2nd National Level Students' Research Conference on
"Innovative Ideas and Invention with Sustainability in Computer Science and IT-2021"
In association with International Journal of Scientific Research in Computer Science, Engineering and Information Technology | ISSN : 2456-3307 (www.ijsrcseit.com)

Finger Gestures Detection Using Convolution Neural Network for Playing Virtual Cricket

Aadesh Dalvi, Shivam Chauhan, Gaurav Shinkar
M.Sc. Data Science & Big Data Analytics, MIT-WPU, Pune, Maharashtra, India

ABSTRACT

This is an era of intelligent machines. With the advancement in artificial intelligence, machine learning, and deep learning, machines have started to impersonate humans. Gesture recognition is a hot topic in computer vision and pattern recognition. We wanted to develop a game with less complexity and in an interactive way so that users of any age group can appreciate it.

In this paper, the proposed model uses convolutional neural networks (CNN) to recognize the hand gesture with accurate results. This process flow consists of placing your palm over the particular segment over the screen and finger recognition using CNN classifier. The fingers are recognized using the Convolution, Normalization, Activation, Max-pooling, and Dropout layers. In this paper, we have compared different models using various combinations of these layers. Based on the performance and complexity of these models, we have selected a model with higher performance and reasonable complexity which helped us to classify the image correctly.

A chatbot is also integrated with the game that will help users to understand the rules of the game. Users need to ask a query to the bot. The Bot will understand the query and return the appropriate answer. A web page will be provided to the user where he/she can play the game and ask queries to the chatbot.

Keywords : Intelligent machines, Artificial Intelligence, Machine Learning and Deep Learning, Convolution, Normalization, Activation, Max-pooling, and Dropout layers

I. INTRODUCTION

This project "Hand Gestures Detection Using Convolution Neural Network for Playing Virtual Cricket" uses deep learning techniques to recognize the hand gesture and to count fingers in the frame to build an interactive and fun game. The built model can analyze hand signs and categorize them. It allows users to give input just by gestures without using a mouse, keyboard, or any other peripheral devices. Such a human-machine interaction game creates an environment that provides a simulation of reality. Based on the user's hand gesture, this model recognizes the gestures and provides the test result. As per the test result, this model gives the users an interactive and easy approach to play the game with the system, providing more enjoyment for people in an effective way.

The second part of the game focuses on a chatbot. A chatbot is a software used for user support in various

industries. There are different ways to develop a chatbot. Based on the requirement and resources available, we can select any method of implementation. A chatbot that can understand the context of the conversation seems to be more user-friendly and effective. Here, we have used the cosine similarity technique to understand input from the user and uses natural language processing (NLP) techniques to maintain the context of the conversation. This type of chatbots can be used in small industries or businesses for automating customer care as user queries will be handled by chatbots thus reducing the need for human labor and expenditure. This chatbot helps the user to understand the rules of the game.

II. LITERATURE REVIEW

The performance of deep learning neural networks often improves with the amount of data available for preprocessing [1]. As the amount of data increases, the deep learning model tends to provide better results hence data plays a very important role while building any deep learning or machine learning model.

Image data augmentation is perhaps the most well-known type of data augmentation and involves creating transformed versions of images in the training dataset that belong to the same class as the original image [2]. Transforms include a range of operations from the field of image manipulation, such as shifts, flips, zooms, and much more.

There are different chatbots available. Some of them use technologies like deep learning or machine learning whereas some of them are rule-based. Rupesh Singh and the co-author used a deep learning approach to develop the chatbot. This method uses TensorFlow for developing the neural network model of the chatbot and uses the NLP techniques to maintain the context of the conversation [3]. It can be used for small-scale businesses. The drawback of this technique is, it requires a large amount of data to learn itself as it is using a neural network.

Human-machine interaction is a study looking at the transmission and communication of information and emotion between humans and machines. For a human to sense the realistic and comfortable relation when interacting with machines, an interface for natural and intuitive interactions will be important for the bridging of the relation between humans and machines. Gestures are the unsaid words of humans which he expresses in the form of actions. It allows individuals to communicate feelings and thoughts with different emotions with words or without a word [4].

