to using robots in children's education; their employees use robots to develop social skills for children with autism. For example, they use the "Milo" robot to simulate facial expressions and teach core vocabulary. In addition, in the curriculum, there are included modules that are emotional, behavioral, and communication related. The modules have different levels of complexity, so, as the children make progress, they are transferred to a higher module. Murray Bridge High School has been using NAO humanoid robots (programmable robots developed by Softbank Robotics, Tokyo, Japan) in the classroom for over two years, accompanying children to different classes such as IT (information technology), social skills improvement, and yoga [7]. Long-term studies suggests that social robots are effective in helping children diagnosed with autism. KASPAR (Kinesics and Synchronization in Personal Assistant Robotics) and NAO robots helped children to develop and improve their social skills and communication. KASPAR has been created to represent an expressive robot and its way of communication is more predictable and repetitive, which makes engagement with the child easier. In addition, the KASPAR robot has helped children to start recognizing emotions and is being used in hospitals [8].

In article [9], the authors describe how they used the interaction of the NAO humanoid robot with children that were diagnosed with autism, by playing a game. The NAO robot shows the children some emotional poses developed with Choregraphe. An Android application was developed to control the robot via WIFI (Wireless Fidelity). In the application there is a button called "Manage Behaviours", where the user can play the specific pose of the robot. There is also a "Refresh" button to reset the connection with the robot. The purpose of these behaviors is to teach the child how to interpret emotions and start learning expressing them. The emotions used for the NAO robot were happy, angry, scared, tired, disgusting, shy, sad, and loving. Sloboda, J. A. et al. explain how music can bring out emotions in the listener [10]. Other studies [11,12] have shown that several emotions, such as anger, happiness, fear, sadness, and anger, arise in those who listen to specific type of music or songs. In article [13], the authors explore the importance of music as a therapy to help children with autism. The aim of their research takes into consideration the number of times the therapist intercedes to help the child engage with the robot and the number of times the child imitates a robot's dance movement.

Paper [14] illustrates how the use of the NAO robots improves the verbal and non-verbal communication of children with autism. In this research, four children from Welfare of Autistic Children (SWAC, Dhaka, Bangladesh) participated, and four sessions were held. The first session was referred to the introduction of the robot and the second session had the purpose of getting answers from the children. The NAO robot asked the children some of the following questions: "How old are you?", "How are you?", "What is your father's name?", "What is your name?", and "What is your mother's name?". The third session was dedicated to physical activities, such as dancing and exercising, and the fourth session was about gathering feedback regarding session 2 and 3. After four weeks, the overall performance increased by 45% in the first child, 70% in the second child, 30% in the third child, and 75% in the fourth child.

The work in article [15] proposes to investigate the feedback received from the therapy session held with children diagnosed with autism by engaging with a robot that uses colors in different conditions. Four colors were used for gathering feedback: yellow, red, green, and blue. The child's favorite color was kept as a goal for the task, and the one that was disliked by the child was eliminated. For the experiment, a Touch Ball was used that had incorporated a force sensor that measured three axial forces. The paper concluded that the

feedback obtained from touching the device is more effective than the one resulting from the touching forces of the caregiver.

All of the papers mentioned above reflect the importance of humanoid robots in helping children with autism. The part that we are introducing is affective state interpretation from drawings, connecting it further to the humanoid robot, and using its capabilities to help children express themselves and communicate better with their parents and with other children. In addition, autism spectrum disorder can be diagnosed from early age (2 years). As soon as the child begins drawing, the “PandaSays” mobile application can help the parent/tutor to monitor the child’s affective state as early as possible.

In our previous work [16] we discussed the Android application PandaSays, which predicts the affective state of children with autism from their drawings, and with regard to integration with humanoid robots activating their movements. More specifically, in [16], the possibility of controlling the robot Alpha 1 Pro [17], from Ubtech company (Shenzhen, China), using Bluetooth communication protocol was mentioned, a subject that is explored in the current paper. The robot is used to perform some action based on the output received from the machine learning algorithm. For example, if the affective state that resulted from a drawing is “sad”, the robot will start dancing in order to distract the child from his current state and make them more joyful. In this context, the current paper reaches the following objectives: testing the PandaSays mobile application, discussing performance evaluation using deep convolutional neural networks and residual neural networks, establishing a strategy for commanding the Alpha 1 Pro robot without its native application and developing a robot module that includes communication protocols with the PandaSays application. The output of the machine learning algorithm involved in PandaSays is sent to the humanoid robot to execute some actions, such as singing, dancing, and so on. Alpha 1 Pro has its own programming language—Blockly—and in order to give the robot specific commands, Bluetooth programming is used in conjunction with a Raspberry Pi. Therefore, the robot motions can be controlled based on the corresponding protocols. After setting up the communication with the robot, the code for controlling the robot is written in Python 3. The other robots that will be used in PandaSays application are NAO (SoftBank) and Marty (robotical.io); these are used for testing the mobile application and to further discuss the performance evaluations.

The article has five sections. The first section is “Introduction”, followed by Section 2 in which the PandaSays mobile application is described, including performance tests by using deep convolutional neural networks and residual neural networks. Section 3 presents the Alpha 1 series Bluetooth communication protocol, emphasizing the whole strategy for establishing the Bluetooth serial communication between Raspberry Pi and Alpha 1P robot, using a SPP (Serial Port Profile) application. Section 4 shows a comparison between Raspberry Pi 4, Native Alpha 1P Android App, and BlueSPP Android App, underlying the evaluation results based on their connectivity time. The last section is dedicated to conclusions and future work.

2. Description of “PandaSays” Mobile Application and Performance Tests Presentation Using Deep Convolutional Neural Networks and Residual Neural Networks

PandaSays is a mobile application dedicated to children diagnosed with autism and for parents and tutors to help them better understand their children’s feelings and expressions [18]. It is an application where the child can draw. This is a simple action, but studies have shown that drawings can reveal emotions and feelings. Our app does more than that; we have tried to automate the understanding and interpretation of the emotion associated with the child by using a neural network. We train the network with a corpus of annotated drawings, and we present our results in the prediction of the obtained model for the affective state of the child.

