AUTOMATIC SNOOKER-PLAYING ROBOT WITH SPEECH RECOGNITION USING DEEP LEARNING

A THESIS

Presented to the Department of Mechanical and Aerospace Engineering

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

Committee Members:

Emel Demircan, Ph.D. (Chair)
Jalal Torabzadeh, Ph.D.
Panadda Marayong, Ph.D.

College Designee:

Hamid Rahai, Ph.D.

By Kunj H. Bhagat

B.E., 2016, Gujarat Technological University, India

December 2018

ABSTRACT

AUTOMATIC SNOOKER-PLAYING ROBOT WITH SPEECH RECOGNITION USING DEEP LEARNING

By

Kunj H. Bhagat

December 2018

Research on natural language processing, such as for image and speech recognition, is rapidly changing focus from statistical methods to neural networks. In this study, we introduce speech recognition capabilities along with computer vision to allow a robot to play snooker completely by itself. The color of the ball to be pocketed is provided as an audio input using an audio device such as a microphone. The system is able to recognize the color from the input using a trained Deep Learning network. The system then commands the camera to locate the ball of the identified color on a snooker table by using computer vision. To pocket the target ball, the system then predicts the best shot using an algorithm. This activity can be executed accurately based on the efficiency of the trained Deep Learning model.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DOF	Degree of Freedom
NN	Neural Networks
CNN	Convolutional Neural Networks
K-NN	K-Nearest Neighbors
AI	Artificial Intelligence
VGG	Visual Geometry Group
SDK	Software Development Kit
ML	Machine Learning
DL	Deep Learning
ReLu	Rectified Linear Unit
API	Application Programming Interface

CHAPTER 1

INTRODUCTION

Machine Learning (ML) is currently revolutionizing the technological world. In today’s market, robotics is being driven by Machine Learning technologies. According to a recent survey [1] published by the Evans Data Corporation Global Development, robotics and Machine Learning are top priorities for developers in 2016, with 56.4% of polled developers indicating that they are building robotics apps; further, 24.7% of all participants stated the importance of Machine Learning for their applications. This illustrates the current prominence of robotics and Machine Learning in science and technology.

Machine Learning applications in robotics, such as for Computer Vision, Imitation Learning, and Self-Supervised Learning, have had powerful and continuous influence on robotic advancements. These ML applications have also had a significant impact in the medical field by providing a variety of services through surgical and biomedical devices and rehabilitation robots. ML has substantially pushed the boundary of robotic capabilities, allowing them to accomplish tasks more accurately and efficiently than humans. Further, these technologies push the boundaries of human performance by allowing for robotic supplements to enhance human performance.

Deep Learning (DL), one of hottest buzzwords in today’s hi-tech world, is a branch of Machine Learning based on learning data representation. As a part of artificial intelligence, DL is a prime factor behind several innovations. It has allowed for great progress in many fields of human research. DL applications, such as image classification, speech recognition, automatic handwriting generation, character text generation, and image caption generation, have started making a meaningful impact in our ease and comfort in our daily lives. This project utilizes two of the most widely-studied DL applications: image classification and speech recognition. Work related to speech recognition in Machine Learning is further examined in Mohamed et al. [2] and Hinton et al. [3].

This research proposes an efficient technique, less complex than existing techniques, to automate a robot’s performance of an activity, directed by speech recognition and using image classification through DL. We introduce a methodology to accurately train the DL models for both speech recognition and image classification. Compared with existing technologies, this method reduces the need for interaction between the humans and the robot to a noticeable level, once the robot is made smart enough to recognize the human speech, detect the objects in an image, and perform the assigned tasks accordingly. This research intends to solve the intelligence problem by bridging the gap between humans and the robots they operate.

In this study, our objective was to program and train a robot to play a game of snooker by itself, simply based on a human operator’s speech input. In layman’s terms, we wanted the robot to execute the task using the same stimulus (color recognition) that a human player would use. The game of snooker is played with a cue (white) ball and colored (target) balls on a billiards table. Players must strike the cue ball to pocket the colored balls in the correct sequence [4].

In this project, speech input was provided by randomly speaking any one of the following color words: red, blue, green, yellow, orange, purple, maroon, or black. We fed the randomly-chosen color of the target ball as an input to the system through an audio device. The recorded speech command was used as input to the Deep Learning network and our trained model was able to recognize the command, i.e., which colored ball is to be hit. Once the speech command is processed by the audio-recognition network, the model commands the computer vision network to detect the specified colored ball on the table. The computer vision network uses DL’s

Convolutional Neural Network (CNN) model and MATLAB’s [5] Image Processing toolbox. The CNN model classifies the colored ball on the table through a learning-based training procedure. Related work is studied in Szegedy et al. [6]. The Image Processing toolbox is used once the colored ball is detected using the CNN model. The toolbox executes one of its in-built functions to locate the pockets on the table as well as the centroids of the target and cue balls. With the knowledge of the location of the pockets, target ball, and cue ball, we run our user-defined function to figure out the best shot for pocketing the target ball. This user-defined function that we developed contains the algorithm to interpret the best (optimal) shot. We thoroughly studied the physics behind this game to develop the algorithm to select the best shot. Work related to the basics of the game and its algorithm is explored in Koehler [7] and Billiards Congress of America [8]. In this research, we use the Machine Learning components to decide how a task can be implemented efficiently in real-time applications.

While used for snooker in this particular instance, the methodology we developed in this model could be used to accomplish a wide variety of real-world applications in crucial fields such as medicine, rehabilitation, and physical therapy. Further, with the help of musculoskeletal modeling, these models could be made capable of performing human skills such as scooping, throwing, and pick-n-place objects; such functions would be ideal for devices such as prosthetic limbs. In addition, visually-impaired patients could also get assistance from ML-enabled robotic devices to perform daily activities. ML models and networks could be trained to predict and prevent injuries to athletes or sports personalities. In this way, the potential scope of this methodology could impact several vital sectors of our lives.

CHAPTER 2

LITERATURE REVIEW

Machine Learning

Machine Learning was first introduced and defined by Arthur Samuel, an American AI innovator, as a branch of computer science that enables a computer to learn different aspects without being coded [9]. This field, which has grown exponentially in recent years, evolved from the study of pattern recognition and computational learning concepts in AI [10]. Machine Learning techniques give us a way of solving real-time problems based on past machinery data. In contrast, conventional statistical methods were intensively used for solving such problems. Modern ML methods provide a base for constructive algorithms that make predictions from the given data. Models developed using these algorithms are employed in a wide range of real-world applications where high-level computational performance is desired. These techniques have also played a significant role in the field of data analytics [11], where it is used to solve the challenging problems in both branches of the field: predictive and descriptive analytics. Prior techniques for data analysis required a model to predict the outcomes for completely unseen data whereas modern techniques necessitate the description of the content revealed by the historical data.

Due to heavily increasing demands in computational technologies, Machine Learning has gained immense popularity in the hi-tech world. Machine Learning not only allows computers to make accurate predictions and make autonomous decisions, it also reduces the need for human intervention and correction. Machine Learning promises to be one of the most impactful techniques for solving intelligence problems.

Different Machine Learning Algorithms

Broadly speaking, Machine Learning algorithms are classified into three types: supervised, unsupervised, and reinforcement learning. The choice between these algorithms is made based on the available data. An explanation of each type is provided below.

Supervised learning: Supervised learning helps to build a compact model from the provided class labels [12]. The model is especially developed based on the features to be used for identifying future outcomes. In other words, this type of Machine Learning is useful to generalize the features hypothetically; this generalization allows the system to perform predictions about future outputs. The resulting classifier performs assigning of class labels to the test cases with known features but with unknown values. Table 1 [12] shows a classification problem where different features are classified, and output is predicted based on known labels.

TABLE 1. Labeled Data in Standardized Format: Classification Problem

Test case	Features				Output
	Feature *1*	Feature *2*	…	Feature *n*	Classification
1	xx	xx		xx	true
2	xx	xx		xx	true
3	xx	xx		xx	false
…	…	…		…	…

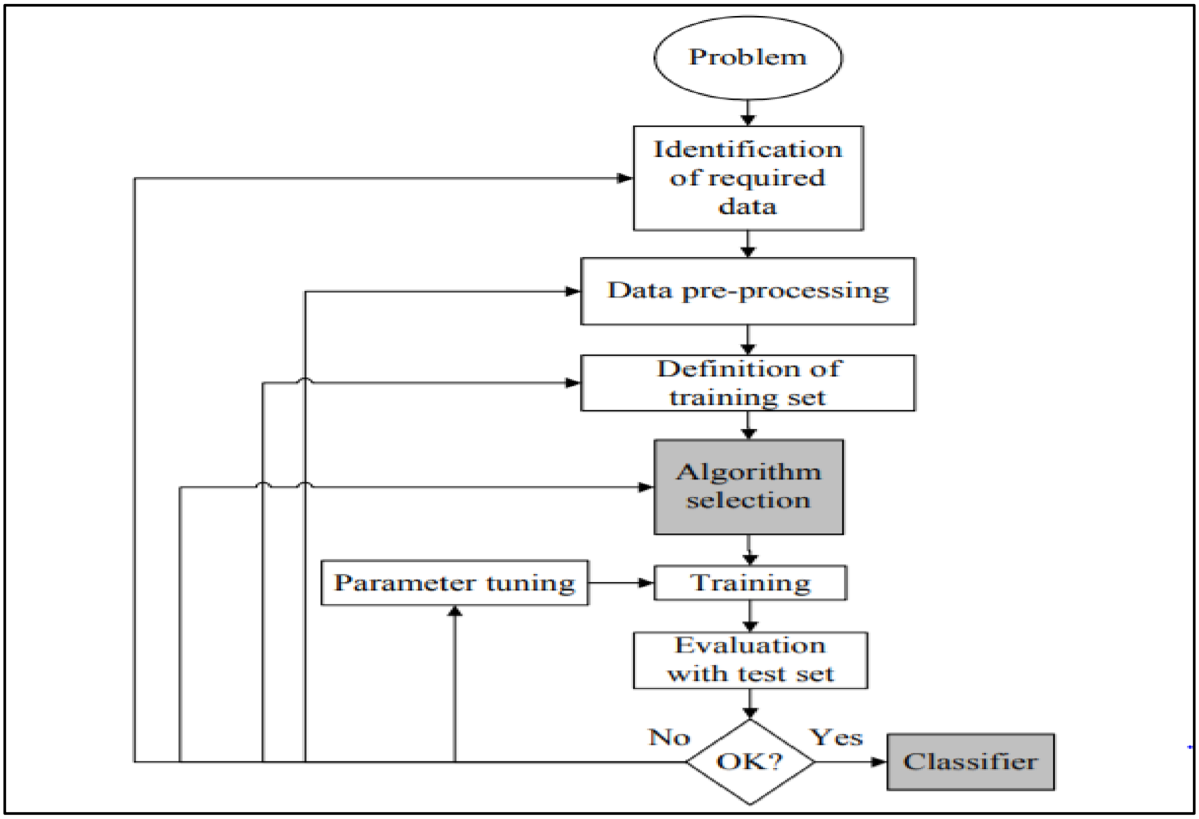


FIGURE 1. Process flowchart of supervised Machine Learning. Image taken from Kotsiantis et al. [12].

The procedural flowchart for applying supervised ML model to a real-time application is depicted in Figure 1. More related work to supervised ML is explored in Dietterich [13], Fatista and Monard [14] and Brighton and Mellish [15].

Unsupervised learning: In this ML type, the smallest feature subset is extracted from the dataset using a defined criterion, which reveals the "natural" clusters (groupings) from the data [16]. Unsupervised learning is a bit more difficult problem than is supervised learning. Unlike supervised ML models, in which classes perform feature extraction, in unsupervised models, the user needs to explicitly define "meaningful" and "natural" features in the data. As a result, this method is applied when the dataset containing the input data does not have labelled instances. Figure 2 shows a sample of how clustering is performed from the input data [17]. An approach

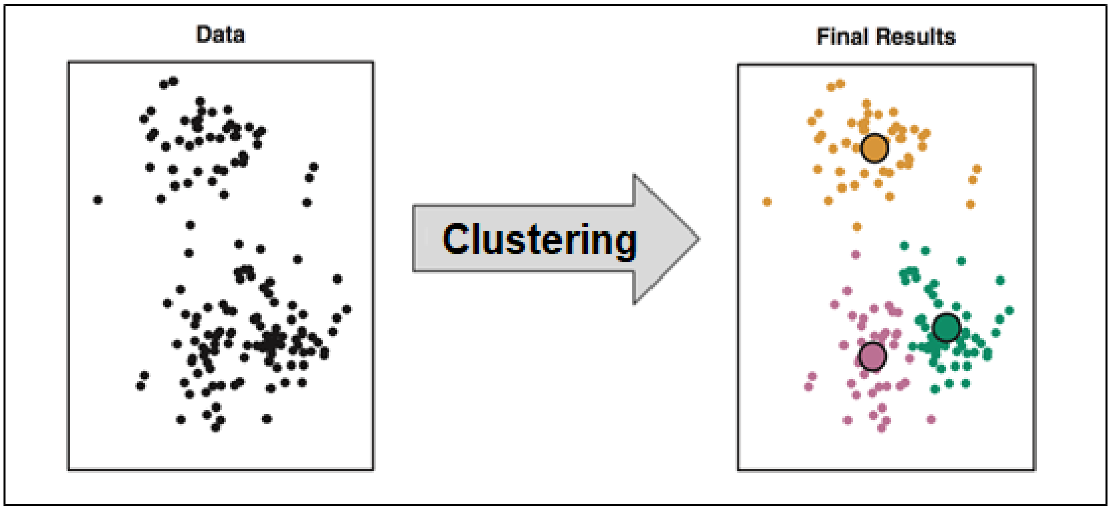


FIGURE 2. Clustering features from the data.

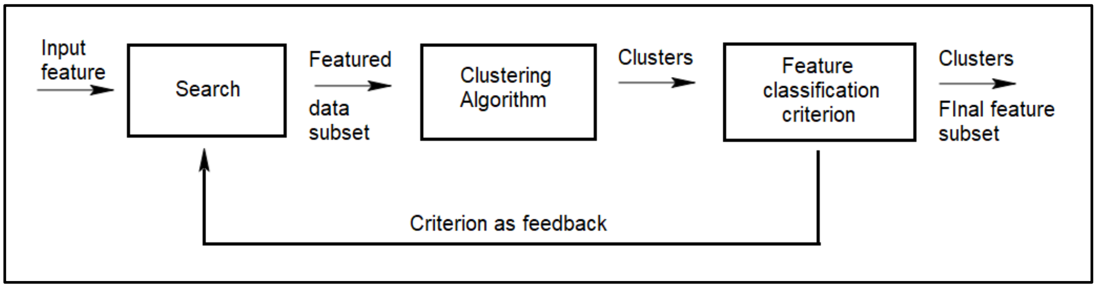


FIGURE 3. Approach flowchart in unsupervised Machine Learning.

flowchart [16] for an unsupervised ML model is shown in Figure 3. Related work to unsupervised learning can be found in the study of Bousquet et al. [18] and Le et al. 19].

Reinforcement learning: Reinforcement learning is the learning of what is to be done. A numerical reward signal is awarded after every action performed by the learner [20]. The learner is unaware of the actions to be performed, but instead learns and trains which action would produce the maximum reward signal by executing them. Another striking feature of this type of learning is that machine studies the complete problem of a goal-oriented learner (agent) working in an environment. Figure 4 depicts the interface between the agent and the environment. Both interact continuously: the agent performs actions and the environment responds to these actions and provides new states or situations to the agent. The environment also develops rewards; the

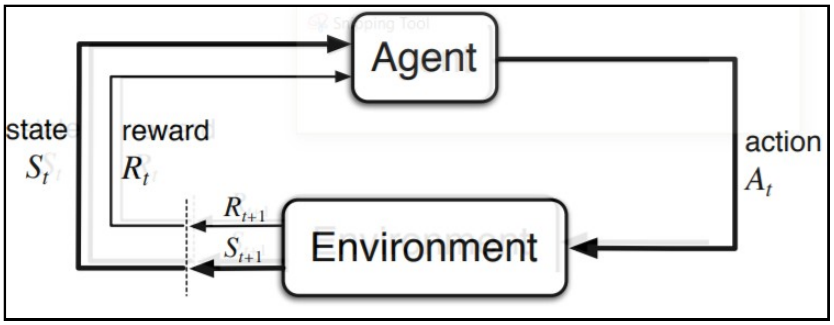


FIGURE 4. Agent-environment interaction in reinforcement learning. Image from Barto and Sutton [20].

agent will then evaluate, based on its past actions and rewards, which available reward offers the greatest value, and executes the necessary action.