Other than language. The hand gesture is a way to express human intention and emotion. Using bode gestures is one of the common and natural ways of communication and interaction [5].

The human hand gestures are detected and recognized using the convolutional neural networks (CNN) classification approach. This process flow consists of hand region of interest segmentation using mask image, fingers segmentation, normalization of segmented finger image, and finger recognition using CNN classifier [6]. P. S. Neethu, R. Suguna and Divya Sathish in their paper “An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks” stated convolutional neural network provides better performance when used with enhancement techniques.

III. PROPOSED SYSTEM

There are 3 different modules of this project. 1. Model to detect hand gestures 2. Chatbot to understand user queries 3. A web page where users can play and ask queries. Based on the study was done and available resources we proposed the following solution approach for our problem statement.

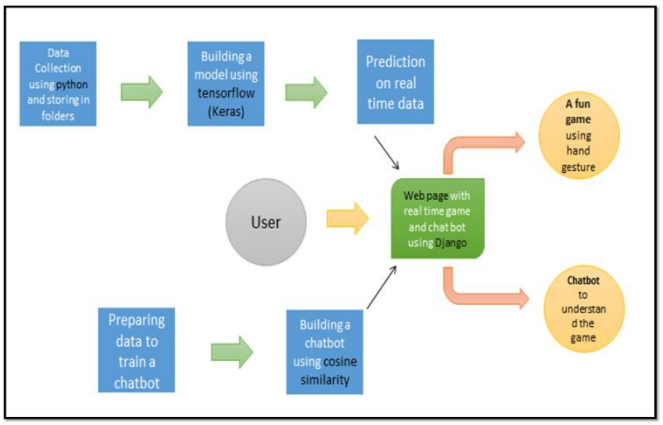

Image 1 Solution Approach

There are different approaches for data collection:

i) Download the images from Google and train the model on those images. We can use web scraping to download images, but we need to delete unrelated images manually. It was a time-consuming task. So, we decided to go with the second approach.

ii) Generating our own dataset to build the model:
We have used python programming language for this. For data collection, we have used OpenCV library by python to capture the images and OS library to store folder-wise images of each sign on our local machines. There are many efficient pre-trained models used for object detection like Yolo (You only look once) and Mobilenet SSD (Single-shot detection). YOLO is an object detection system for real-time data processing. It divides images into frames & then forms a bounding box around the object & predicts class. Mobilenet SSD is designed for mobile embedded apps. It extracts a feature map & applies convolution. It is fast and it has simple architecture.
To learn and explore more about neural networks, we built our own model for hand sign recognition. To build this model we have used Keras which is an open-source neural network library written in Python. After building an acceptable model, using OpenCV and the model, we were able to recognize hand signs on real-time data.

To build the chat, we used the cosine similarity method. It requires less data, space, time, and processing power. In this method, we check the similarity between two strings and return a numeric value. This technique helps to understand the intent of the user. We have provided data to the chatbot, which has different intent along with potential patterns/ questions asked by users and expected answers from the bot. using this data, we were able to understand the user input and give a proper response.
The last module of this project is web integration. We have used Django framework for this. Django is a high-level Python Web framework that helps in rapid development. There are other frameworks too like flask. We opt for Django over flask as it allows database connections. Flask is single page application preferable for personal blogs, forums, etc. So, we choose Django as its complete package for web development.

3.1 TECHNOLOGY STACK

Python - It is a high-level, interpreted programming language. It has a simple syntax and it also allows developers to write programs with fewer lines.
OpenCV – It is an open-source computer vision and machine learning software library mainly aimed at real-time computer vision.
OS - It is a module in python that provides functions for interacting with the operating system & using its dependent functionality. It comes under Python's standard utility modules.
Keras - It is an open-source neural-network library & is capable of running on top of TensorFlow. It is designed to enable fast experimentation with deep neural networks; it focuses on being user-friendly, modular, and extensible.
Django – It is a high-level Python Web framework that encourages rapid development. It Helps developers take applications from concept to completion as quickly as possible and avoid many

common security mistakes with the ability to scale quickly and flexibly.