The dataset is composed of 1279 drawings, divided into 5 affective states, representing 5 classes: “happy”, “sad”, “fear”, “insecure”, and “angry” (Figure 1). The whole train-

ing data set has been labeled by a professional psychologist specialized in interpreting children's drawings.



Figure 1. Number of trained images per class.

In our previous work [18], we realized a critical analyze of Convolutional Neural Networks and Feedforward Neural Networks, in order to find the best model to predict the affective state of the children from their drawings, and the conclusion was that MobileNet performed better than an artificial neural network. In the current work, increasing the number of drawings from the data set, we again trained the model using Convolutional Neural Networks and ResNet Neural Networks, and the new results will be also presented further in this article. The dataset was split as follows: 80% for training and 20% for testing. For the image data processing, Keras ImageDataGenerator [19] has been used. Figure 2 represents the original drawing and Figure 3 shows the drawings processed with ImageDataGenerator from the Keras library. The parameters and their values applied to the original image were the following:

- rotation_range = 40
- width_shift_range = 0.1
- height_shift_range = 0.1
- shear_range = 0.2
- zoom_range = 0.3
- horizontal_flip = True
- fill_mode = 'nearest'
- brightness_range = [0.7, 1.2]



Figure 2. Drawing without the augmentation.

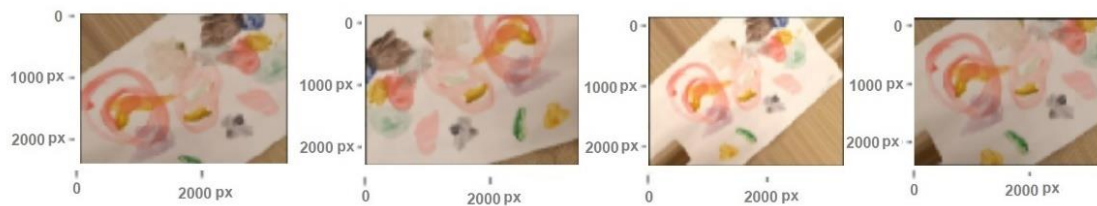


Figure 3. Drawing with Keras augmentation.

The `rotation_range` refers to the fact that the image is rotated randomly within the range of 40 degrees; `width_shift_range` means that the image is shifted horizontally (left or right) with 0.1 percent of the total image width, and `height_shift_range`, means that the drawing will be shifted vertically with 0.1 percent of the total image height. The image will be distorted along an axis with the range of 0.2 (`shear_range`), in order to simulate, to a computer, the way humans see things from different angles. The “`zoom_range`” parameter, refers to zooming inside the drawings. The drawings have `horizontal_flip` set to “True”, which means that the images will be randomly flipped horizontally. Parameter `fill_mode` is used for filling in the new pixels that were created, which can result after a rotation or a vertical/horizontal shift. The value “0.7” from the `brightness_range` indicates that the image will be darker, and the value 1.2 means that the image will be brightened.

First, for training the model, VGG16 (“Vision Geometry Group”) was used. For the training, the fully connected output layers of the model are not added and for weights, the “ImageNet” was utilized. The “ImageNet” dataset was considered because it contains approximately 14 million images and every node is illustrated by hundreds and thousands of images [20]. This dataset is relevant, because it contains classes of animals, trees, and sports that children can draw. In addition, the database of drawings from PandaSays application has a small number of images, and transfer learning helped to reaching some valuable results. Further, three convolution dense layers with a ReLU (Rectified linear activation unit) activation function were added, in order to allow the model to learn more complex functions and to obtain better results.

For compiling the model, “Adam” (adaptive moment estimation) optimizer was used, which represents an optimization algorithm that operates sparse gradients on noise, with a learning rate of 0.001 and for loss, “categorical_crossentropy”. The VGG16 model summary is described in Table 1. Next, the first 20 layers of the network were set to be non-trainable and the result is presented in Table 2; there are 14, 714, and 688 non-trainable parameters and 2, 102, and 277 trainable parameters.

Table 1. Vgg16 model summary.

Vgg16 Model Summary
Total params: 16,816,965
Trainable params: 16,816,965
Non-trainable params: 0

Table 2. Vgg16—model summary for first 20 layers.

Vgg16 Model Summary—First 20 Layers Non-Trainable
Total params: 16,816,965
Trainable params: 2,102,277
Non-trainable params: 14,714,688

The results of the VGG16 model were loss = 1.8508 and accuracy = 50.9375% (Figure 4). The batch size was 16 and the number of epochs was 40.

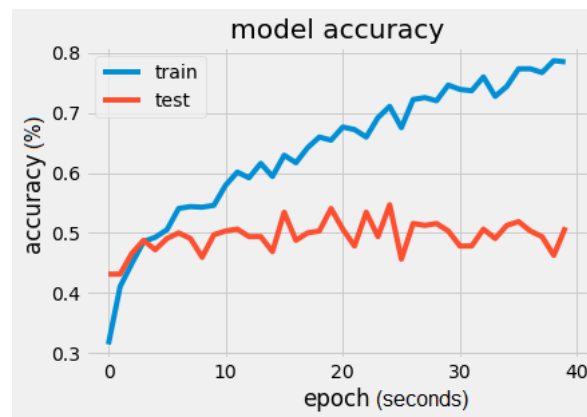


Figure 4. VGG16 model accuracy.

It can be noticed that the test loss (red line in Figure 5) is increasing while the training loss is decreasing. This indicates that the model is overfitting. The loss represents the difference between the expected output and the final output resulting from the machine learning model and is a double data type.



Figure 5. VGG16 model loss.