To illustrate the concept of reinforcement learning, we consider a simple example: the children's game of tic-tac-toe (Figure 5). The problem can be formulated as follows: How can we develop a model that would find the opponent's imperfections and learn to win the game? Here, the role of numerical reward or value function comes into action. Value is given each possible state from 0 to 1 as probabilities of winning. For instance, we are playing all O's and if

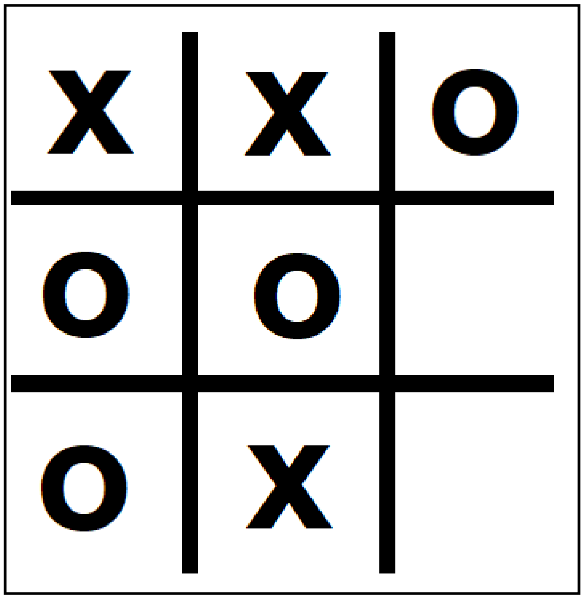


FIGURE 5. Tic-tac-toe game board.

three O’s are already present in a row, the value given that state would be 1, as we have already won; by contrast, when three X’s are lined up in a row, the value function is assigned 0, as no chance of winning exists. 0.5 is given for 50% chance of winning. Numerous games are played against the opponent, and the state leading to increased chance of winning is selected. Every possible move will be executed; following this training, the machine can then make optimal choices based on past experience. This example highlights the significant features of reinforcement learning. More description about this example is presented in Barto and Sutton [20]. Further examples and applications of reinforcement learning are analyzed and explored in Tadepalli and Ok [21], Sutton [22] and Schwartz [23].

Deep Learning: Neural Networks

The term “deep” in Deep Learning relates to the number of layers through which data is processed. Deep learning is a special class of Machine Learning techniques that uses a set of several layers of non-linear functions for feature extraction and prediction [24]. Each successive layer takes input from the output of the preceding layer. Essentially, Deep Learning networks learn from a hierarchy of instances. Each layer learns to yield more abstract and representation from the input data. In an image classification problem for text recognition, for example, a matrix containing the pixels of the image is fed as raw input; the first layer in the network abstracts edges from those pixels; the second layer manipulates arrangement of the edges; the third layer detects straight and curved lines in each letter in the text; the fourth layer recognizes the content of the text. Like with many other Deep Learning applications, our model is based on neural networks (NNs). A neural network is a processing system that contains performance characteristics which are analogous to biological neural networks [25]. A neural network has a large number of processing components called nodes or neurons. Each node is connected to other

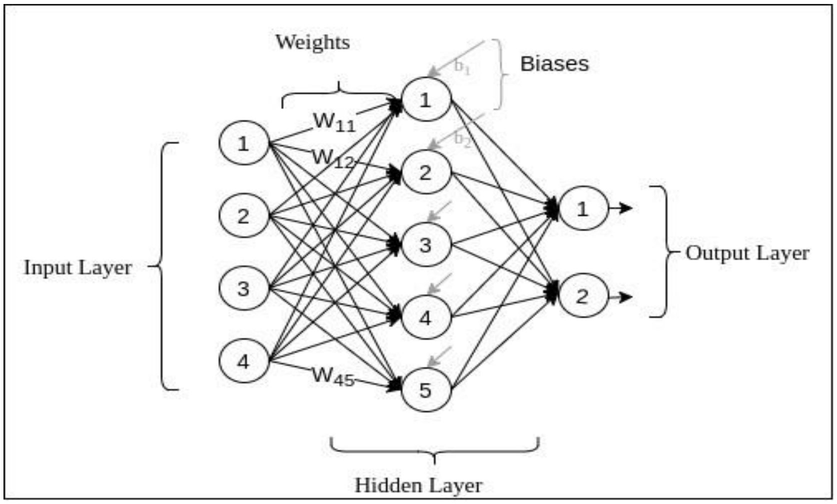


FIGURE 6. Neural network architecture.

nodes by means of connecting links with appropriate weights and biases. Figure 6 shows the architecture for a typical neural network. Figure 7 shows the model where nodes or neurons propagate in the forward direction; the left most column of nodes refers to the input layer; the central column represents the intermediate layers (usually referred as hidden layers); the right most column of nodes is the output layer. Detailed discussions on neural networks can be found in can be found in Haykin [26] and Lippmann [27].

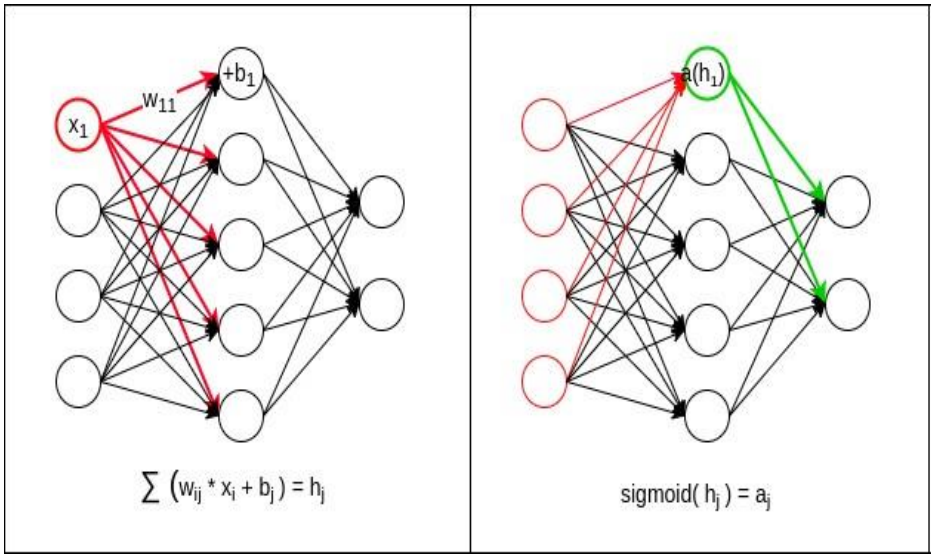

x_1
w_{11}
$+b_1$
$a(h_1)$
$\sum (w_{ij} * x_i + b_j) = h_j$
$\text{sigmoid}(h_j) = a_j$

FIGURE 7. Forward propagation in neural network.

We now review the most frequently used types of neural networks: recurrent neural networks, generative adversarial networks, and convolutional neural networks. These are frequently applied to various problems prevailing in the technological sector.

*Recurrent neural network (RNN)***:** RNN models belong to a class of neural networks where the linkage between the neurons sequentially generates a directed graph [28]. Because the connections are formed in a sequence; the model shows progressive nature in a given time sequence. Figure 8 [29] shows a simple (left) and a fully-connected (right) recurrent neural network. A simple RNN can be employed in elementary applications, such as validating a string of characters. Some of the nodes exhibit temporally sequential information and receive feedback from nodes of the previous layer. The feedback is usually time-delayed. The network model can be trained to anticipate the succeeding letter in a string to eventually validate the string.
By contrast to simple RNN, the fully connected RNN do not possess specific input nodes. In addition, each node has input from other nodes and it can also exhibit feedback to itself. Advanced recurrent neural network techniques and applications are studied in Li et al. [30], Graves et al. [31], and Fernandez et al. [32].

Generative adversarial networks (GANs): These network models include execution of an application by two neural networks interacting with one another on a zero-sum based framework. The zero-sum is conceptually a representation where a contestant's gain or loss is precisely traded-off by the opponent's loss or gain. In other words, a network generates contestants for modeling while the other assesses their actions [33]. The technical purpose of this network is to enhance error frequency for the distinctive network [34]. Related work is studied in Gross et al. [35].

Convolutional neural networks (CNN): This class of deep neural networks is intensively

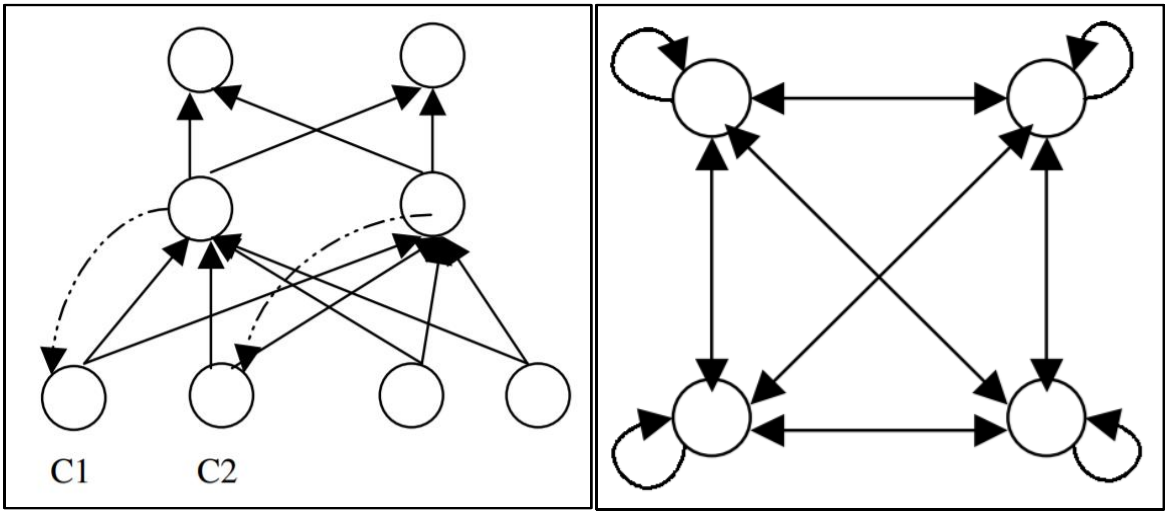


FIGURE 8. Simple RNN (left) and fully-connected RNN (right).

used in Machine Learning applications where imagery visualization is necessitated. A CNN network has multiple layers, where these layers can be of three types: convolutional layer, max-pooling, and fully-connected [36, 37]. Convolutional layer has a matrix of nodes or neurons. The previous layer of this layer also needs to be in form of a rectangular section or a matrix, so that each neuron in convolutional layer can accept an input from the previous layer. Hence, the convolutional layer is merely image convolutional of the preceding layer. Shown in Figure 9 is an architecture for a typical CNN. Research and related work of CNN is explored in Nam and Han [38] and Wang et al. [39].

Transfer learning: In this type, knowledge obtained while solving a classification problem is used in solving other problems. This enables the user to reutilize the previously obtained training data rather than building it right from scratch. For instance, a trained model to detect a car can be interchangeably used to detect a huge truck. Transfer problem, a question of how these neural networks are recycled to solve new problem, was first formulated and presented in Pratt et al. [40]. The neural network model is trained with large-scale data and the learned model is stored for future developments. The prominent advantage of this technique is

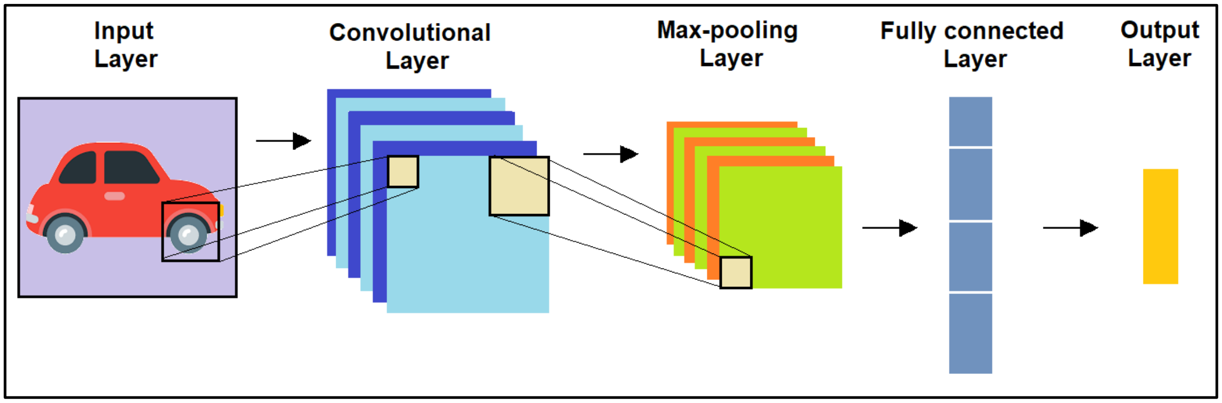


FIGURE 9. Convolutional neural networks-architecture.

the remarkable reduction in learning time. This methodology could simplify the development of complex neural networks associated with real-time systems. Some special applications of transfer learning are text clustering, sentiment classification, collaborative filtering, sensor-based location estimation, AI planning and navigation, more applications are illustrated in Gu and Zhou [41], Li et al. [42], Le et al. [43] and Zhou et al. [44]. Related work to transfer learning is further presented in Pratt [45] and Dietterich [46].

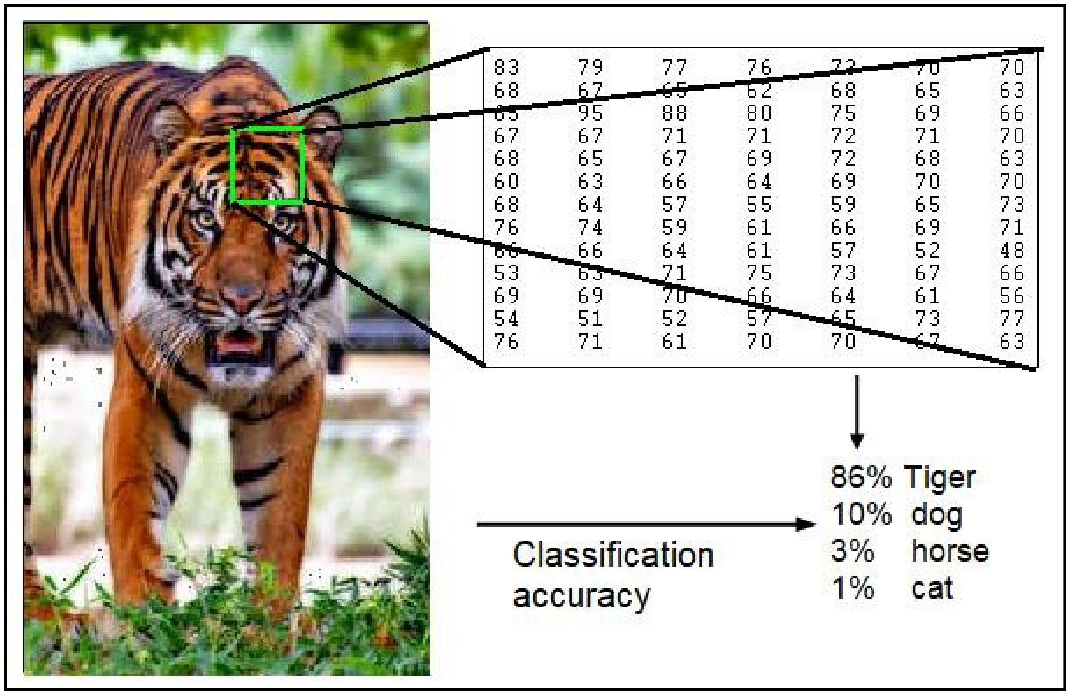
Deep learning was chosen for this project over other Machine Learning techniques due to its capabilities to abstract essential features and make predictions. Since Deep Learning technique is based on NNs, among NN types, CNN was selected for image classification and speech recognition. The reason behind the selection was the manner in which it processes images and audio for classification. CNN is a type of artificial NN which is used to divide a larger component into subcomponents such as a small pattern, curve in an image or an outline of a subcomponent in the image [36]. In comparison, RNNs process exactly opposite of how CNNs do - they combine minor components to reconstruct a larger structure [29]. Once the CNN learns to recognize these subcomponents in the input then it learns to join all these components to construct and predict the actual larger objects such as face of a person or ball lying on a table. Moreover, we desired to provide audio and image input sequentially to our network, CNNs can handle such sequential input data due to their ability to build internal spatial relationships. CNN model developed in this thesis used transfer learning to minimize the learning time and lessen the complexity associated with real-time systems.