3.2 Design Model

We will provide a web page to user where he/she can play game or understand rules of the game by asking questions to chatbot. On the home page, user can start playing game or ask question in the text box given. When he/ she ask any question, it will try to understand the question and return appropriate response using the model built. Once user starts the game, he/she has to make hand signs defined in the region of interest. Model will recognize the signs and simultaneously generate a number. Based on the rules of the game, system number and signs of the user, game will stop or continue till 6 rounds. In the end, result will be displayed on the screen itself. Users need to refresh the web page in order to start the game again.

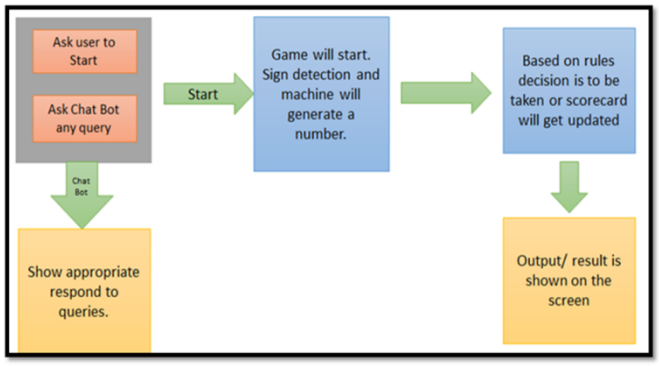

Image 2 Process Flow

IV. DATASET DETAILS

The data was collected by compiling a data collection function (Code) where separate folders for every sign (finger count) were generated. On running the code, a frame was generated along with ROI (Region of Interest). So, the respective sign was to be shown in the given ROI. Strictly while collecting the data background was kept as a plain surface without any distraction to obtain maximum accuracy.

There were approximately 1000 such images collected for each sign and further the data folders were split as train, test, validation. A total of 6240 images were collected. The size of data collected is 187Mb.

Some basic things to be considered while collecting images are as follows:

Plain Background for ROI.

Finger sign to be shown in ROI (Region of Interest) i.e., the blue frame shown in the upper image.

The only hand gesture is to be fitted within the boundaries of ROI, no other body-part or object should be included in ROI while capturing.

Gestures

The only respective key should be pressed while collecting images for that sign. Example: 1 for One, 2 for Two, and so on till 6.

Only set of gestures to be used as shown in image above.

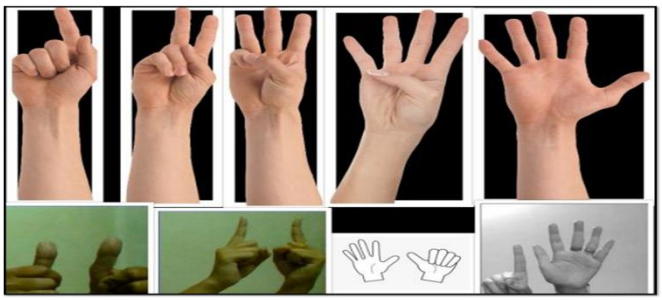

Image 3 Dataset Overview

To improve the model accuracy, we considered invariance while collecting the data.

Invariance means, you can recognize an object as an object, even when its appearance varies. This is generally a good thing, because it upholds the object's identity, category, (etc.) across changes in the aspects of the visual input, for example the relative positions of the viewer/camera and the object.

Note: Here the hand/Fingers sign is the object/target.

1) Positional / Transnational Invariance:

Ability to detect positional shifts, or translations of the target in the image.

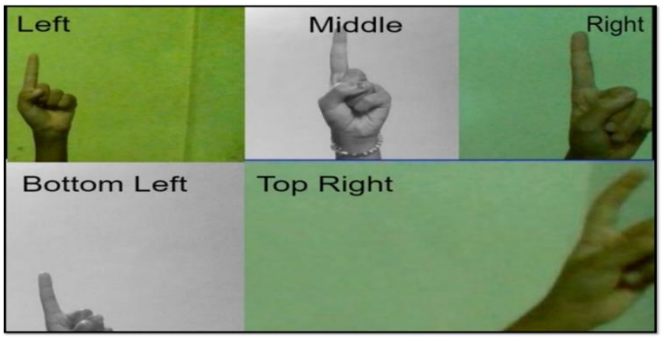

Image 4 Positional / Transnational Invariance

2) Rotation / View-Point Invariance:

Ability to detect circular movement of an object around a center (or point) of rotation. It's a change in the viewpoint of seeing the object.