In the classification report from Figure 6, the data shown from 0–4 represents 5 classes (applied on all classification reports—MobileNet and ResNet) described as follows: 0—insecure; 1—happy; 2—fear; 3—angry; 4—sad. The highest precision was represented by class “happy”, followed by “insecure”, and the lowest precision was represented by the class “angry”. “Happy” class has 337 drawings, while “angry” has only 248 drawings. One of the reasons for the lower precision “angry” is the color of images: some have multiple colors, and some are black/gray. We are using the RGB (red–green–blue) color space and for the future training we will take in consideration other color spaces, such as:

	precision	recall	f1-score	support
0	0.55	0.25	0.34	69
1	0.52	0.39	0.44	44
2	0.69	0.62	0.65	95
3	0.36	0.58	0.45	50
4	0.46	0.66	0.54	62
accuracy			0.51	320
macro avg	0.51	0.50	0.48	320
weighted avg	0.54	0.51	0.50	320

Figure 6. Vgg16 classification report.

- CIE Lab (color space determined by the International Commission on Illumination)
- HSV (hue, saturation, value)
- HIS (hue, intensity, saturation)
- HSL (hue, saturation, kightness)
- YCbCr (green (Y), blue (Cb), red (Cr)) [21]

The best F1-score is obtained by the class “fear” and the lowest F1-score was represented by the class “insecure”, 0.34, which means that there are some false positives and some high false negatives. The best F1-scores are represented by the classes “fear” and “sad”. The F1 score has the following formula: $F1 = 2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$ [22], and represents the mean of the precision and the recall, which is helpful to determine if there are large number of actual negatives.

For MobileNet, the “imagenet” dataset was used as weights; the batch size was 16 and the number of epochs was 40. The model contained the fully connected classifier and the summary is given in the Table 3.

Table 3. MobileNet model summary.

MobileNet Model Summary	
Total params	4,253,864
Trainable params	16,816,965
Non-trainable params	21,888

Because we used a pre-trained model, we trained only the last layer and froze the other layers. The trainable parameters are from the last layer, named “autismOutput” (Figure 7).

```

autismOutput (Dense)          (None, 5)          5005
=====
Total params: 4,258,869
Trainable params: 5,005
Non-trainable params: 4,253,864
=====

```

Figure 7. Last layer model summary.

It can be noticed that there is a significant difference regarding the trainable parameters between the VGG16 model and the MobileNet model. MobileNet has 5005 trainable parameters and VGG16 has 2,102,277. The metrics values were loss = 1.5260 and accuracy = 56.2500%, as seen in Figures 8 and 9. In Figure 9, the validation loss is a little greater than the training loss, which means that the model is performing well and is not overfitting. It

is also important to mention that the dataset is constantly changing from the addition of new drawings.

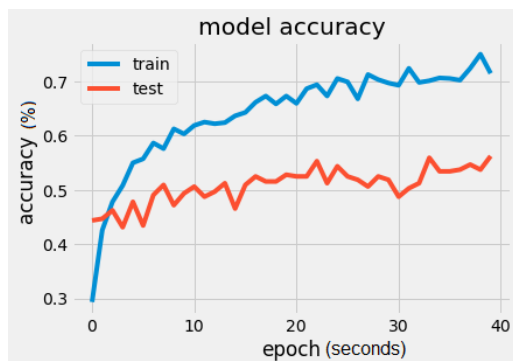


Figure 8. MobileNet model accuracy.

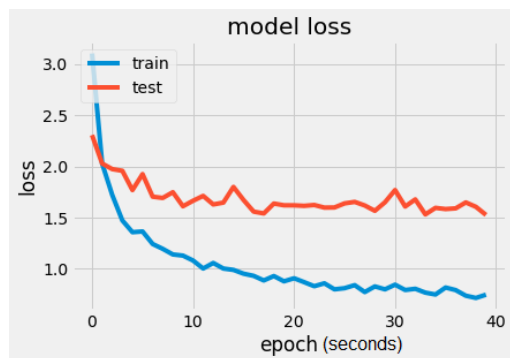


Figure 9. MobileNet model loss.

Figure 10 indicates that the highest precision was represented by the class “angry”—66%, followed by “insecure” with 60%, and the lowest precision was represented by the class “sad” at 48%. The highest F1-score is represented by the class “insecure” and the lowest by the class “sad”. As shown in Figure 11, the main output can be observed, which shows that the drawing represents a “happy” state, with 94% accuracy, when using PandaSays app with the MobileNet model.

	precision	recall	f1-score	support
0	0.60	0.65	0.62	71
1	0.54	0.65	0.59	48
2	0.57	0.61	0.59	75
3	0.66	0.42	0.51	60
4	0.48	0.48	0.48	66
accuracy			0.56	320
macro avg	0.57	0.56	0.56	320
weighted avg	0.57	0.56	0.56	320

Figure 10. MobileNet classification report.

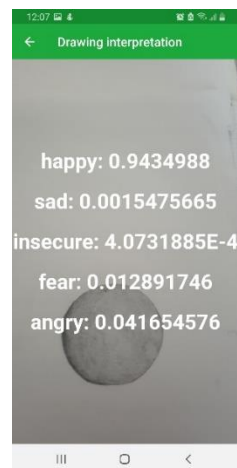


Figure 11. PandaSays drawing interpretation.

Another neural network used for training the mode, was ResNet50 (Residual Neural Networks) from Keras. For weight, it was used the “imagenet” dataset, as input tensor it was used the input of shape (224,224,3), representing the width (224 pixels), height (224 pixels) and the number of channels (3), and a single pre-trained convolutional block was included (include_top = False). The ResNet base model is formed as follows:

```
base_model = tf.keras.applications.ResNet50(weights = 'imagenet', include_top = False,
input_tensor = input_tensor)
```

The first 143 layers of the network were set to non-trainable, as seen in the code bellow:

```
for layer in base_model.layers[:143]: layer.trainable = False
```

The pretrained model (base_model) will be connected with new layers of a new model. Global pooling, a flatten layer, and a dense layer was added, with a softmax classifier.

The model summary is presented in Table 4.

Table 4. ResNet50 model summary.