Computer Vision

Computer vision is a multidisciplinary field of technology dealing with highly complex computational formulation of images and videos. Computer vision system shows us how the computers deal with images and videos from the perspective of a human eye [47]. This captivating technique consists of approaches such as processing, acquiring, analyzing, and manipulating the content in the digital image or videos, extracting the unique features from the large set of imagery data from the actual world, and meaningfully using them to produce decisions [48, 49]. Computer vision have a wide range of applications such as image scene reconstruction, event detection, vehicle detection, track a moving object, image classification, object recognition, and image restoration. Furthermore, we examine the related work done in computer vision pertaining to the image classification.

Computer Vision for Image Classification

Image classification problem is a one of the principal problems in Computer Vision, which has a variety of applications. The typical approach used in image classification problems is the data-driven approach. The reason behind this is it considers accumulating the training data before anything else. To exemplify a simple problem, consider the image classification model shown in Figure 10. The problem is to predict the animal by classifying among a fixed set of animal classes: tiger, dog, horse, and cat. The number matrix on the right reveal what the computer sees while classifying. The integer numbers in the matrix are extracted by the classifier based on the pixel values of the selected rectangular section in the image (green rectangle in the



83	79	77	76	73	70	70
68	67	65	62	68	65	63
85	95	88	80	75	69	66
67	67	71	71	72	71	70
68	65	67	69	72	68	63
60	63	66	64	69	70	70
68	64	57	55	59	65	73
76	74	59	61	66	69	71
66	66	64	61	57	52	48
53	63	71	75	73	67	66
69	69	70	66	64	61	56
54	51	52	57	65	73	77
76	71	61	70	70	67	63

FIGURE 10. Image classifier that classifies the animal in the image. Image from Karpathy [50].

figure). This pixel values range from 0 to 255. So, the task is converting this set of million numbers into a single label named ‘tiger’. This classification task is quite trivial for human to do, but the computer vision system has few noticeable challenges that are prevalent during the execution, such as perspective variation, scaling variation, deformation, partial visibility, illumination variations, and background cluttering. Examples of these challenges are demonstrated in Figure 11. The image classification task is usually formulated as: First capture input, next is learning the model (or training), and finally evaluating the quality of the model. Related work to computer vision and image classification is reviewed in Victor et al. [51]. To learn more about image classifier, we next review the nearest neighbor classifier called k-nearest neighbor.

K- Nearest Neighbor

K-Nearest Neighbor is a non-parametric algorithm constituting an instance-based

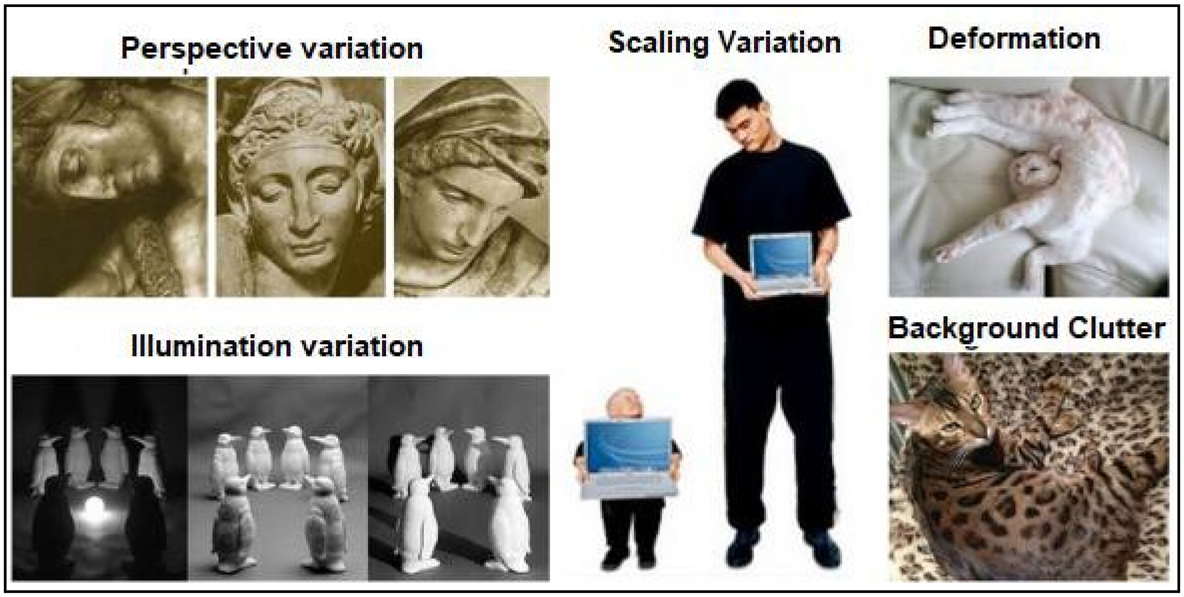


FIGURE 11. Computer vision challenges during image classification. Image from Karpathy [50].

learning type [52]. An image classifier is built with the help of k-NN algorithm and neural networks. The image classified is characterized as a featuring vector and this vector addresses to the label specifying the recognized data (e.g. object, audio or text) from the input. The duo of the characterized feature vector and the classified label forms a labeled dataset. Then, the newly formed dataset is fed to k-NN algorithm as input. The k-NN algorithm builds the classifier network and trains it based on the feature vector dataset of different input data. The output is dependent on whether *k*-NN is deployed for classification or regression [53]. For *k-NN classification*, output is an object of a class. Majority of its neighbors classify the class object with another assigned object frequently used among its *k* nearest neighbors. For *k* = 1, the object is merely assigned to the class of that nearest neighbor. In case of *k-NN regression*, the value of an output is the property value for the class object. This value refers to the average of the values of its *k* nearest neighbors. Based on the type of data to be worked with, the best value of *k* is chosen, *k* being a small, positive integer. Larger value of *k* would have boundaries between the classes less distinct, however, it helps to alleviate noise effect during the classification process. The k-NN algorithm is one the simplest among all Machine Learning algorithms. Related work is stated in Samworth [54].

Further, we review, in detail, the work done in the fields of speech recognition, science behind the snooker game, and robot manipulation in relation to our proposed research.

Speech Recognition

Speech recognition (or voice recognition) is an inter-disciplinary field in computational technology that is developed to recognize or translate a given speech command. Many speech recognition systems necessitate training where the user (speaker) provide separate words or sentences as speech commands to the system. The system examines the speech command based on a person's specific accent, tune or articulation pattern. Speech recognition is an important aspect in the human-machine interface technology. Not surprisingly, it has recently been popular in Machine Learning technologies leading to integration of voice-user interface in real-world applications and few of such developments are demonstrated in Baker [55]. These applications include voice dialing, information search, assisting visually impaired patients, speech-to-text processing, data entry, and domestic appliance control [56]. Speech recognition process is framed in order of input speech, speech preprocessing, feature extraction, speech recognition, and the output [57]. Now we analyze the role played by neural networks in speech recognition process.

Speech Recognition Using Neural Networks

There are several speech classifiers, but we assess the approach that uses Hidden Markov models (HMMs) [58], since it solves time-based and acoustic problems parallelly considering statistical data. Like many other neural networks, neural-network based speech recognizers typically consists of multi-layered hierarchies [59]. In this type of recognizers, the initial layer is made up of a group of HMMs, which are responsible for recognizing the speech command. The second layer utilizes the output of previous layer as input and extracts special features during recognition. The third layer in the system uses the output from preceding layer and spots the patterns in the speech recorded. Finally, the fourth layer recognizes the correct spoken word and the model returns the word label as output.

Then we investigate the speech spectrograms [60]. It is a representation that enables a user to understand the information content in the speech commands. A spectrogram characteristically has all the informative content that a machine model requires for processing. This helps the model to formulate the data for efficient training. Moreover, to increase the robustness of our model, we study the influence of background noises inevitable during the speech recognition progression [61]. More research-based work on speech recognition using neural networks is stated in Tebelskis [62] and Furui [63].

Snooker Game

Snooker (also known as billiards) is one of the most popular cue sports. There has been boost in advancements in snooker-based research. These research developments are majorly expected to serve the purpose of training and improving the snooker players. Moreover, there has been lot of research on computer-based snooker simulations which meticulously simulates the real-world snooker environment [64, 65]. The game also has gained noticeable interest from computer and Machine Learning scientists, who plan to incorporate artificial intelligence to articulate strong game strategies [66, 67]. Snooker game is all about manipulating different colored balls into the pocket (also called pot) in a correct sequence. Further, we analyze the ysics, mathematical theory of spin, surface friction, and collision of balls in context of the game.

Physics Behind Snooker

The collision between snooker balls is governed by the physics [68-70]. The collision between two snooker balls is almost elastic, where the kinetic energy of the whole system is conserved before and after the collision. So, for the sake of convenience, one can assume the collisions between snooker balls to be completely elastic. Just like many other collisions, momentum is conserved during the impact. It is also assumed that there is no friction acting between the ball and the table. Also, all the snooker balls are regarded to have the same mass. We now briefly review the ball collision and the sweet spot here.

Collision of balls: To exemplify, we consider a general collision between two balls cue ball (A) and target ball (B) (Figure 12) and it is supposed that ball B is stationary initially (no

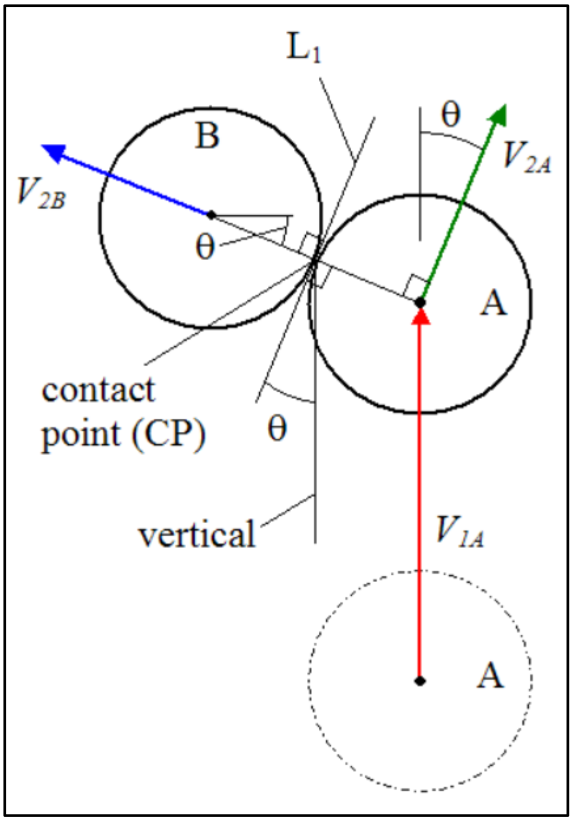


FIGURE 12. A collision of a cue ball (A) and target ball (B). Image from Real World Physics Problems [69].

velocity). After collision at contact point (CP), ball B advances in the direction of the line connecting the center of the balls, as shown in the figure. The reason behind this is ball A exerts an impulsive force normally to the surface of ball B, under the assumption that no friction acts between them. Thus, ball B rolls in the direction of this force. Moreover, after collision, ball A rolls in the direction perpendicular to that of ball B. However, this analysis does not hold true for the collisions that are head-on where (θ = 90°). This is because the total velocity of ball A is fully transferred to ball B. More in-depth analysis is presented in Real World Physics Problems [69]**.**

Sweet spot: Sweet spot is a point on the snooker ball where one can strike the cue stick to avoid friction between the ball and the table [69]. In other words, a ball is hit at the sweet spot to have pure rolling of the ball and no relative sliding or slipping. To demonstrate the sweet spot, we consider Figure 13 depicting the position of the cue stick at height *h*. This height *h* is determined such that no friction is produced at point P when the ball is hit by the stick. Brief derivation of the height h is discussed below. Free-body diagram of the ball-cue system is shown in Figure 13. Indicated notations in the figure are described as follows:

F is force the cue exerts on the ball when it strikes,

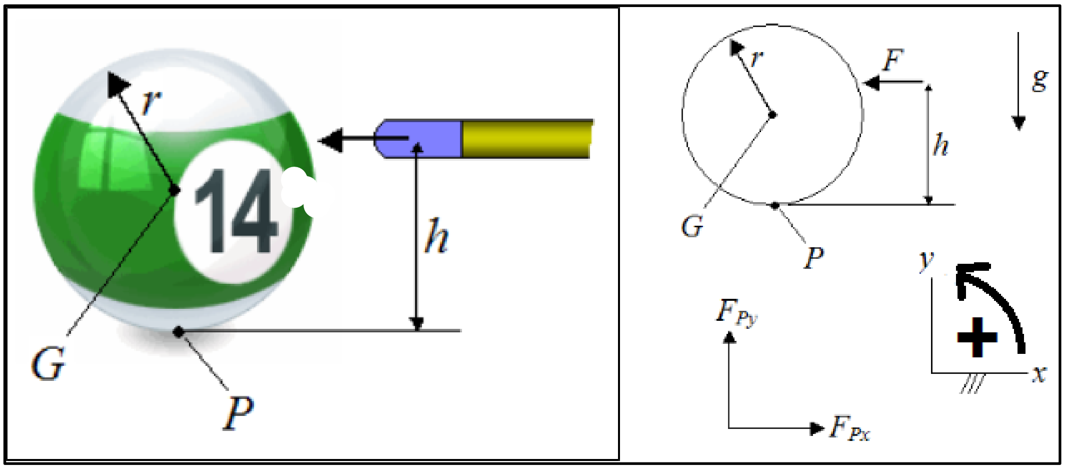


FIGURE 13. Sweet spot and free-body diagram of ball-cue system. Image from Real World Physics Problems [69].

h is the height at which sweet spot can be determined,

r is radius of the ball,

G is center of mass of the ball,

g is gravitational acceleration, 9.81 m/s^2,

P is the contact point of the ball with the snooker table,

F_{Px} (frictional force) is the *x*-component of the force exerted on the ball by the table, at P,

F_{Py} is the *y*-component of the force exerted on the ball by the table, at P.

Using Newton's second law of motion, force equation in *x* and *y* direction is

$$\sum F_x = ma_{G_x} \tag{1}$$

$$\sum F_y = ma_{G_y} \tag{2}$$

a_{G_x} and a_{G_y} are the accelerations in *x* and *y*-direction respectively,

Equation (1) becomes

$$F_{P_x} - F = ma_{G_x}$$

Since no frictional force is considered, $F_{Px} = 0$;

We get

$$-F = ma_{G_x} \tag{3}$$

From equation (2), a_{Gy}=0 as ball does not move in *y* direction

$$F_{P_y} - mg = ma_{G_y} = 0$$

So,

$$F_{p_y} = mg$$

Now the moment equation for rotation of a body about its center is given by

$$\sum M_G = I_G\alpha$$

where I_G is the moment of inertia of the ball about its center of mass, with its axis coming out of the plane of the page,

α is the ball's angular acceleration, and

$\sum M_G$ is the summation of all the moments about the center of mass,

Since we considered pure rolling of the ball, we have

$$\alpha = \frac{-a_{G_x}}{r} \quad \text{(negative sign indicates clockwise direction)}$$

Now the moment equation can be written as

$$F \cdot (h - r) = I_G \left(-\frac{a_{G_x}}{r}\right) \tag{4}$$

From equations (3) and (4),

$$h = \frac{I_G}{mr} + r$$

The moment of inertia for a solid sphere,

$$I_G = \frac{2}{5} mr^2$$

So,

$$h = \frac{7r}{5}$$

The ball is to be struck at this height to guarantee frictionless motion of the ball. Regardless of the power applied on the shot while striking at the sweet spot, the ball would have frictionless rolling. If the ball is hit above or below this height, relative slipping takes place due to lack of friction. A more detailed discussion of the sweet spot can be found in Real World Physics Problems [69].