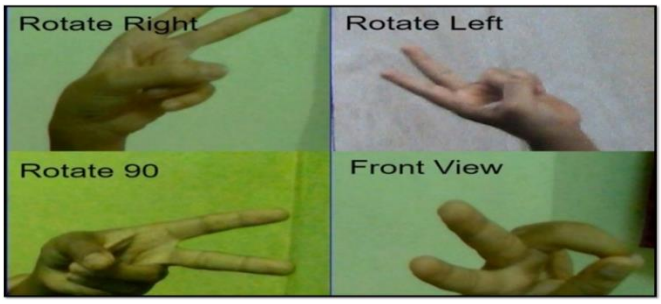

Image 5 Rotation / View-Point Invariance

3) Size Invariance:

Ability to detect a change in the size of the target in the image.

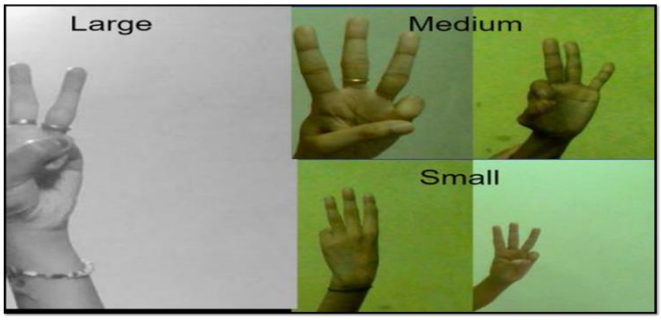

Image 6 Size Invariance

4) Illumination Invariance:

Ability to detect the target in the image even if visibility is low due to light intensity or shadows etc.

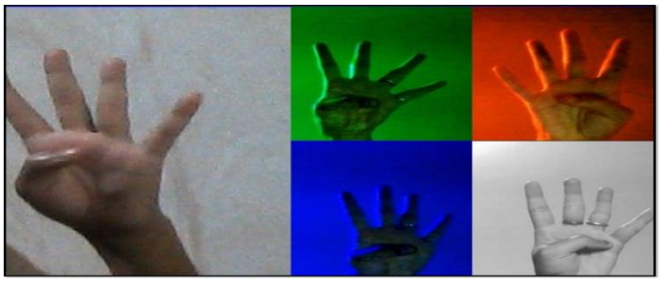

Image 7 Illumination Invariance

5) Space between the Fingers:

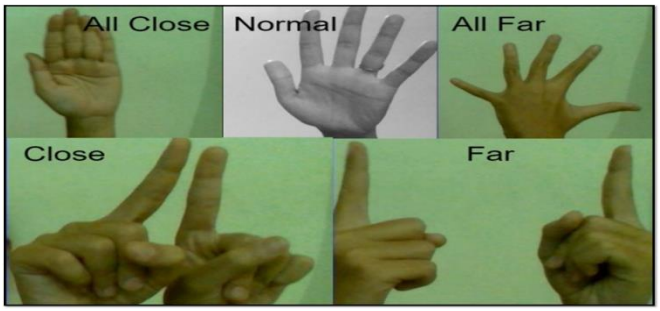

Image 8 Different Spacing between the Fingers

Ability to detect the sign even if fingers are close or wide apart.

V. IMPLEMENTATION

We have used Keras to build a model for hand sign recognition. There were many different models with different combinations of layers and the number of neurons built. Different hyperparameters need to be considered while building any neural network. This is the structure of our model used for hand sign recognition.

Our model architecture consists of a Batch Normalization layer followed by Convolution layers. We have used 6 convolution 2D layers (number of filters being 32,32,128,128,256,256 and kernel size as 3,4,3,3,3,3) each accompanied by activation function 'relu' and MaxPooling layer.