ResNet50 Model Summary	
Total params	49,722,757
Trainable params	49,468,037
Non-trainable params	254,720

The ResNet50 model’s loss was 1.5041 and the accuracy was 47.6562%. As an optimizer for compiling the model, RMSprop (Root Mean Square Propagation) was used from Keras, with a learning rate of: 2×10^{-4} . The accuracy is displayed in Figure 12 and the model loss is presented in Figure 13. The validation loss is significantly greater than the training loss, which means that the model is overfitting.

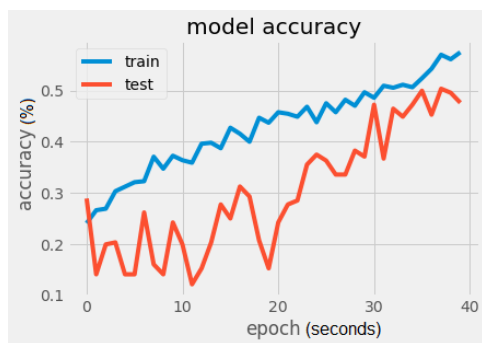


Figure 12. ResNet50 model accuracy.

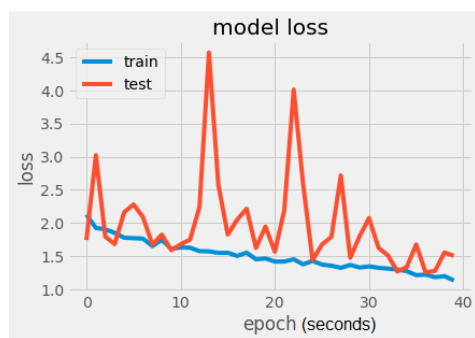


Figure 13. ResNet50 model loss.

Figure 14 illustrates that the highest precision was represented by the class “happy”—62%, followed by “sad” with 59%, and the lowest precision was represented by the class “angry”, with 33%. The highest F1-score is represented by the class “insecure” class and the lowest by the classes “sad” and “angry”—0.36.

	precision	recall	f1-score	support
0	0.42	0.70	0.52	53
1	0.50	0.47	0.49	36
2	0.62	0.52	0.57	75
3	0.33	0.39	0.36	41
4	0.59	0.25	0.36	51
accuracy			0.48	256
macro avg	0.49	0.47	0.46	256
weighted avg	0.51	0.48	0.47	256

Figure 14. ResNet50 classification report.

As noticed in Table 5, the highest accuracy is obtained by MobileNet neural network with 56.25%, followed by VGG16 with 50.93%. Although the smallest accuracy is obtained by ResNet50 neural network, its loss is the smallest, followed by MobileNet’s loss of 1.5260%. For the PandaSays application, this was chosen as the model trained with MobileNet neural network because of its accuracy results.

Table 5. Neural network accuracy and loss comparison.

Neural Network	Testing Accuracy (%)	Training Accuracy (%)	Loss (Double Data Type)
MobileNet	56.25	70.56	1.5260
VGG16	50.93	73.93	1.8508
ResNet50	47.65	57.53	1.5041

In Table 6, it is shown that VGG16 is the most complex model, because it has the largest number of parameters (138 million) [23], followed by ResNet50 with over 23 million parameters [24] and MobileNet, which is the least complex model, with 13 million parameters [25]. In terms of latency, ResNet50 has the highest latency, and VGG16 has the lowest. ResNet50 has the lowest time of convergence and MobileNet has the highest time—422.074 seconds, regarding the time of convergence.

Table 6. Neural network metrics comparison.

Neural Network	Complexity (Number of Parameters)	FLOPs (Floating-Point Operations)	Latency (s)	Time of Convergence (s)
MobileNet	13 million	16.89 million	0.01689	422,074
VGG16	138 million	16.81 million	0.01681	364,129
ResNet50	>23 million	49.90 million	0.04990	151.33

3. Alpha 1 Pro Server–Client Connection and Bluetooth Communication Protocol

One of the objectives of the current research is to present how the Alpha 1 Pro robot can be commanded without its native application and considering the integration with PandaSays application. Alpha 1 Pro has 16 servo joints, equivalent to humans' joints. The robot can dance, move, fight. Based on interpretation of drawings, the robot will execute these actions, helping with children's therapy and mood interpretation. In this section, the communication protocol will be presented based on the Bluetooth communication. The Alpha 1 Pro robot can be programmed through Bluetooth using the Bluetooth communication protocol [26].

The command format formula is:

Header + Length + Command + [first parameter][second parameter][third parameter] + Check + End Character

BT handshake:

Field Header (2B) is set to 0XFB 0XBF. Length (1B) represents the total number of bytes of (header + length + command + parameter + CHECK). Field Command (1B) refers to a specific command and Parameter (nB) represents one parameter at least. If the parameter is not needed, then the default value 0x00 is used. Field CHECK (1B) is composed of length + command + parameter and the End character (1B) is set to 0XED.

The following actions can be implemented by the robot, writing the specified code:

- Handshake (Read): FB BF 06 01 00 08 ED
- Obtaining an action list (Read): FB BF 06 02 00 08 ED
- Implementing an action list (Write): FB BF 06 03 00 09 ED
- Sound switch (Write): FB BF 06 06 00 08 ED
- Play control (Write): FB BF 06 07 00 08 ED-pause
- Heartbeat packet (Write): FB BF 06 08 00 08 ED
- Reading robot state (Write): FB BF 06 0A 00 10 ED
- Volume adjustment (Write): FB BF 06 0B 09 08 ED
- Powering off all servos (Write): FB BF 06 0C 00 12 ED

- Controlling all servo indicators (Write): FB BF 06 0D 00 08 ED
- Clock calibration (Write): FB BF 06 0E 08 ED
- Reading clock parameters (Read): FB BF 11 0F 06 18 ED
- Setting clock parameters (Write): FB BF 12 10 07 20 ED
- Reading the software version number of the robot (Read)
- Reading battery capacity of the robot (Read): FB BF 06 11 00 08 ED
- Controlling the motion of a single servo (Write): FB BF 08 23 03 13 ED
- Controlling the motion of multiple servos (Write): FB BF 08 23 03 12 ED
- Setting offset value of a single servo (Write): FB BF 08 26 03 12 ED
- Setting offset value of multiple servos (Write): FB BF 07 27 02 09 ED
- Play completion (Automatic report of BT): FB BF 05 31 00 05 ED
- Allowing change during play (Write/Automatic report of BT): FB BF 06 32 01 08 ED
- Completing action list sending (Automatic report of BT): FB BF 05 81 00 05 ED

These commands will be used further with the BlueSPP application to transmit to the robot one of these actions. More details about the BlueSPP communication with the robot is presented in Section 3.2.