Mathematical Theory Behind the Game

The mathematical aspect of the game is to understand the scientific theory behind the

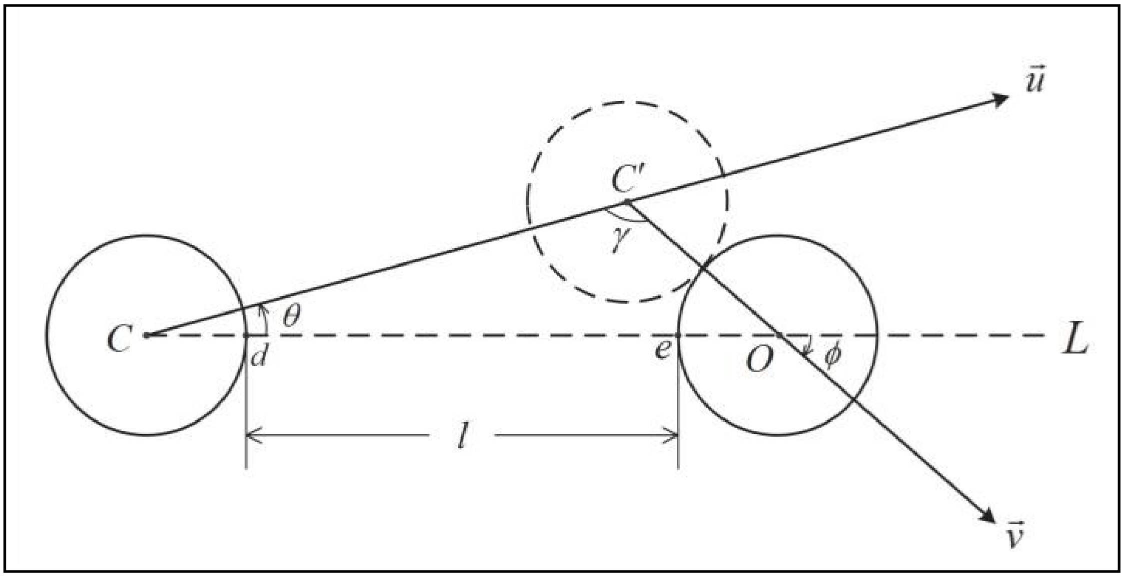


FIGURE 14. An oblique collision of cue ball and target ball. Image from Leckie and Greenspan [67].

motion of balls [71]. To illustrate, a general collision of cue and the target ball is depicted in Figure14, often referred as an oblique collision. The ball with center C and C` is the cue ball whereas the one with center O is target ball (or object ball). C' is the cue ball at the point of collision with the target ball. In most cases, centers O, C and C` form acute triangle, in some cases like in Figure 15 and 16, they form a straight line and right triangle respectively. In Figure 14, the angle formed by reference line L and the vector *v* is referred to as the shot angle *φ*.

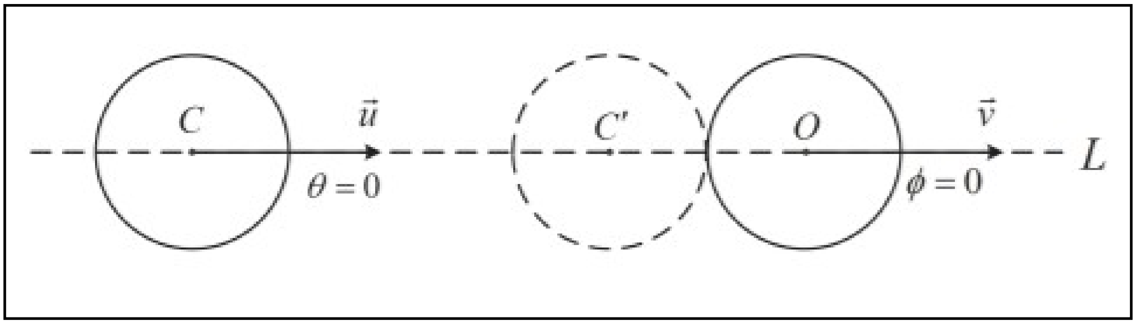


FIGURE 15. A normal collision of cue ball and target ball. Image from Leckie and Greenspan [67].

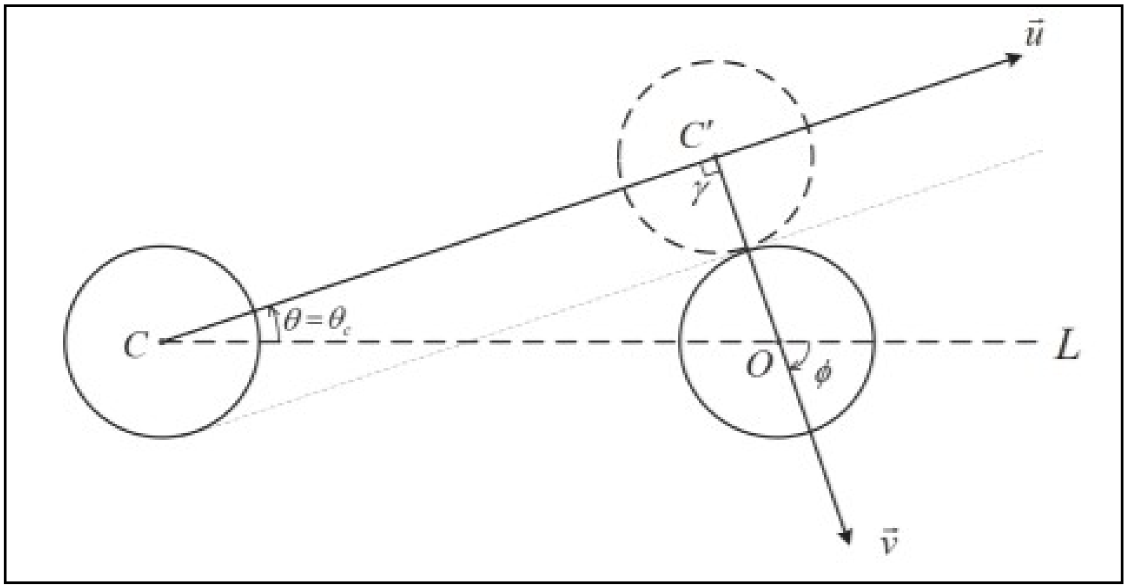


FIGURE 16. A tangential collision of cue ball and target ball. Image from Leckie and Greenspan [67].

Similarly, the angle θ formed by the vector u and base line L is called aiming angle.

Mathematical relation between these critical angles θ and φ is examined thoroughly. Also, we assume that a collision will occur, so the absolute aiming angle is always presumed to be smaller than the critical angle i.e., $|\theta| \leq \theta c$. Otherwise, in case of $|\theta| > \theta c$, no collision will occur. By applying mathematical knowledge of geometry, equations (5) and (6) are formulated, which represents the shot angle to be selected while striking the cue in case of Figure 14 and 16 respectively. No angle computation is necessary in case of Figure 15, as directly striking the ball would result into linear motion of the target ball.

$$\varphi = \theta - \sin^{-1} ((p + 1) \sin \theta) \tag{5}$$

$$\varphi_c = (\frac{\pi}{2}) - \sin^{-1}(\frac{1}{p+1}) \tag{6}$$

where p is the number of balls that can be pocketed in between the two balls.

These formulated equations are used to develop the computer algorithm for selecting the best shot. Detailed derivation and explanations of these equations can be found in Nadler [68].

Robot Motion Planning and Actuation

Robot motion planning [72] is a procedural technique to break down a desired continuous-task movement to obtain a smooth optimized motion of the robot while taking motion constraints into concern. A simple robot motion planning problem requires the robot to start the motion at a static point and perform the manipulation in its configuration space. To understand and program the manipulation of our JACO [73] robot, we referred to the manufacturer Kinova’s manual and its software development kit (SDK). The available SDK also provided the API, which helps the user to modify the code to manipulate and control the robotic arm in a desired manner.

CHAPTER 3

METHODOLOGY

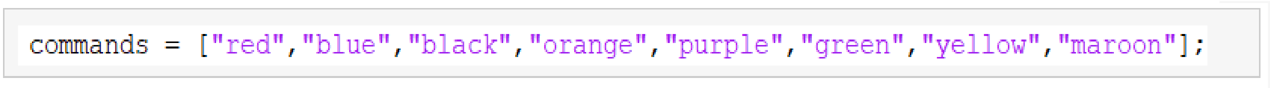
This research was accomplished in three significant phases: speech recognition, computer vision, and shot selection algorithm and robot actuation. For the first two phases, we used Deep Learning models to train the robot. Training of these models for both phases was performed using a pre-trained model and MATLAB. The second phase uses MATLAB’s Image Processing toolbox to determine the best shot. The last phase was conducted with the execution of the best shot by our robotic arm. Detailed explanations of each phase are offered in this chapter.

Speech Recognition

This phase describes how we trained the Deep Learning model that detects the presence of speech commands in an input audio. Our model uses the Speech Commands Dataset [74] to train a neural network to recognize a given set of commands.

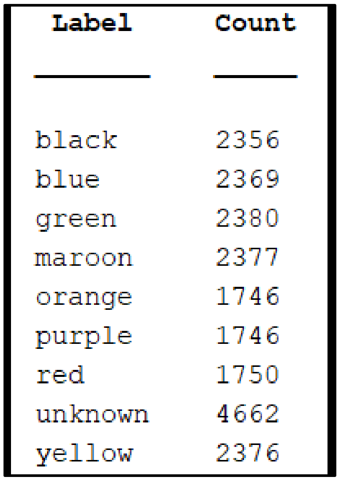
Specifying Words for Recognition

Command words which the system was desired to recognize were specified (color of the ball to be pocketed) (Figure 17). All other words not among commands were labelled as *unknown*. The reason for labelling these words is that these words must approximate the distribution of all other words. Each word (often referred to as a class in machine-learning terminology) had more than 1700 audio examples (Figure 18) which helped us to achieve significant training accuracy. Each word was recorded as an audio example in different scenarios such as different accent, varying speech tones or inconsistent audio pitch.



```
commands = ["red","blue","black","orange","purple","green","yellow","maroon"];
```

FIGURE 17. Specifying which words are to be recognized from the dataset.



```
  Label       Count
_______       _____

black         2356
blue          2369
green         2380
maroon        2377
orange        1746
purple        1746
red           1750
unknown       4662
yellow        2376
```

FIGURE 18. Number of audio examples (files) for each word/class.

Computing Speech Spectrograms

To formulate the data for effective training of a CNN network, speech waveforms were converted to log-bark auditory spectrograms [75]. Spectrograms for all the training, validation, and test sets were computed using the auditory spectrogram function. To minimize the number of spikes present in the dataset, a logarithm of the spectrograms was taken with a small offset *epsilon*. Computed spectrograms were then plotted along with the waveforms of a few training audio examples. Labels of the corresponding sample were marked on top of the waveform. Waveforms and speech spectrograms for classes yellow, blue, and red are shown in Figure 19. DL neural networks take less training time when the fed inputs are considerably smoother in distribution and normalized. To ensure smooth distribution of the data, a histogram of the pixel values of the audio training data was visualized (Figure 20).

Consideration of Background Noise

The developed DL network should not only be able to detect the uttered words, but also be able to identify if a word is spoken with some background noise. To ameliorate the efficiency of our network, we created audio samples featuring different kinds of background noise.

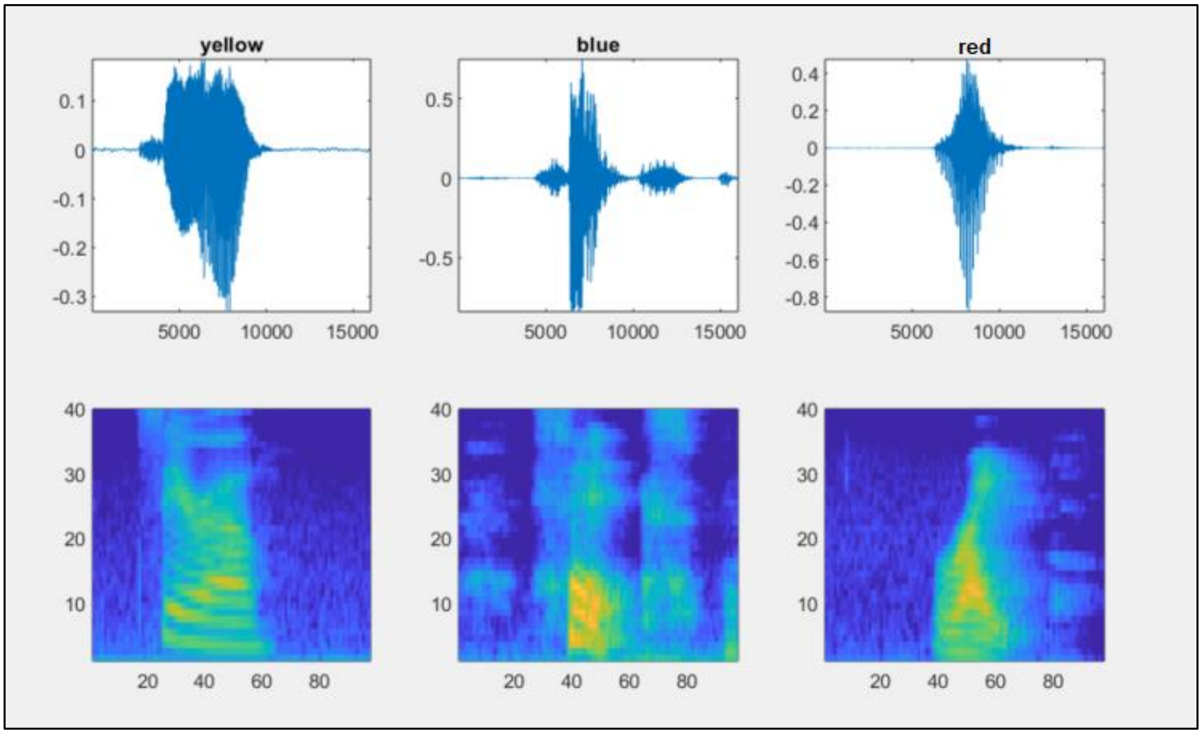


FIGURE 19. Waveforms and speech spectrograms for words yellow, blue and red.

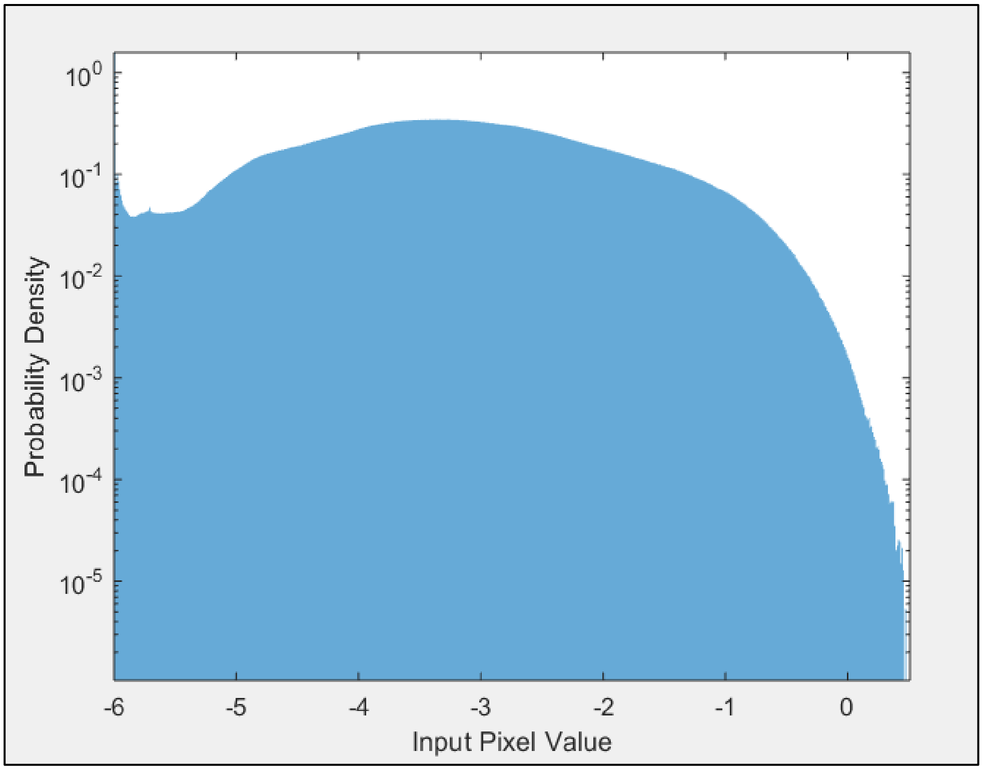


FIGURE 20. Histogram of the pixel values of the training data.

Background noise samples were mixed with speech samples to improve the robustness of the model. This enabled our model to identify any unwanted background noise during training, validation, or testing. Distribution of different classes during training and validation is shown in Figures 21 and 22. The sole purpose of plotting these distributions is to visualize the evenness of the distributed samples for each class.