We then used a flattening layer to give input to the feed-forward network. Total 5 dense layers were used having 256, 128, 64, 32, 7 neurons that take 'relu' as an activation function. Dropout layers with a rate of

0.2 go along with each of these dense layers for regularization.

Sequential: A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

In the project, input tensors of size (n batches, 310, 310, 1) and output vector of size (7, 1) are used.

Batch Normalization: It is used to normalize the output of the previous layers. The activations/filters scale the input layer. Using batch normalization learning becomes efficient also it can be used as regularization to avoid over-fitting of the model.

1 batch normalization layer is used just before the CNN layers.

Convolution Layers: It is an important layer that is used to do image recognition, image classification, etc. Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using pixels of input data. It is a mathematical operation that takes 2 inputs such as an image matrix and a filter or a kernel.

In the project, 6 convolution layers are used. The motive is to keep adding layers until over-fit, after which the regularization techniques could be used for generalization.

The convolution layers were accompanied by activation function 'relu', the reason is that the images are naturally non-linear, so the rectifier function was preferred.

Kernels: The kernel is a filter that is used to extract the features from the images. In the project, 32, 128, 256 numbers of kernels are used in different Convolution layers. These convolution kernels act as a filter to create a feature map. Kernels of size 4, 3 were used because the benefit that smaller kernel size provides that it reduces computational costs and weight sharing and extract the more granular features as we move along the layers. The padding function used is 'same', doing this improves performance as it retains the information at the borders.

Pooling-layer Parameters: The pooling layer aims to down-grade the input (image, hidden-layer output matrix, etc.), reducing its dimensionality by keeping the max value (activated features) in the sub-regions binned to reduce the cost of operations. For the project, 6 max-pooling layers (size: 2) were used after every CNN layer.

Dense Layers: Dense layers are keras's alias for Fully connected layers. These layers give the ability to classify the features learned by the CNN. For the project, 5 dense layers are used (neurons as 256, 128, 64, 32, 7). Activation function 'Relu' is used for hidden layers while 'Softmax' is used for the output layer to classify the 7 outcomes (0-6).

Regularization: To over-fit the model, we tend to add more layers to the model. Once we obtain higher accuracy in our training set, we use regularization like l1 / l2 regularization, dropout, batch norm, data augmentation, etc. to reduce over-fitting. For the project, Dropout layers are used by switching off 20% of neurons in Dense layers.

Adam - The superiority of the Adam optimizer lies in its adaptive learning rate and is favored due to its relatively fewer parameters tuning.

Loss function: Categorical cross-entropy is used in the project as we have multiple classes where each example belongs to a single class.

Performance Graph of Model

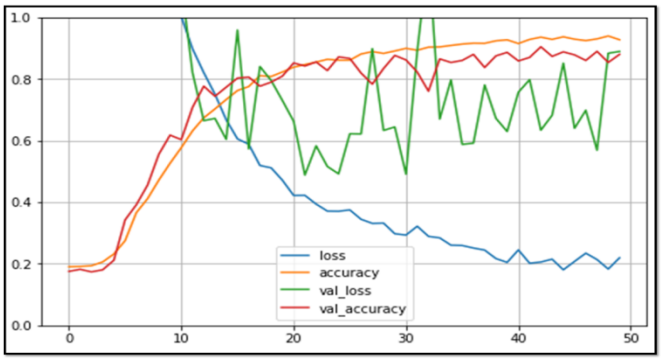

Image 9 Performance of the model

Below is the table-description for the models created for the project:
Model 10 was chosen as the final one as it stands out for the best performance in terms of accuracy and complexity.