Alpha 1P robot has dual-mode Bluetooth 3.0/4.0 BLE (Bluetooth Low Energy) + EDR (Enhanced Data Rate). Bluetooth Classic (or BR/EDR) is the Bluetooth radio that is mostly used in streaming applications and in the following devices/applications wireless headsets, wireless speakers, wireless printers and keyboards, and file transfers between devices. Bluetooth LE (low energy), is utilized in low-bandwidth applications that rarely need data transmission between devices. Bluetooth LE is known for its very low power consumption and is present in the following devices/applications: smartphones, computers and tablets [27], mobile payment applications, healthcare system, ticketing or access control applications.

Every Bluetooth chip is marked with a globally unique 48-bit address, which represents the Bluetooth address, device address or MAC (Media Access Control) address. This is identical in nature to the MAC addresses of Ethernet [27]. Some Bluetooth protocols are utilized in the same contexts as internet protocols. For connecting the raspberry pi device with Alpha 1P robot, it will be used RFCOMM (Radio frequency communication) protocol -set of transport protocols, providing emulated RS-232 (Recommended Standard 232) serial ports. RFCOMM provides mainly the same attributes as TCP (Transmission Control Protocol); the difference between the two is that TCP supports up to 65,535 open ports on a single machine, while RFCOMM only allows 30 [28].

3.1. Establishing Bluetooth Communication with Python

To provide access to Bluetooth system resources on GNU/Linux computers, we use PyBluez, which is a Python extension. The PyBlueZ module provides a high-level socket interface for establishing a connection between two Bluetooth devices. One acts like a client and the other as server. In our case, Raspberry pi is the server and Alpha 1P is the client. The Python code for establishing the client-server connection of the robot and Raspberry Pi can be found in the links provided in references [29,30].

3.2. Establishing Bluetooth Serial Communication between Raspberry Pi and Alpha 1P Robot Using a SPP Application

Bluetooth programming in Python is based on the socket programming model, a socket being an endpoint of a communication channel. When sockets are first created, they are not connected. In order to complete a connection, it must be called connect (client application) or accept (server application) [31]. For sending messages to Alpha 1P via Bluetooth, a Samsung Note 9 mobile phone and BlueSPP android application were used. The communication between the server and the client is illustrated in Figure 15.

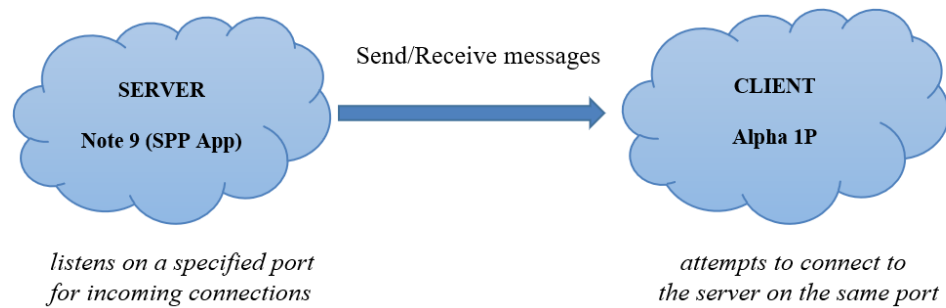


Figure 15. Server–client communication.

According to the Bluetooth communication protocol document, the robot can perform numerous actions. For powering off of all the servos, we transmit to the robot the next command: FB BF 06 0C 00 12 ED (Figure 16).

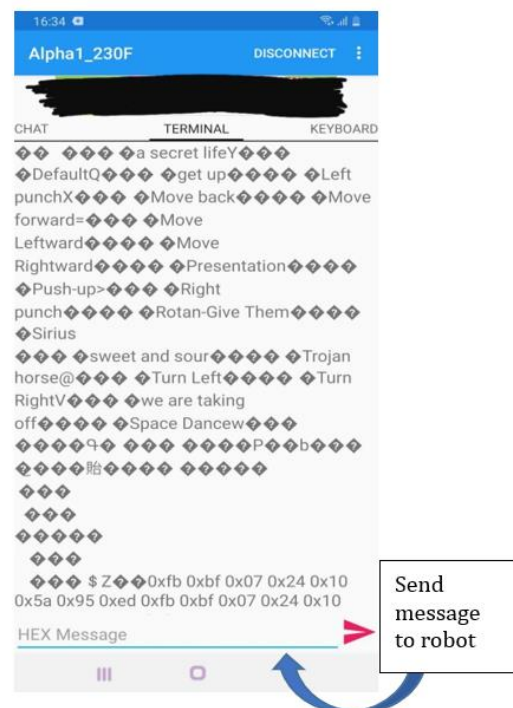


Figure 16. Message sending to robot through SPP android application.

In order to see the status of the robot, the next code is transmitted: FB BF 06 0A 00 10 ED. The send operation of the message is illustrated in Figure 16. The question marks in this figure are shown, because the numbers are sent randomly from Alpha1 Pro to the BlueSPP app, so the application is interpreting the bytes as ascii literals.