Defining DL Neural Network Architecture

The first stage in speech recognition module is the convolution neural network which interprets the audio input in a computable format. A convolutional neural network architecture

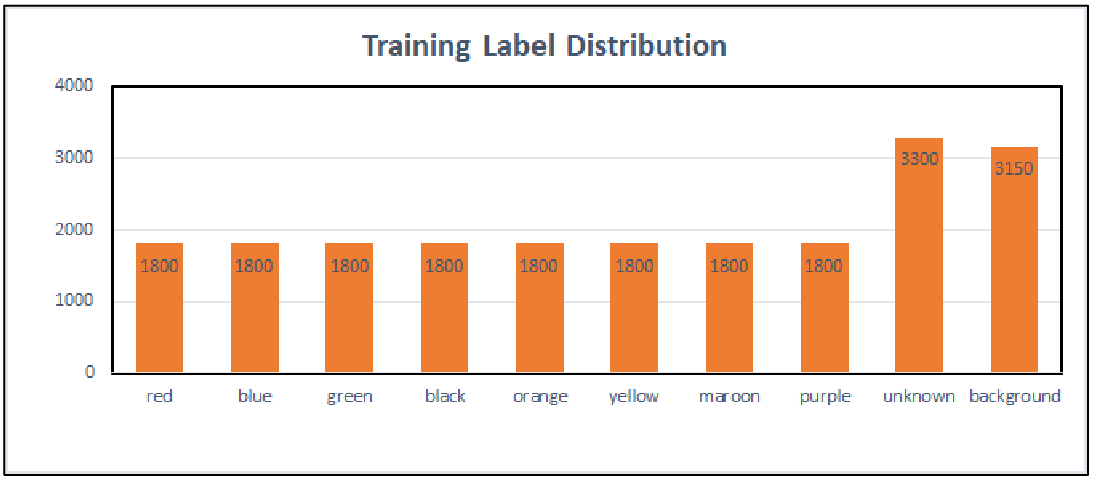


FIGURE 21. Distribution of different word labels during training.

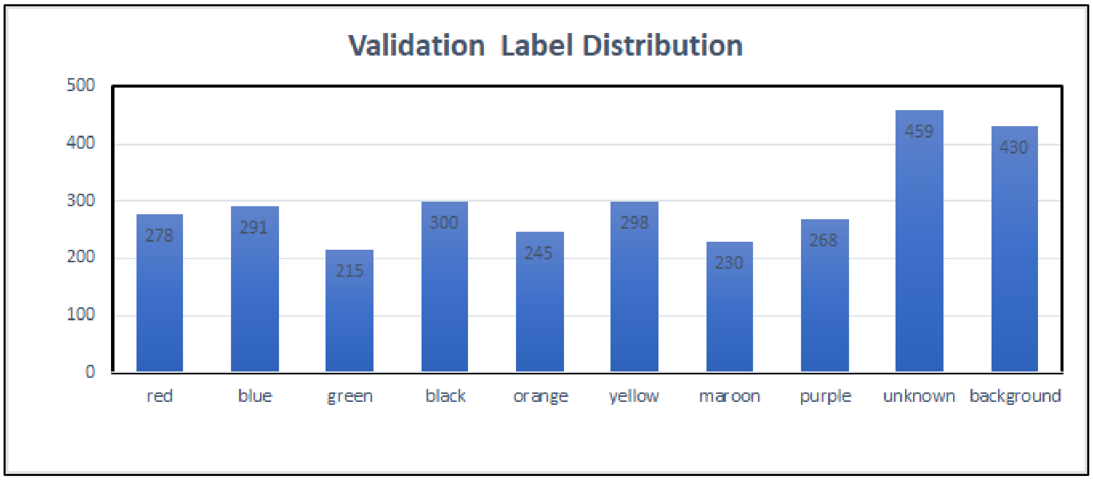


FIGURE 22. Distribution of different word labels during validation.

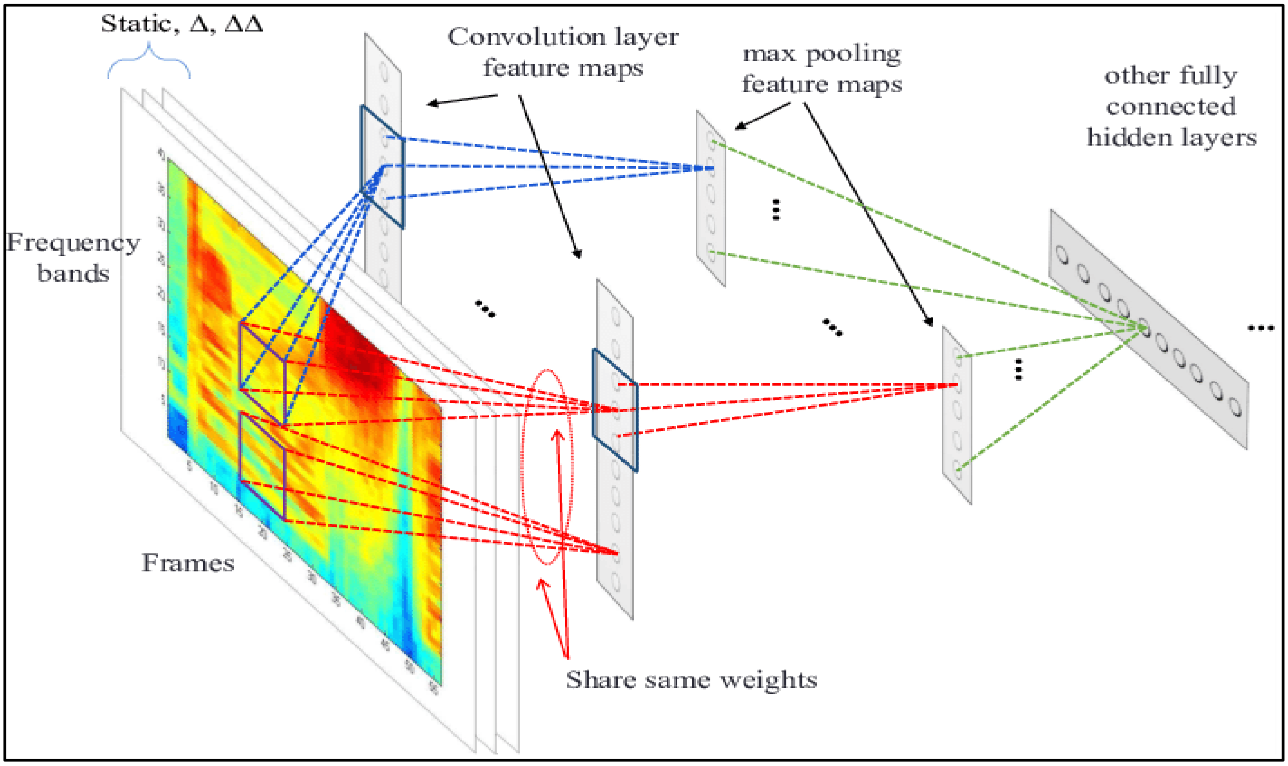
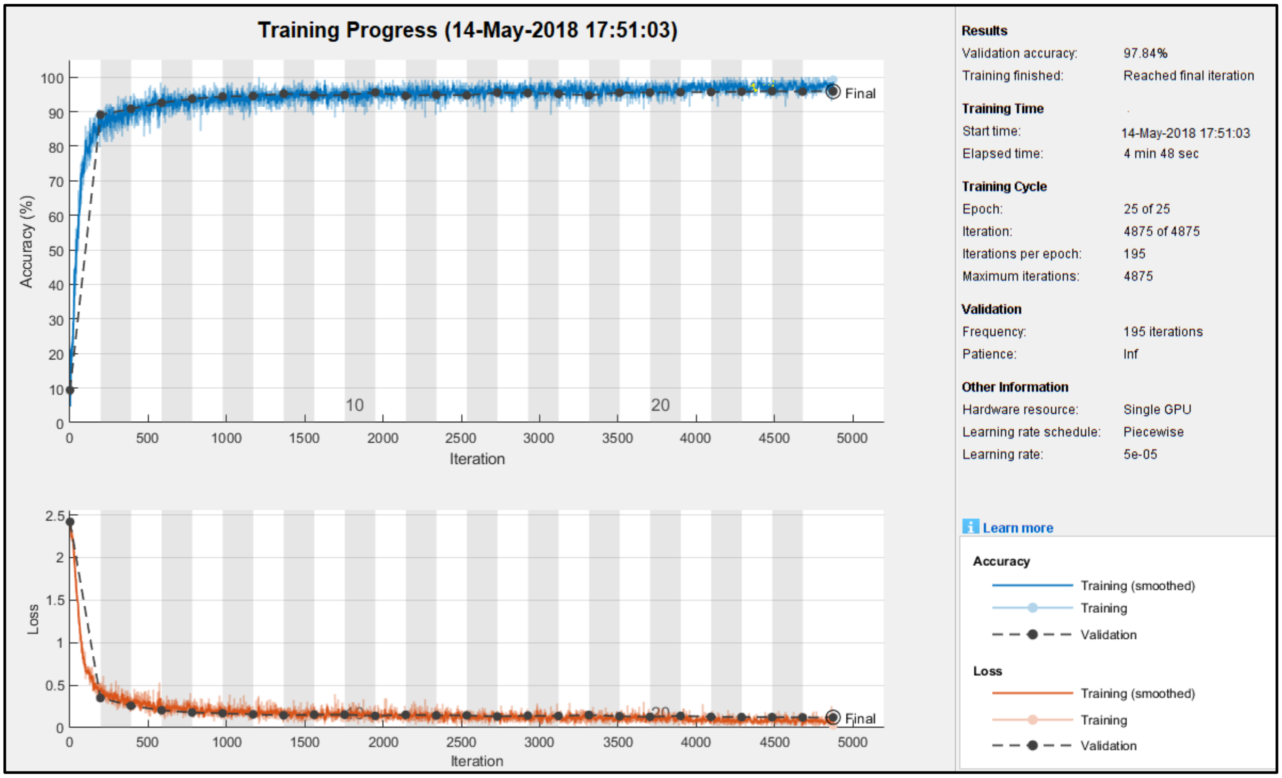


FIGURE 23. CNN structure for speech recognition. Image from Abdel-Hamid et al. [77].

[76] is constituted by an array of layers, and each layer is defined as required. The recorded audio waveforms are converted to time-frequency domain and fed into the layers of CNN, as demonstrated in Figure 23 [77]. Max pooling layers are used to lower the sampling frequency; this reduction is significant in reducing the number of parameters in the fully connected layer. A small dropout is added to the inputs to the layers with highest number of parameters; this is done to prevent the network from memorizing some specific features from the training datasets. In fact, these layers are the convolutional layers with the highest number of filters.

Evaluating the Trained Model

The model achieved 97.84% accuracy during the training process with a small amount of training and validation errors (1.4356% and 3.2432% respectively). A computer with a GPU is used to make the training process faster. Validation accuracy and data loss for the mode while training for 4,875 iterations is shown in Figure 24. Following the training process, a confusion



Results

Validation accuracy: 97.84%

Training finished: Reached final iteration

Training Time

Start time: 14-May-2018 17:51:03

Elapsed time: 4 min 48 sec

Training Cycle

Epoch: 25 of 25

Iteration: 4875 of 4875

Iterations per epoch: 195

Maximum iterations: 4875

Validation

Frequency: 195 iterations

Patience: Inf

Other Information

Hardware resource: Single GPU

Learning rate schedule: Piecewise

Learning rate: 5e-05

FIGURE 24. Validation accuracy and data loss during the complete training process.

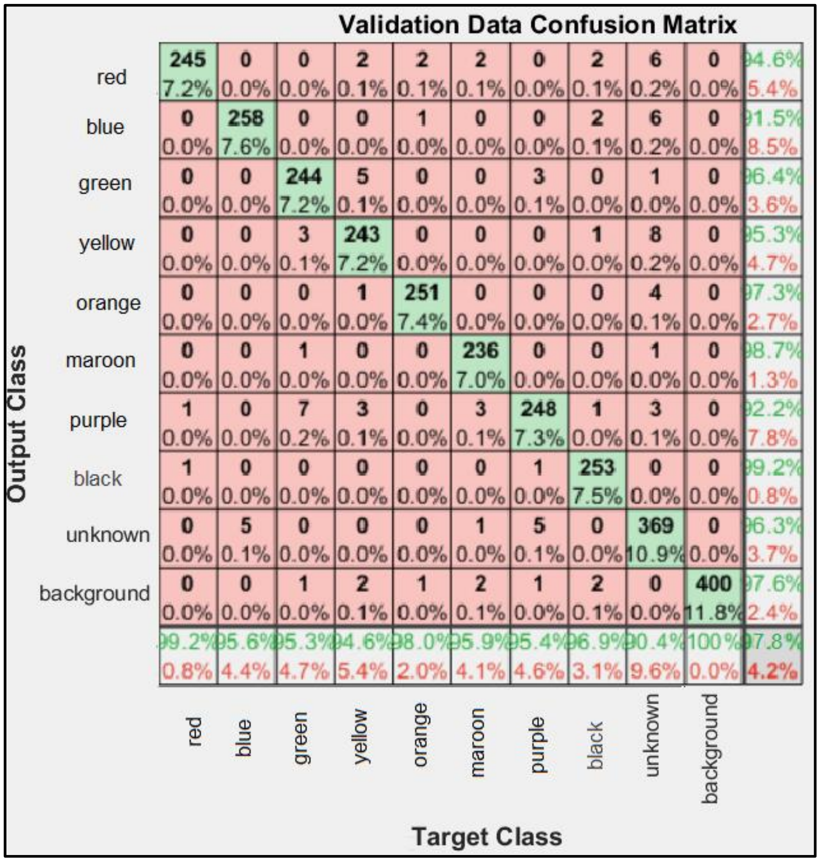
matrix is plotted to evaluate and obtain a better idea of our classification model and to know the factors that prevent accurate predictions (Figure 25). A confusion matrix provides a summary of recognized results on a classification problem [78]. This matrix confirms the accuracy of the model.

Feeding Input: Detect Commands from an Audio Device

The trained model was tested by feeding streaming audio through a microphone. One of the words from different classes was spoken and a figure representing the waveform of the spoken command was shown along with label of the recognized word (color 'red' in Figure 26).

Computer Vision

This phase of our experiment was also conducted using DL's convolutional neural network for image classification. Transfer learning, as aforementioned in the literature review,



Validation Data Confusion Matrix

Output Class \ Target Class	red	blue	green	yellow	orange	maroon	purple	black	unknown	background	
red	245 7.2%	0 0.0%	0 0.0%	2 0.1%	2 0.1%	2 0.1%	0 0.0%	2 0.1%	6 0.2%	0 0.0%	94.6% 5.4%
blue	0 0.0%	258 7.6%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	2 0.1%	6 0.2%	0 0.0%	91.5% 8.5%
green	0 0.0%	0 0.0%	244 7.2%	5 0.1%	0 0.0%	0 0.0%	3 0.1%	0 0.0%	1 0.0%	0 0.0%	96.4% 3.6%
yellow	0 0.0%	0 0.0%	3 0.1%	243 7.2%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	8 0.2%	0 0.0%	95.3% 4.7%
orange	0 0.0%	0 0.0%	0 0.0%	1 0.0%	251 7.4%	0 0.0%	0 0.0%	0 0.0%	4 0.1%	0 0.0%	97.3% 2.7%
maroon	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	236 7.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	98.7% 1.3%
purple	1 0.0%	0 0.0%	7 0.2%	3 0.1%	0 0.0%	3 0.1%	248 7.3%	1 0.0%	3 0.1%	0 0.0%	92.2% 7.8%
black	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	253 7.5%	0 0.0%	0 0.0%	99.2% 0.8%
unknown	0 0.0%	5 0.1%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	5 0.1%	0 0.0%	369 10.9%	0 0.0%	96.3% 3.7%
background	0 0.0%	0 0.0%	1 0.0%	2 0.1%	1 0.0%	2 0.1%	1 0.0%	2 0.1%	0 0.0%	400 11.8%	97.6% 2.4%
	99.2% 0.8%	95.6% 4.4%	95.3% 4.7%	94.6% 5.4%	98.0% 2.0%	95.9% 4.1%	95.4% 4.6%	96.9% 3.1%	90.4% 9.6%	100% 0.0%	97.8% 4.2%

FIGURE 25. Confusion matrix.