Model	Input Image	Architecture	Training Accuracy	Validation Accuracy
1	64*64	2 Conv2D Filters (num=32,32; size=3,3) Padding = valid + 2 Max Pooling + 4 Dense Layers	92	77
2	64*64	2 Conv2D Filters (num=32,64; size=3,3) Padding = valid + 2 Max Pooling + 2 Dense Layers	86	70
3	128*128*1	2 Conv2D Filters (num=32,32; size=3,3) Padding = valid + 2 Max Pooling + 2 Dense Layers	25.6	23.43
4	128*128*1	4 Conv2D Filters (num=32,6,128,128; size=3,4,3,2) Padding = same + 4 Dense Layers	52.12	50.5
5	128*128*1	Batch Normalization + 4 Conv2D Filters (num=32,32,128,128; size=3,4,3,2) Padding = valid + 4 Max Pooling + 4 Dense Layers + Dropouts	52.71	56.76
6	180*180*3	3 Conc2D Filters (num=32,64,128; size=3,3,3) Padding=valid + 1 Max Pooling + 2 Dense Layers + Dropout	73	60
7	256*256	3 Conc2D Filters (num=32,128,256; size=3,3,3) Padding=valid + 3 Max Pooling + 3 Dense Layers	96	74
8	310*310*1	Batch Normalization + 6 Conv2D Filters (num=32,32,128,128,256,256; size=3,4,3,3,3,2) Padding = valid + 6 Max Pooling + 5 Dense Layers + Dropouts	91	88
9	310*310*1	Batch Normalization + 6 Conv2D Filters (num=32,32,128,128,256,256; size=3,4,3,3,3,2) Padding = valid + 6 Max Pooling + 5 Dense Layers + Dropouts	94	89
10	310*310*1	Batch Normalization + 6 Conv2D Filters (num=32,32,128,128,256,256; size=3,4,3,3,3,3) Padding = valid + 6 Max Pooling + 5 Dense Layers + Dropouts	92	88

Most of the models were disregarded because of their input size; the higher the image resolution, the better the accuracy.

Some were set aside because of the over-fitting that caused the higher error for the unknown inputs. (Model 7, 1, 2, 6)

Some of them were overlooked because they completely under-fit the training data. (Model 3, 4, 5).

Model 10 was chosen for the final evaluation because of the fine stability between bias and variance. (Model 9 is equally preferable though)

VI. FUTURE WORK

The system can be further extended to Game Intelligence where the machine can predict the number generated by the human in ROI. It will be challenging for a user to defeat the bot.

A voice module can also be added to increase the game's ability and agility. In this module, rather than making hand signs, a user needs to utter any random number.

The change of background with increasing its ability to detect off a variation can be made available. This will help to overcome the limitation of a plain background.

The web app can contain more information like user score per delivery, strike rate and high score till now, etc. A scoreboard can be further integrated to increase the creative aspect of the system.

VII. RESULT

This model uses human hand gesture recognition to determine the output. These experiments cover various models aimed to discover the best combination of layers needed that classify the fingers into separate classes. CNN is a multi-layered neural network that is one of the deep learning techniques used efficiently in the field of gesture recognition. On-going through the results, we concluded that CNN is the most appropriate method to be used in the hand gesture system because of its highly accurate result. We have used a machine learning approach to create a bot in this project. Chatbots based on machine learning do not understand the meaning of sentences. It learns how to respond based on previous experience. Though we have used some NLP functions, the actual process through which response is generated is using machine learning. As said earlier, we created the model and trained it with the intent file thus more diverse the intent file more accurate will be the result.

VIII. REFERENCES

[1]. https://keras.io/api/preprocessing/image/

[2]. https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

[3]. Singh, R., Paste, M., Shinde, N., Patel, H., & Mishra, N. (2018). Chatbot using TensorFlow for small Businesses. 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). doi:10.1109/icicct.2018.8472998

[4]. Seyfeddinipur, Mandana and Gullberg, Marianne (2014) 'Introduction to visible action as utterance.' In: Seyfeddinipur, Mandana, (ed.), From gesture in conversation to visible action as utterance. Amsterdam: Benjamins, pp. 1-12. 21 Nov 2015

[5]. Tseng, K.-T., Huang, W.-F., & Wu, C.-H. (2006). Vision-Based Finger Guessing Game in Human Machine Interaction. 2006 IEEE International Conference on Robotics and Biomimetics. doi:10.1109/robio.2006.340271

[6]. P. S. Neethu, R. Suguna, Divya Sathish, “An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks”, 23 March 2020.