As it can be seen in Figure 17, the robot receives the next code associated to each status:

- FB BF 07 0A 00 01 12 ED (Speaker)
- FB BF 07 0A 01 00 12 ED (Play)
- FB BF 07 0A 02 80 93 ED (Volume)
- FB BF 07 0A 03 01 15 ED (Servo LEDs)

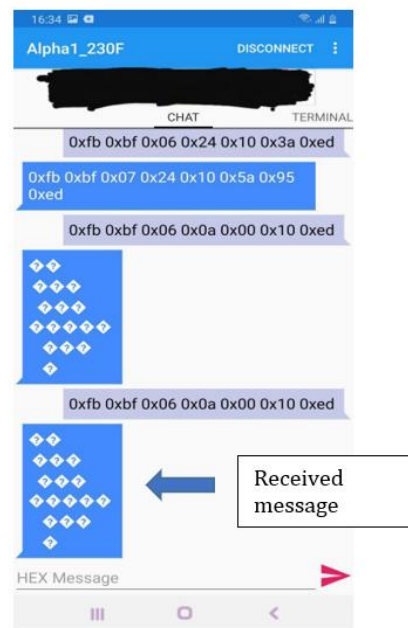


Figure 17. Response of the robot.

Figure 17 illustrates that the robot responded to the message sent using Bluetooth communication protocol.

4. Comparison of the Communication Times from Candidate Devices

The BlueSPP android app is a Bluetooth Serial Port Profile communication tool. The device used was a Samsung S10+ Android phone. To measure the time, a digital watch was used. The required time to connect to the robot was 3.8 s. Bluetooth latency, representing Alpha 1P robot's execution of a certain action, for BlueSPP Android app, on Samsung S10+, was 5 s (Figure 18).

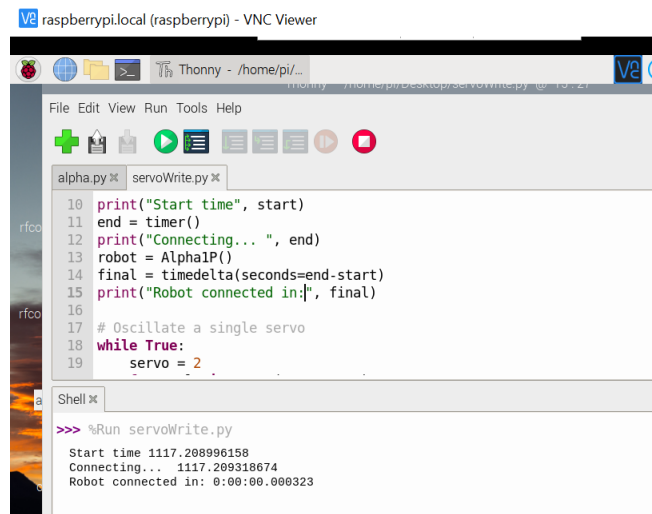
```

Select Command Prompt
12-01 16:59:35.281 27055 27055 D BluetoothLeScanner: Start Scan with callback
12-01 16:59:35.283 2378 3882 D BtGatt.ContextMap: add() - appId: 1000, appId: 27055, appName: android.uid.system
12-01 16:59:35.283 2378 2694 I bt_stack: [INFO:gatt_api.cc(958)] GATT_Register c527d2e9-a0ae-4a78-a1c1-879012692758
12-01 16:59:35.284 2378 2694 I bt_stack: [INFO:gatt_api.cc(995)] allocated gatt_if=10
12-01 16:59:35.284 2378 2465 D BtGatt.GattService: onScannerRegistered() - UUID=c527d2e9-a0ae-4a78-a1c1-879012692758, scannerId=10, status=0
12-01 16:59:35.285 27055 31123 D BluetoothLeScanner: onScannerRegistered() - status=0 scannerId=10 mScannerId=0
12-01 16:59:35.286 2378 3882 D BluetoothAdapter: STATE_ON

```

Figure 18. BlueSPP app execution time of action for Alpha 1P robot.

For connecting the Raspberry Pi 4 to the Alpha 1P Robot, a Bluetooth adapter with frequency of 2.4–2.4835 (GHz) was used. The PyBluez library was used to establish a client-server connection. Figure 19 shows the time in seconds for the connection to the robot—3.23 s.



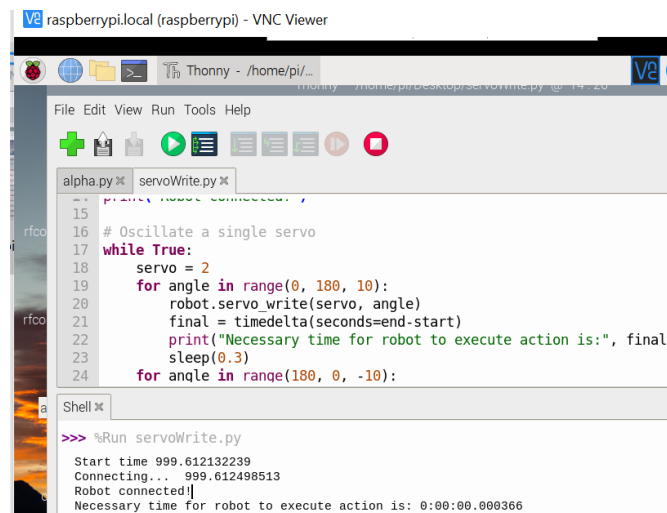
```

raspberrypi.local (raspberrypi) - VNC Viewer
Thonny - /home/pi/...
File Edit View Run Tools Help
alpha.py x servoWrite.py x
10 print("Start time", start)
11 end = timer()
12 print("Connecting... ", end)
13 robot = Alpha1P()
14 final = timedelta(seconds=end-start)
15 print("Robot connected in:", final)
16
17 # Oscillate a single servo
18 while True:
19     servo = 2
Shell x
>>> %Run servoWrite.py
Start time 1117.208996158
Connecting... 1117.209318674
Robot connected in: 0:00:00.000323

```

Figure 19. Raspberry Pi connection to Alpha 1P Robot.