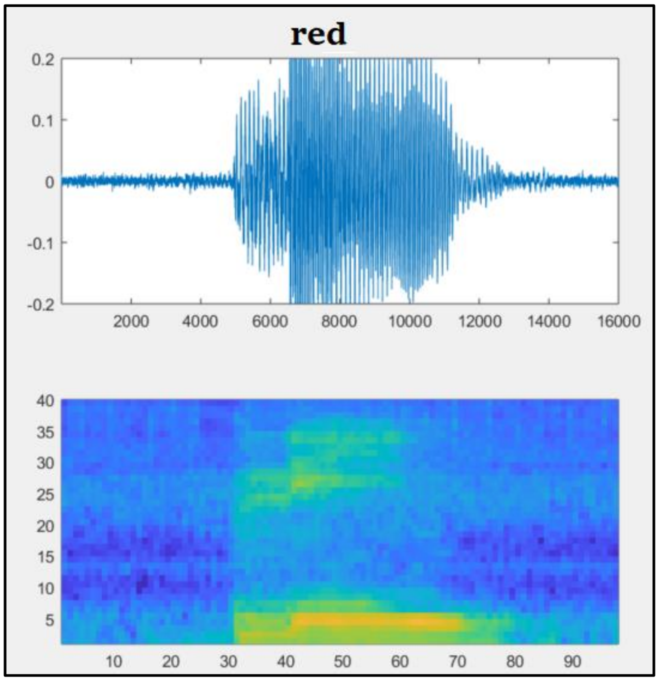


FIGURE 26. Detecting speech command from the streaming audio.

a technique in which an already trained model is used for solving similar classification problems. We used this technique by loading a pre-trained neural network model with VGG architecture [79, 80]. VGG is a convolutional neural network model and its architecture is shown in Figure 27. The model has 92.74% accuracy in ImageNet [81], a dataset of over 14 million digital images belonging to more than 1000 classes. The final layer of the VGG16 network is a function called SoftMax, which detects the input in one of the 1,000 ImageNet classes depending upon the 4,096 features learned by the preceding layer (fully connected + ReLu layer output). For this study, the prominent features were extracted after the fully connected + ReLu layer, and the SoftMax function was substituted by the model trained using the k-NN algorithm [52, 53]. Before initiating the training process, calibration of the camera was performed to avoid any external hindrance during the process. The camera executed the calibration function and the camera set the frame to capture only the table, as shown in Figure 28. Once the system was calibrated, the network was trained by showing it 800 images of each of the eight colored-balls on a snooker table through a USB camera and were resized to an image of size 224 x 224 x 3, which is the size of the input layer in the VGG network architecture. The training process

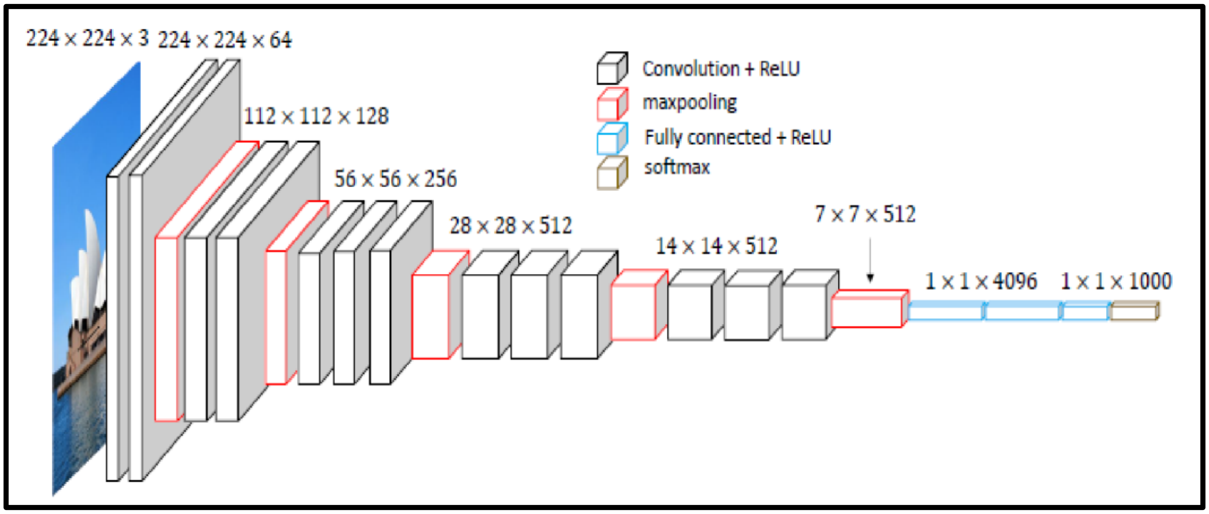


FIGURE 27. VGG16 convolutional neural network.

(Figure 29) took only a few minutes to complete, courtesy of a GPU-installed computer, as was used in the case of the previously discussed speech recognition training. Then, the live detection interface was turned on and the network detected the colored ball from the given speech command, as shown in Figure 30. Once this model was trained to classify the colored ball,

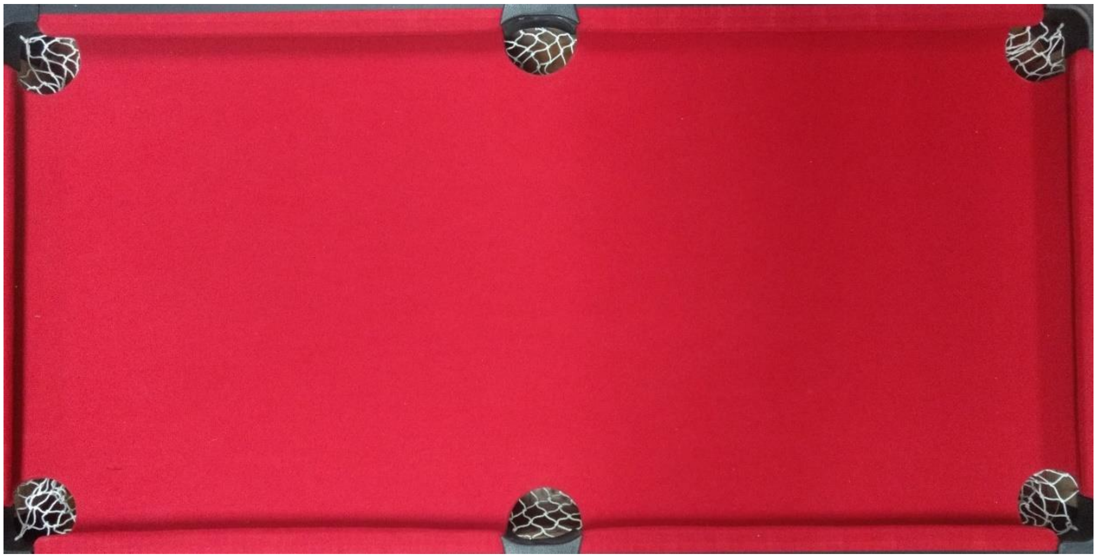
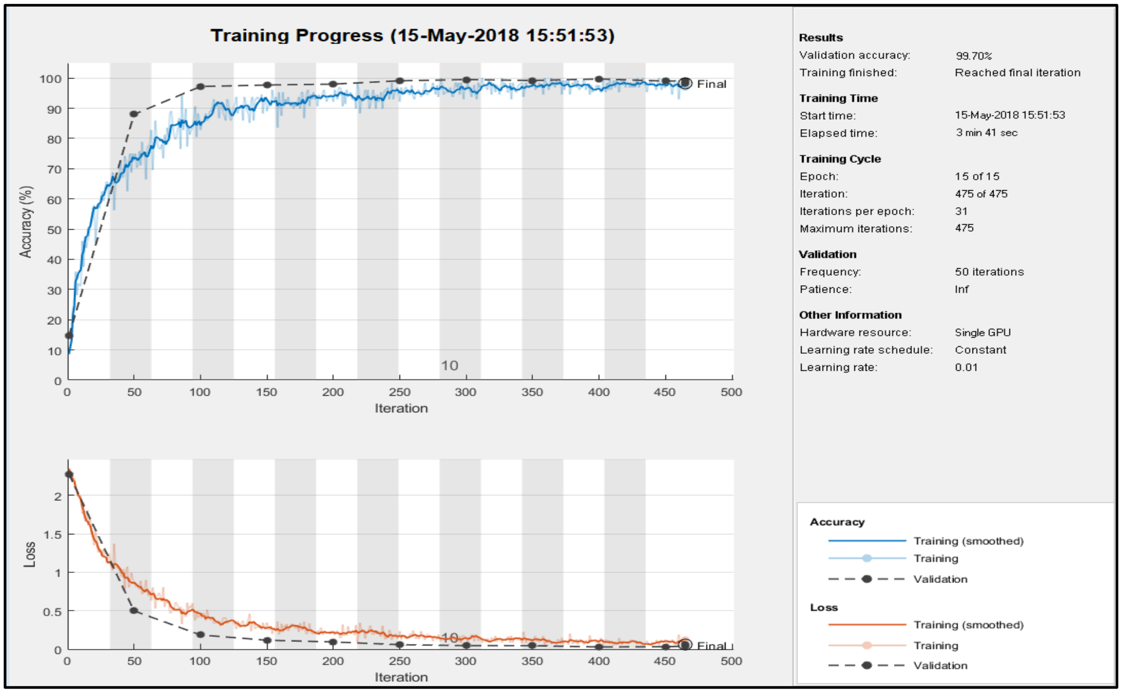


FIGURE 28. Calibration base frame.



Training Progress (15-May-2018 15:51:53)

Results
Validation accuracy: 99.70%
Training finished: Reached final iteration

Training Time
Start time: 15-May-2018 15:51:53
Elapsed time: 3 min 41 sec

Training Cycle
Epoch: 15 of 15
Iteration: 475 of 475
Iterations per epoch: 31
Maximum iterations: 475

Validation
Frequency: 50 iterations
Patience: Inf

Other Information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.01

FIGURE 29. Training progress for image classification.

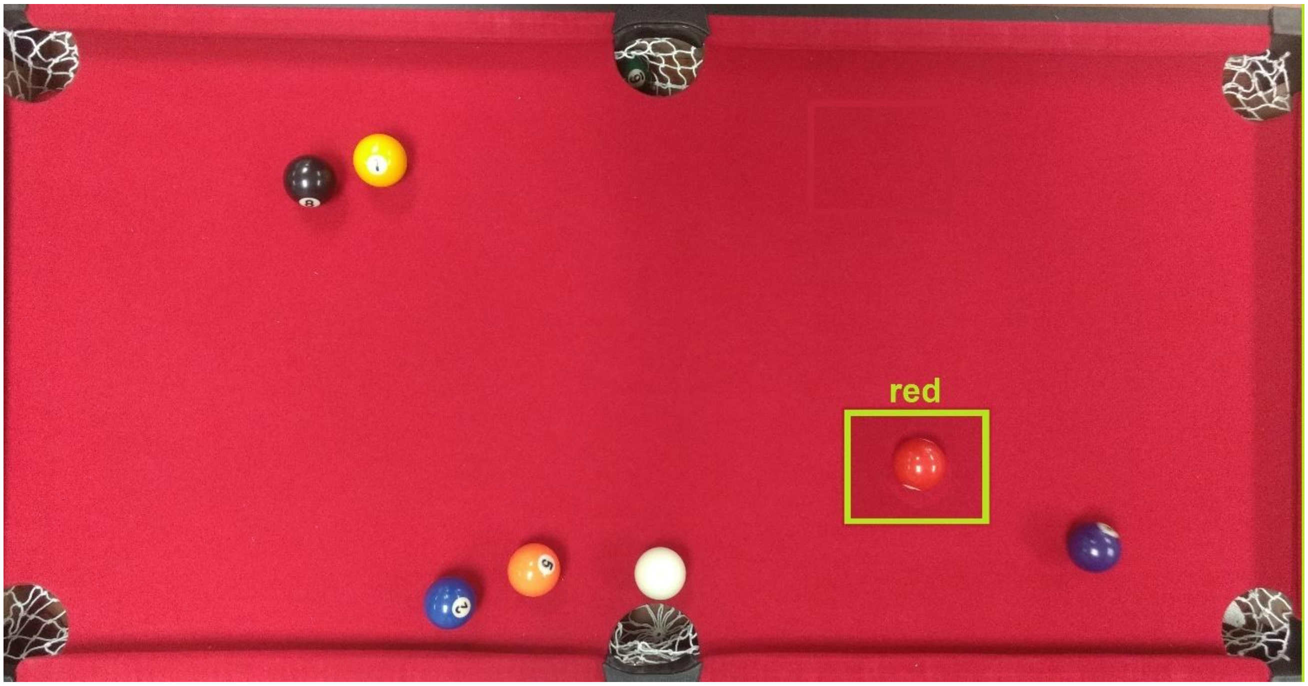


FIGURE 30. Trained model classifies the target ball.

MATLAB's Image Processing toolbox helped complete the remainder of this phase. The first priority in image processing was to detect the pockets through a user-defined function, *pocket detector*, and pull out the static region of interest (Figure 31). This function was created using

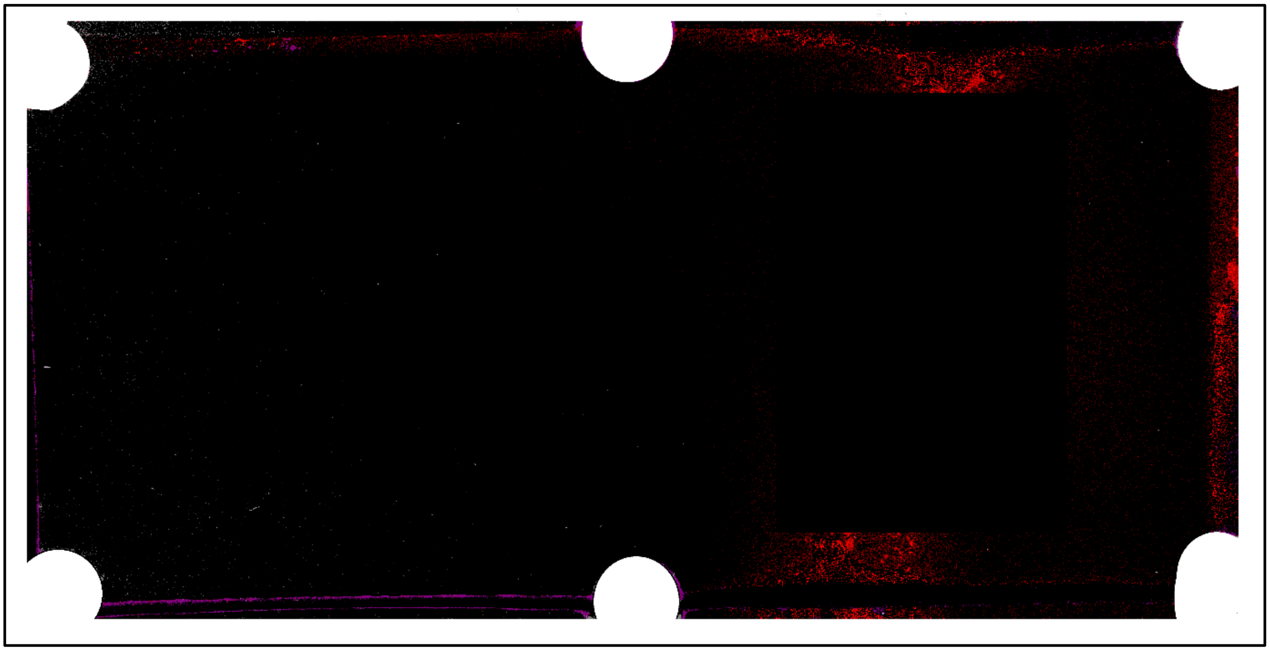


FIGURE 31. Static region of interest.

‘*imfindcircles*’ function available from MATLAB’s toolbox, to detect the pocket by recognizing circular area of the pockets. If the *imfindcircles* function fails to detect all the pockets, the function ‘*approx_hole*’ is executed which locates the pockets by approximating hole locations based on symmetrical geometry of the table. Sometimes our model located balls that were already in the pocket. The region of interest is significant because it ensures that the system only accounts for the balls in the table’s playing area. The toolbox runs one of its built-in functions to locate the pockets on the table as well as the centroids of the target and cue balls. Then the function to encircle the recognized color-ball and the cue ball is executed. After locating the target ball, cue ball, and the pockets, we run the user-defined function to determine the best shot for pocketing the target ball. This function contains the algorithm to interpret the best (optimal) shot. Subsequently, the optimal/best shot is selected through the shot selection algorithm (Figure 32). There are very few chances that the algorithm would not generate a feasible shot, but if it does, the trial would be regarded as a failed trial. The shot selection algorithm was developed

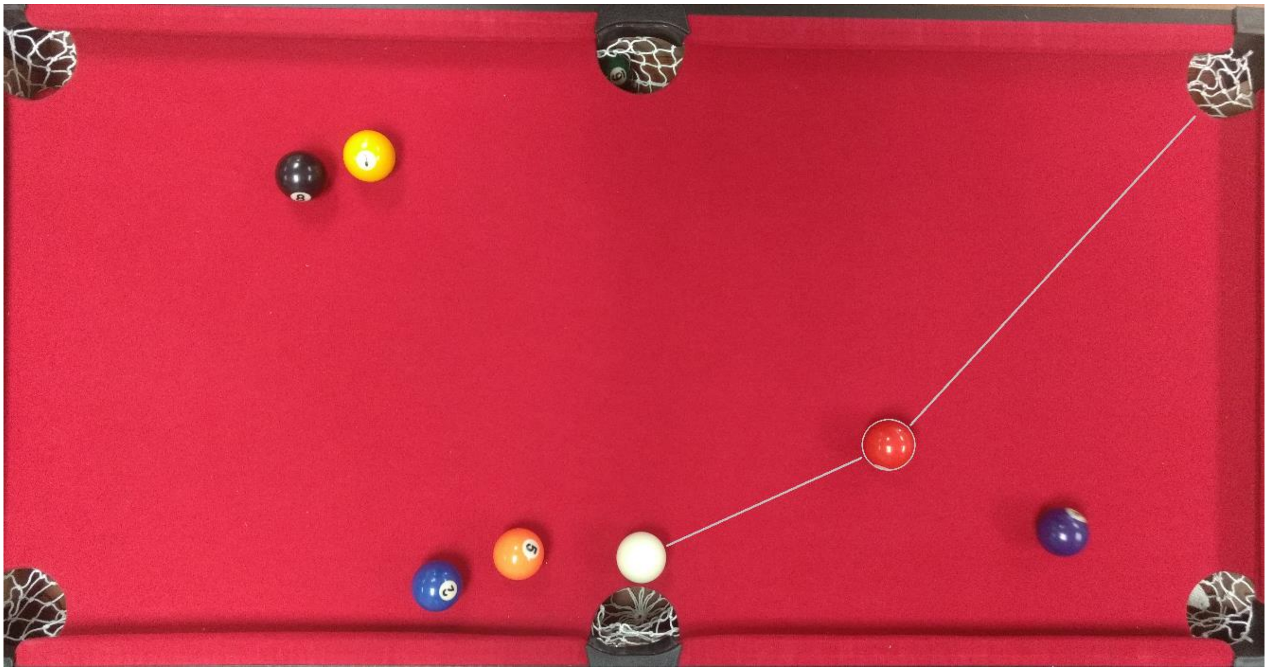


FIGURE 32. Target and cue balls encircled, and optimal shot is selected.

according to the physics and other scientific properties behind this game [7]. The collision between the balls was assumed to be perfectly elastic. In addition, to avoid development of frictional forces between the ball and table, our robot was programmed to strike the cue ball only on the sweet spot, which was discussed earlier in the literature review. This helped to make better shot selections.

Robot Actuation

For this experiment, a Kinova JACO robotic arm [71] with 6 DOFs was used. It is shown in Figure 33. The robot system is equipped with a USB camera for computer vision ability and the robot's end-effector holds a single-acting spring-return pneumatic cylinder. The mounted cylinder is designated to shoot the cue ball upon actuation. The executable code to actuate the robotic arm was developed with the API developed by Jaco SDK and related libraries using C++ language. The developed code included a function that performs the robot's motion planning. Motion planning function was written based on cartesian control method. To start with motion

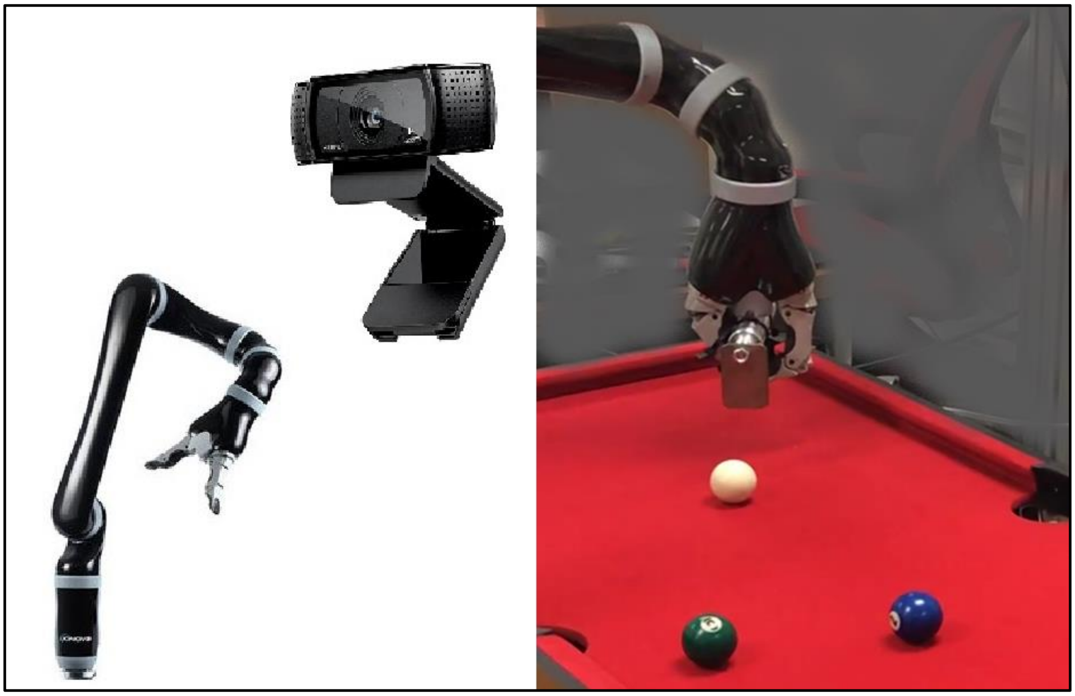
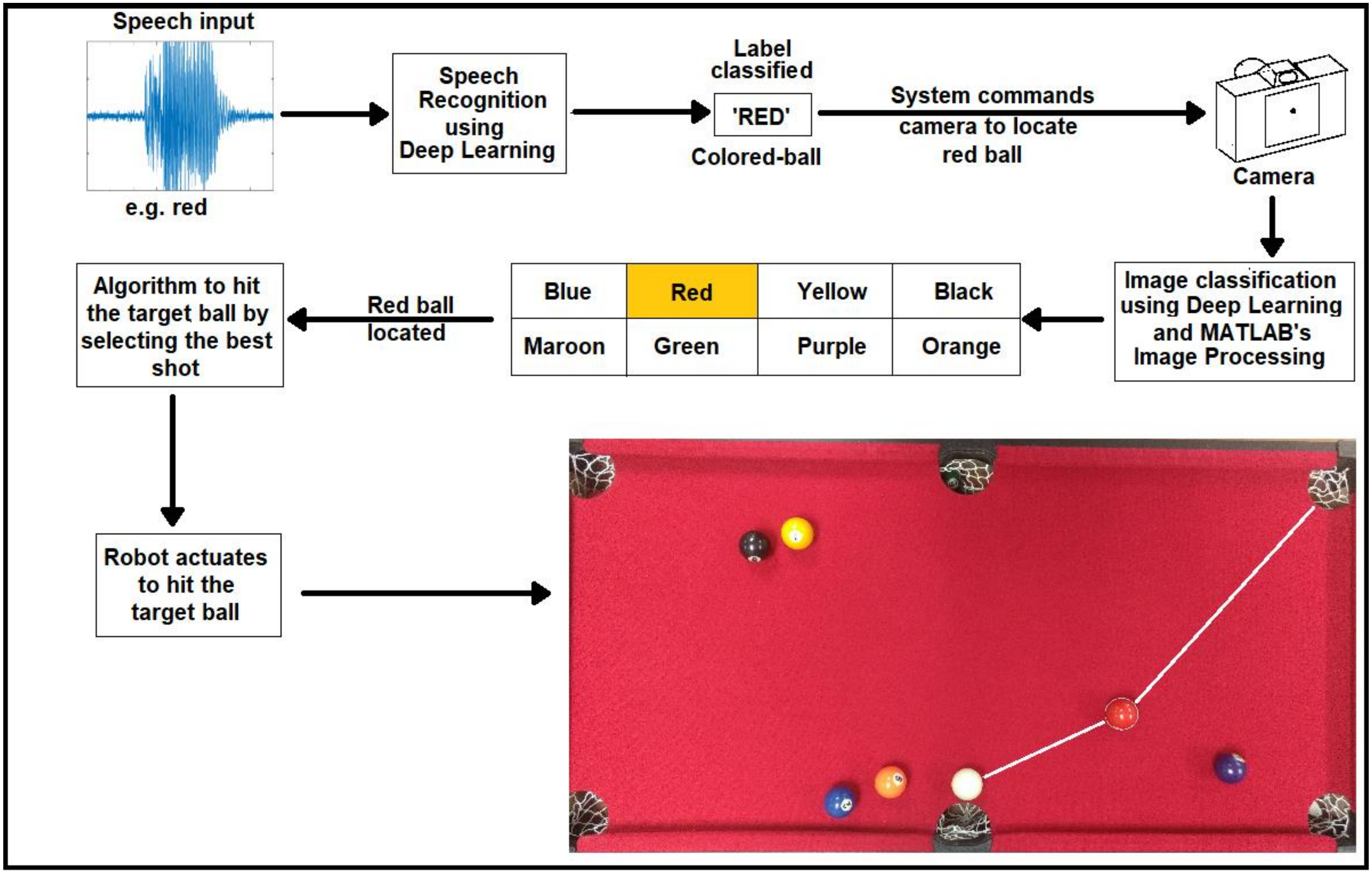


FIGURE 33. Kinova Jaco arm equipped with USB camera.

planning, the robot was set to reach a user-defined origin in *x*-*y* plane (*x*-*y* plane is parallel to plane of table's surface) at height of 1.4 times radius of the ball above the table. This height corresponds to the height at which the cue ball must be hit to prevent relative sliding or slipping, as discussed earlier in literature review. The code also checks for balls on the table before starting execution. There must be at least one color-ball on the table before the code is implemented. Once the end-effector reaches the origin, the code commands the robot to approach the cue ball. Then, to adjust itself to the optimal hit point, the robot rotates the end-effector by required angle to set up the best shot. The required angle is provided to it based on the algorithm that calculates the angle for a perfect shot. In addition, it is ensured that the end-effector reaches the sweet spot using angular feedback control function. This feedback function checks all the joint angles of the robot right before striking the ball. The pneumatic cylinder is actuated 800 milliseconds after the end-effector reaches the sweet spot.

Figure 34 describes the end-to-end architecture of the proposed application to encapsulate our research framework at a glance. The experiment was performed with consistent illumination of the surrounding throughout the process to avoid any lapse during image classification. Also, the user was allowed to speak any color of the ball among those on table. During testing, different test scenarios were performed to test the robustness of the model. Examples of such scenarios were: placing two similar color-tone balls (such as red and maroon) together to check image classification, speaking same color of ball in different accent to check speech recognition, optimal shot success and failure rate. Trials in which practically no shot was possible were considered as failed trials. Foul shots were also regarded among failed trials category. The foul shots principally include shots in which the target ball is not pocketed. The execution time for the experiment was also reported for each test sample and the time taken by the robot to pocket the ball accurately in the optimized shot was reported as the average time for execution. The optimal shot was decided based on the accuracy while striking the target ball, smooth trajectory of the robot, and color-ball classification accuracy.


Speech input
e.g. red
Speech Recognition using Deep Learning
Label classified
'RED'
Colored-ball
System commands camera to locate red ball
Camera
Image classification using Deep Learning and MATLAB's Image Processing
Red ball located
Algorithm to hit the target ball by selecting the best shot
Robot actuates to hit the target ball

Blue	Red	Yellow	Black
Maroon	Green	Purple	Orange

FIGURE 34. End-to-end architecture of the application in execution.

CHAPTER 4

RESULTS

The goal of this research was to integrate the decision-making capability in manipulative robots to perform a task in dynamic environments. This integration is greatly significant because the robot can accomplish a task with methods that mimic human. In this study, we used a Deep Learning algorithm for real-time speech recognition. The model obtained 97.84 percent validation accuracy for the training dataset. The real-time plot depicting the accuracy of the model is shown in Figure 35. We also used Machine Learning algorithms, CNN and k-NN, for real-time image classification. 100% accuracy was achieved for colored-ball image classification algorithm for k = 5. The performance of image classification model for different values of k is shown in Table 2. The accuracy of both the models mentioned here confirms the efficiency of the models when they are implemented individually. Both models were then executed one after the other, first speech recognition then image classification. We termed this combined model as integrated model. To evaluate the performance of the integrated model, we performed the

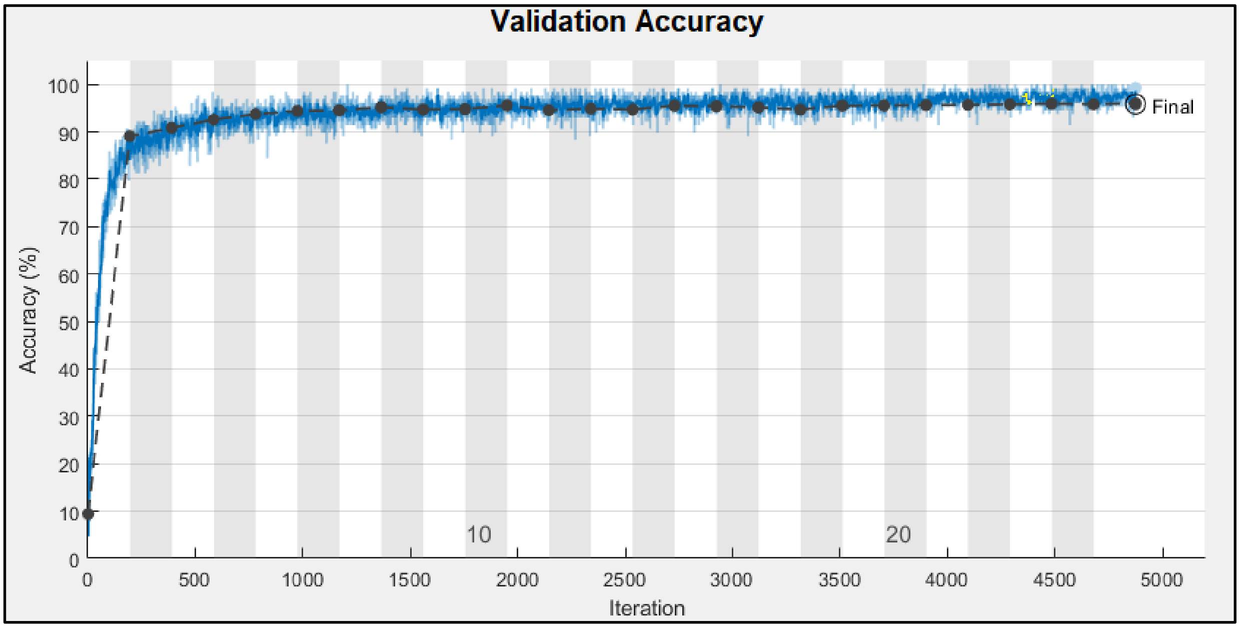


FIGURE 35. Accuracy plot for speech recognition model.

TABLE 2. K-NN Model Performance for Different Values of K

K-value	Classification Accuracy
1	77%
2	84%
3	80%
4	89%
5	**100%**
6	91%

experiment repeatedly to collect 2,000 test samples and its performance were reported. The model classified the color-ball 1,972 times correctly during the experiment and the model's accuracy was reported as 98.60%. The performance for the integrated model in terms of accuracy and loss is shown in Figures 36 and 37, respectively. Consequently, we tested the model's effective accuracy by testing the robot's ability to pocket the desired ball. Effective accuracy of the model would be considered only for successfully-completed trials. We manually define and report a successfully-completed trial as a trial for which a spoken-color ball is classified correctly, the robot hits the desired ball perfectly and pockets it properly. Trials, for which any one of these requirements was unmet, were regarded as a failed. In addition, if the shot selection algorithm failed to generate a feasible shot, the trail was observed as a failure. During the testing, the robot was able to complete 1154 successful trials over 1200 trials. These successfully-completed trials, as per our definition, constituted 96.17% effective accuracy of the developed model. The execution time was recorded from the moment speech input was provided through to the moment the target was accurately pocketed. We disregarded the execution time for the test samples that failed to perform the task correctly. The average execution time reported was 7.46 seconds. We also examined the 46 failed trials to improve the model and inaccurate end-effector position caused failure of 12 trials.