The Bluetooth latency, representing Alpha 1P robot's execution of a certain action, was 3.66 s for Raspberry Pi, as shown in Figure 20.



```

raspberrypi.local (raspberrypi) - VNC Viewer
Thonny - /home/pi/...
File Edit View Run Tools Help
alpha.py x servoWrite.py x
15 # Oscillate a single servo
16 while True:
17     servo = 2
18     for angle in range(0, 180, 10):
19         robot.servo_write(servo, angle)
20         final = timedelta(seconds=end-start)
21         print("Necessary time for robot to execute action is:", final)
22         sleep(0.3)
23     for angle in range(180, 0, -10):
24
Shell x
>>> %Run servoWrite.py
Start time 999.612132239
Connecting... 999.612498513
Robot connected!
Necessary time for robot to execute action is: 0:00:00.000366

```

Figure 20. Execution time for Alpha 1P robot with Raspberry Pi.

The Alpha 1P robot has its native android application called "Alpha 1" [32], which is available here: <https://play.google.com/store/apps/details?id=com.ubt.alpha1s>, accessed on 10 February 2022. The total time representing the successful execution of pairing and the connection to the robot was 28.18 s. The Bluetooth latency, representing Alpha 1P robot's execution of a certain action, for the Android "Alpha 1" Android app on a Samsung S10+ was 31.22 s.

Table 7 presents the comparative results regarding the connectivity time of the robot. The best time is represented by Raspberry Pi (3.23 s) through the Bluetooth version, which is lower than the other devices.

Table 7. Connectivity time results for Raspberry Pi, Alpha 1, and BlueSPP Android Apps.

Device/Application	Raspberry Pi/PyBluez Library	Alpha 1 Android App	BlueSPP Android App
Connectivity time (seconds) to Alpha 1P Robot	3.23	28.18	3.8
Bluetooth version	Bluetooth 2.0	Bluetooth 5.0	Bluetooth 5.0
Bluetooth version of Alpha 1P robot	Bluetooth 3.0/4.0 BLE + EDR		

Abbreviations: BLE = Bluetooth low energy; EDR = enhanced data rate.

Table 8 shows that the best time for the Alpha 1P robot to execute an action is obtained by Raspberry Pi, followed by BlueSPP Android App.

Table 8. Latency results for execution time of Alpha 1P robot's action.

Device/Application	Raspberry Pi/PyBluez library	Alpha 1 Android App	BlueSPP Android App
Latency time (seconds) representing executing an action by Alpha 1P Robot	3.66	31.22	5.0
Bluetooth version	Bluetooth 2.0	Bluetooth 5.0	Bluetooth 5.0
Bluetooth version of Alpha 1P robot	Bluetooth 3.0/4.0 BLE + EDR		

5. Conclusions

Humanoid robots play an important part in helping children diagnosed with autism. As several studies demonstrated, children are keener to interact with a robot than with their parents, doctors, or tutors. Robots can teach children how to express feelings, how to interact with other children, and how to communicate better. The interaction with the robot is one of the essential parts in the PandaSays application for the machine learning algorithm to be complete.

Raspberry Pi with PyBluez can be used to create a fast Bluetooth connection and to start programming a robot to execute some actions. The time difference represented on the Android devices is due to the Bluetooth latency as is well analyzed in [33].

The contributions brought by this paper are:

- In this paper, it was shown that the best trained model was the one trained with MobileNet, because of its highest accuracy—56.25%. MobileNet is the least complex neural network, with 13 million parameters, in comparison with VGG16 and ResNet50. Because of its low complexity and small size, MobileNet is more suitable for mobile applications. Those are the reasons why the MobileNet model was chosen to be used in PandaSays mobile application.
- Establishment of a control methodology for connecting the robot Alpha 1 Pro with PandaSays application, using Bluetooth communication protocol.
- Development of a robot module that includes the communication protocols with the app PandaSays, which will be used further to control the robot and send the machine learning output to it, in order to perform a specific action.
- Python module implementation for setting the client–server communication.
- The configuration setup of the Raspberry Pi and robot's Bluetooth communication protocol, used to measure latency and connectivity time.

- The efficiency of using Raspberry Pi with PyBluez to create a client–server connection, represented by the lowest latency—3.66 s and by the connectivity time—3.23 s, which was faster than other devices (Android Device, BlueSPP app).
- Emphasis of the importance of the humanoid robots in helping children diagnosed with autism.

Given the small data set, we used a pretrained model and applied Transfer Learning. In future work, the next step will be to enlarge the data set and to apply the following algorithms to train the model: K nearest neighbor, decision tree, and support vector machines.

Another step for us would be to use the PandaSays application and one of the humanoid robots (Alpha, NAO, or Marty) in Autism Centers in Romania to gather feedback and improve the machine learning algorithm.

Author Contributions: Conceptualization, A.-L.P. and N.P.; methodology, N.P.; software, A.-L.P.; validation, C.D., E.-S.A. and D.P.; formal analysis, N.P.; investigation, A.-L.P.; resources, C.D.; data curation, D.P.; writing—original draft preparation, A.-L.P.; writing—review and editing, A.-L.P. and N.P.; visualization, N.P.; supervision, N.P.; project administration, N.P.; funding acquisition, C.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors gratefully acknowledge the help of Georgiana Soricica, a psychologist, for professional evaluation of children’s drawings and the Autism Help Association.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

BLE	Bluetooth Low Energy
EDR	Enhanced Data Rate
LE	Low Energy
MAC	media access control
RFCOMM	Radio frequency communication
RS-232	Recommended Standard 232
TCP	Transmission Control Protocol

References

1. Autism and Health: A Special Report by Autism Speaks. Available online: <https://www.autismspeaks.org/science-news/autism-and-health-special-report-autism-speaks> (accessed on 5 June 2020).
2. Villano, M.; Crowell, C.R.; Wier, K.; Tang, K.; Thomas, B.; Shea, N.; Schmitt, L.M.; Diehl, J.J. DOMER: A Wizard of Oz interface for using interactive robots to scaffold social skills for children with Autism Spectrum Disorders. In Proceedings of the 2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Lausanne, Switzerland, 8–11 March 2011; pp. 279–280. [CrossRef]
3. Shamsuddin, S.; Yussof, H.; Ismail, L.; Hanapiah, F.A.; Mohamed, S.; Piah, H.A.; Zahari, N.I. Initial response of autistic children in human-robot interaction therapy with humanoid robot NAO. In Proceedings of the 2012 IEEE 8th International Colloquium on Signal Processing and Its Applications, Malacca, Malaysia, 23–25 March 2012; pp. 188–193. [CrossRef]
4. Krithiga, R. Socially Assistive Robot for children with Autism Spectrum Disorder. In Proceedings of the 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), Bangalore, India, 15–16 November 2019; pp. 1–4. [CrossRef]
5. Alcorn, A.M.; Ainger, E.; Charisi, V.; Mantinioti, S.; Petrović, S.; Schadenberg, B.R.; Tavassoli, T.; Pellicano, E. Views on Using Humanoid Robots with Autistic Learners in Special Education Settings in England. *Front. Robot. AI* **2019**, *6*, 107. [CrossRef] [PubMed]
6. Documentation Regarding Robots Used for Helping Children with Autism. Available online: <https://www.robokind.com/robots4autism> (accessed on 10 September 2021).

7. Research Regarding Humanoid Robots. Available online: <https://www.csiro.au/en/research/technology-space/ai/social-robots-support-children-with-autism> (accessed on 28 September 2021).
8. Documentation Regarding KASPAR Robot. Available online: <https://www.herts.ac.uk/kaspar/impact-of-kaspar> (accessed on 30 September 2021).
9. Miskam, M.A.; Shamsuddin, S.; Samat, M.R.A.; Yussof, H.; Ainudin, H.A.; Omar, A.R. Humanoid robot NAO as a teaching tool of emotion recognition for children with autism using the Android app. In Proceedings of the 2014 International Symposium on Micro-NanoMechatronics and Human Science (MHS), Nagoya, Japan, 10–12 November 2014; pp. 1–5. [CrossRef]
10. Beer, J.M.; Boren, M.; Liles, K.R. Robot assisted music therapy a case study with children diagnosed with autism. In Proceedings of the 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Christchurch, New Zealand, 7–10 March 2016; pp. 419–420. [CrossRef]
11. Juslin, N.P. Cue utilization in communication of emotion in music performance: Relating performance to perception. *J. Exp. Psychol. Hum. Percept. Perform.* **2000**, *26*, 1797–1813. [CrossRef]
12. Rigg, M.G. The mood effects of music: A comparison of data from four investigators. *J. Psychol.* **1964**, *57*, 427–438. [CrossRef]
13. Sloboda, A.J. Music structure and emotional response: Some empirical findings. *Psychol. Music.* **1991**, *19*, 110–120. [CrossRef]
14. Farhan, S.A.; Khan, M.N.R.; Swaron, M.R.; Shukhon, R.N.S.; Islam, M.M.; Razzak, M.A. Improvement of Verbal and Non-Verbal Communication Skills of Children with Autism Spectrum Disorder using Human Robot Interaction. In Proceedings of the 2021 IEEE World AI IoT Congress (AllIoT), Seattle, WA, USA, 10–13 May 2021; pp. 356–359. [CrossRef]
15. Lee, J.; Obinata, G. Developing therapeutic robot for children with autism: A study on exploring colour feedback. In Proceedings of the 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Tokyo, Japan, 3–6 March 2013; pp. 173–174. [CrossRef]
16. Popescu, A.L.; Popescu, N. Machine Learning based Solution for Predicting the Affective State of Children with Autism. In Proceedings of the 2020 International Conference on e-Health and Bioengineering (EHB), Iasi, Romania, 29–30 October 2020; pp. 1–4. [CrossRef]
17. Alpha 1 Pro Robot. Available online: <https://www.ubtrobot.com/products/alpha-1p?ls=en> (accessed on 10 January 2022).
18. Popescu, A.-L.; Popescu, N. Neural networks-based solutions for predicting the affective state of children with autism. In Proceedings of the 2021 23rd International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 26–28 May 2021; pp. 93–97. [CrossRef]
19. Image Data Processing. Available online: <https://keras.io/api/preprocessing/image/> (accessed on 10 January 2022).
20. ImageNet Description. Available online: <https://www.image-net.org> (accessed on 14 January 2022).
21. Kim, H.K.; Park, J.H.; Jung, H.Y. An Efficient Color Space for Deep-Learning Based Traffic Light Recognition. *J. Adv. Transp.* **2018**, *2018*, 2365414. [CrossRef]
22. F1-Score Explained. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (accessed on 10 February 2022).
23. VGG16 Architecture. Available online: <https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/> (accessed on 24 February 2022).
24. ResNet50 Architecture. Available online: <https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33> (accessed on 24 February 2022).
25. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
26. Alpha1 Series Bluetooth Communication Protocol. Available online: <https://www.robotshop.com/community/forum/uploads/default/original/3X/8/e/8e7299b1389b4526389fb02278b7ba27e40d3135.pdf> (accessed on 21 August 2021).
27. Types of Bluetooth Technology. Available online: <https://www.bluetooth.com/blog/wireless-connectivity-options-for-iot-applications-technology-comparison/> (accessed on 15 July 2021).
28. Rfcomm Tutorial. Available online: <https://www.btframework.com/rfcomm.htm> (accessed on 20 September 2021).
29. Rfcomm Client Code. Available online: <https://gist.github.com/AuraLoredana/7533a48cb4e4c80e4cc0e84b68d1112e> (accessed on 24 February 2022).
30. Rfcomm Server Code. Available online: <https://gist.github.com/AuraLoredana/a95427ffd87d7f49b52c43973bfbcf8f> (accessed on 24 February 2022).
31. Huang, A.S.; Rudolph, L. *Bluetooth. Essentials for Programmers*; Cambridge University Press: Cambridge, UK, 2009. [CrossRef]
32. Alpha Robot Specifications. Available online: <https://downloads.ubtrobot.com/> (accessed on 20 September 2021).
33. Robert Triggs. Android’s Bluetooth Latency Needs a Serious Overhaul. 2021. Available online: <https://www.soundguys.com/android-bluetooth-latency-22732/> (accessed on 20 September 2021).