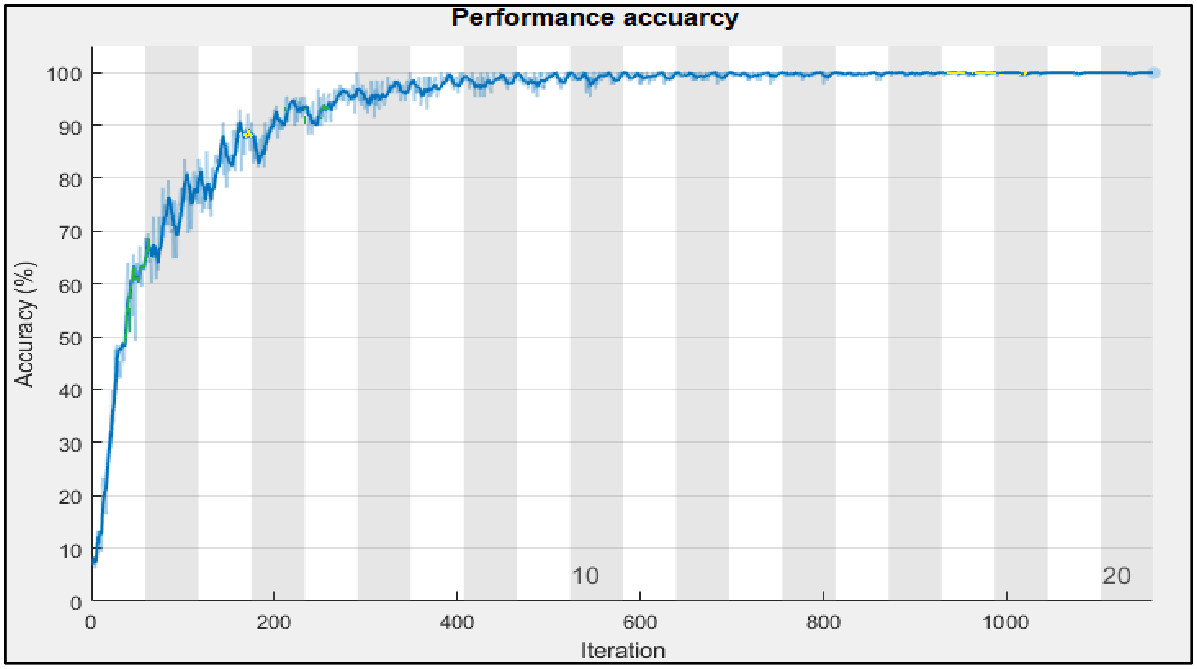


FIGURE 36. Performance accuracy during real-time execution of the integrated model.

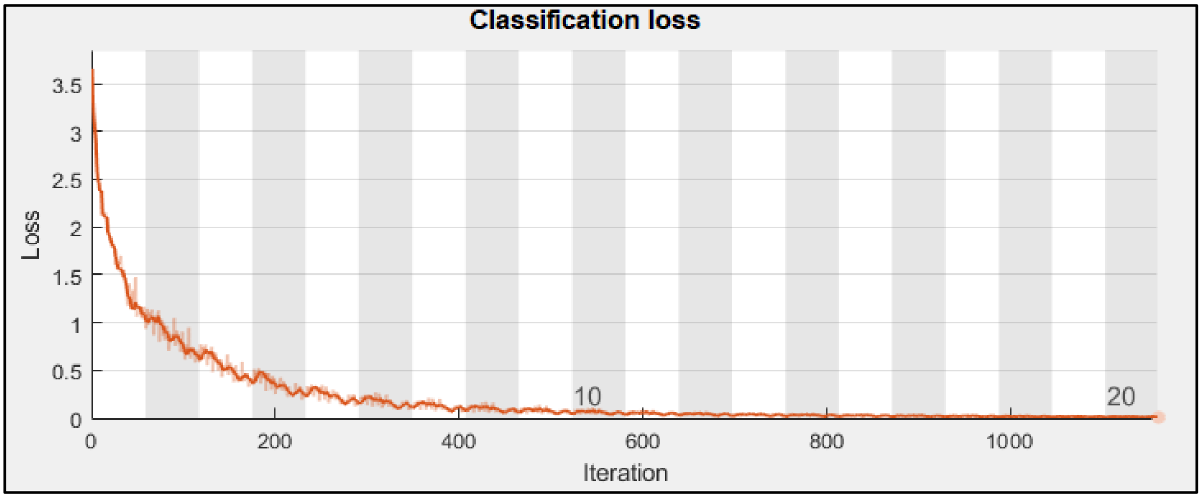


FIGURE 37. Performance loss during real-time execution of the model.

CHAPTER 5

CONCLUSION

Robotic systems have an exceptional flexibility in terms of mechanical strengths, capabilities, and performance. To model and train a robot to implement a desired activity efficiently, understanding and analyzing the activity as if we humans are to perform it is very important. The motivation for this proposed study is that robots should be able to perform a task from a human's perspective. Moreover, robots can be modeled and trained to make decisions autonomously, thereby decreasing the need for human intervention and, more importantly, reducing execution time.

In this research, we introduced a methodology to integrate two Deep Learning models for speech recognition and image classification. We used a convolutional neural network of DL to train and validate the network for both the modules. The speech recognition model classified the spoken word (colored-ball) and sent it to the image classification network. Image classification network was trained accurately to point out the colored ball based on the input from the preceding model. Subsequently, image processing was performed, post image classification, using MATLAB. Image processing executed the algorithm to compute the best shot for the robot. The robot was then programmed to strike the cue ball using its API. As a result of these powerful modules, the robotic arm was able to pocket the desired ball completely autonomously. Consequently, the developed model showed accurate and precise overall performance during testing.

Due to their large amount of computational power, robots can be made more powerful by training them using learning-based algorithms and other Machine Learning methods. Additionally, more human features can be incorporated into robots, so as to better mimic human activities. This knowledge can be applied in designing robotic systems with varied capabilities and challenging situations. In addition, voice-activated intelligent assistive robots can be developed by training them to recognize and interpreting the voice. Such voice-activated assistants can be found on nearly every smartphone these days (e.g. Apple introduced voice-activated assistant Siri, Google developed Google Now, the voice-activated assistant for Android).

The principles of this thesis could be used to create voice-activated robots with a much wider variety of applications than currently exist. Building such voice-enabled robots could greatly comfort disabled people in their daily activities. Furthermore, computer vision could also be installed into robotic systems to accomplish high-level operations such as mobile robot navigation.

We plan to extend this research work by training the network using large-scale data with large number of diverse human subjects for speech recognition and collecting more and more imagery data. Deploying image feedback, to know whether the desired ball is pocketed or not, would also be a part of future extension of this work. Image feedback methodology would reanalyze the image after task completion to ensure whether or not the task is performed accurately. By incorporating Machine Learning techniques such as DL, the robot can learn to visualize and interpret like humans. This would also enable the robotic machine to take significantly decisive steps in a dynamic environment. We plan to strengthen the extension of this study by inculcating machine intelligence into the application, which would train the robot to perform its programmed tasks in such real-time scenarios.

In future research, we intend to develop robots that can execute activities in real-time applications with lesser processing times, such as automatic target detection (ATD). This kind of project would provide promising results in highly challenging environments; this research study could be stretched to sectors where enhanced situation awareness is required in applications such as automatic military operations and surveillance missions. Thus, it can be seen that Deep Learning techniques will be the principal drive towards innovation in this era of machinery data.

REFERENCES

REFERENCES

[1] Columbus, L., “Internet of Things, Machine Learning & Robotics Are High Priorities for Developers in 2016,” Forbes, June 18, 2016, https://www.forbes.com/sites/louis columbus/2016/06/18/internet-of- things-machine-learning-robotics-are-high-priorities-for-developers-in-2016 /#65612a987270.

[2] Mohamed, A., Dahl G.E., and Hinton G., 2012, “Acoustic Modeling using Deep Belief Networks,” IEEE Transactions Audio, Speech. And Language Processing, 20 (1), pp. 14 –22.

[3] Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A., Jaitly, A., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B., 2012, “Deep Neural Networks for Acoustic Modeling in Speech Recognition,” IEEE Signal Processing Magazine, 29 (6), pp. 82 –97.

[4] "Official Rules of the Games of Snooker and English Billiards" (PDF), The World Professional Billiards & Snooker Association Limited, November 2014, https://www.wpbs a.com/wp-content/uploads/2016/03/official-rules-of-the-game.pdf.

[5] “MATLAB (R2018a), MathWorks” accessed March 14, 2018, https://www.mathworks.com /products/matlab.html.

[6] Christian, S., Toshev, A., and Erhan, D., “Deep Neural Networks for Object Detection,” 2013, Proceedings of NIPS, Lake Tahoe, NV, December 5- December 10, Paper No. 5207.

[7] Koehler, J., 1995, The Science of Pocket Billiards, Sportology Publications, Marinette, WI.

[8] Billiards Congress of America, 2014, Billiards: The Official Rules and Records Book, Billiards Congress of America, Broomfield, CO.

[9] Arthur, S. L., 1988, "Some Studies in Machine Learning Using the Game of Checkers. I". Computer Games I, D. N. L. Levy, ed., Springer, New York, NY, pp. 335–365.

[10] Kohavi, R., and Provost, F., 1998, “Glossary of Terms," Machine Learning, 30 (2-3), pp. 271-274.

[11] "Machine Learning: What It Is and Why It Matters," accessed April 19, 2018, https://www. sas.com/itit/insights/analytics/machine-learning.html.

[12] Kotsiantis, S., Zaharakis, I., and Pintelas, P., 2002, “Supervised Machine Learning,” TR-02-02, Department of Mathematics, University of Patras, Patras, Greece.

[13] Dietterich, T. G., 1998, “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms,” Neural Computation, 10(7), pp 1895–1924.

[14] Batista, G., and Monard, M.C., 2003, "An Analysis of Four Missing Data Treatment Methods for Supervised Learning," Applied Artificial Intelligence, 17, pp.519-533.

[15] Brighton, H., and Mellish, C., 2002, "Advances in Instance Selection for Instance-Based Learning Algorithms," Data Mining and Knowledge Discovery, 6, 153–172.

[16] Dy, J. G., and Brodley, C., 2004, "Feature Selection for Unsupervised Learning," Journal of Machine Learning Research, 5, 845–889.

[17] James, G., Witten, D., Hastie, T., and Tibshirani, R., 2013, An Introduction to Statistical Learning: with Applications in R, Springer, New York.

[18] Bousquet, O., Raetsch, G., and von Luxburg, U., 2004, "Advanced Lectures on Machine Learning," LNAI 3176, Springer-Verlag, Tübingen, Germany.

[19] Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y., 2012, "Building High-Level Features Using Large Scale Unsupervised Learning," Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, Canada, May 26- May 31, pp. 8595-8598.

[20] Barto, A. G., and Sutton, R., 1997, Introduction to Reinforcement Learning, MIT Press, Cambridge, MA.

[21] Tadepalli, P., and Ok, D., 1998, "Model-Based Average Reward Reinforcement Learning Algorithms," Artificial Intelligence, 100, pp. 177–224.

[22] Sutton, R. S. 1992, "A Special Issue on Machine Learning on Reinforcement Learning," Springer, Boston, MA.

[23] Schwartz, A., 1993, "A Reinforcement Learning Method for Maximizing Undiscounted Rewards," Proceedings of the 10th Annual Conference on Machine Learning, Amherst, MA, pp. 298–305.

[24] LeCun, Y., Bengio, Y., and Hinton, G., 2015, "Deep Learning," Nature, 521 (7553), pp. 436–444.

[25] Fausett, L., 1994, Fundamentals of Neural Networks, Prentice-Hall, Englewood Cliffs, NJ.

[26] Haykin, S., 1999, Neural Networks: A Comprehensive Foundation, Prentice Hall, Upper Saddle River, NJ.

[27] Lippmann, R. P., 1987, "An Introduction to Computing with Neural Network," IEEE Acoustic, Speech, and Signal Processing Magazine, 4, pp. 4-22.

[28] Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., and Schmidhuber, J., 2009," A Novel Connectionist System for Improved Unconstrained Handwriting Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, 31 (5), pp. 855-868.

[29] Medsker, L., and Jain, L., 1999, Recurrent Neural Networks: Design and Applications, CRC Press, Boca Raton, FL.

[30] Li, S., Li, W., Cook, C., Zhu, C., and Gao, Y., 2018, "Independently Recurrent Neural Network (INDRNN): Building A Longer and Deeper RNN," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, June 18-June 22, pp. 5457-5466.

[31] Graves, A., Mohamed, A.-R., and Hinton, G., 2013, "Speech Recognition with Deep Recurrent Neural Networks," Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, May 26-May 31, pp. 6645–6649.

[32] Fernández, S., Graves, A., and Schmidhuber, J., 2007, "Sequence Labelling in Structured Domains with Hierarchical Recurrent Neural Networks," Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI, Hyderabad, India, January 6- January12, pp. 774–779.

[33] Goodfellow, I., Jean Pouget-A., Mirza, M., Xu, B., Warde-Farley, D., Ozair, O., Courville A., and Benjio Y., 2014, "Generative Adversarial Nets," Advances in Neural Information Processing Systems, Montreal, Canada, December 8 – December 13, pp. 2672-2680.

[34] Luc, P., Couprie, C., Chintala, S., and Verbeek, J., 2016, "Semantic Segmentation Using Adversarial Networks,"NIPS Workshop on Adversarial Training, Barcelona, Spain, December 9.

[35] Gross, R., Gu, Y., Li, W., and Gauci, M., 2017, "Generalizing GANs: A Turing Perspective," Proceedings of the Thirty-first Annual Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, December 4- December 9, pp. 1–11.

[36] “Convolutional Neural Networks (LeNet) -Deep Learning 0.1 Documentation," Deep Learning 0.1. LISA Lab, accessed on June 01, 2018, http://deeplearning.net/tutorial/ lenet.html.

[37] Krizhevsky, A., Sutskever, I., and Hinton, G.E., 2012, “ImageNet Classification with Deep Convolutional Neural Networks,” Proceedings of the Advances in Neural Information Processing Systems (NIPS), Lake Tahoe, NV, USA, December 3- December 8, pp. 1097–1105.

[38] Nam, H., and Han, B., 2016, “Learning Multi-Domain Convolutional Neural Networks for Visual Tracking,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27-June 30, pp. 4293-4302.

[39] Wang, L., Ouyang, W., Wang, X., and Lu, H., 2016, “STCT: Sequentially Training Convolutional Networks for Visual Tracking,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27-June 30, pp. 1373-1381.

[40] Pratt, Y. L., Mostow, J., and Kam, C. A., 1991, “Direct Transfer of Learned Information Among Neural Networks,” Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91), Anaheim, CA, July 14-19, pp. 584-589.

[41] Gu, Q. Q., and Zhou, J., 2009, “Learning the Shared Subspace for Multi-Task Clustering and Transductive Transfer Classification,” Proceedings of the International Conference on Data Mining (ICDM), Miami, Florida, USA, December 6-9, pp. 159-168.

[42] Li, T., Sindhwani, V., Ding, C., and Zhang, Y., 2009, “Knowledge Transformation for Cross-Domain Sentiment Classification,” Proceedings of the 32nd SIGIR, Boston, Massachusetts, USA, July 19-23, pp. 716–717.

[43] Li, B., Yang Q., and Xue, X. Y., 2009, “Transfer Learning for Collaborative Filtering via a Rating-Matrix Generative Model,” Proceedings of the 26th Annual International Conference on Machine Learning (ICML), Montreal, Quebec, Canada, June 14-18, pp. 617–624.

[44] Zhou, H., Yang, Q., Hu, D. H., and Li, L., 2008, “Transferring Knowledge from Another Domain for Learning Action Models,” Proceedings of the Pacific Rim International Conferences on Artificial Intelligence (PAICAI), LNCS, Springer, Heidelberg, December 15-19, pp. 1110–1115.

[45] Pratt, L. Y., 1993, "Discriminability-Based Transfer between Neural Networks", NIPS Conference: Advances in Neural Information Processing Systems, 5, November 29-December 02, pp. 204–211.

[46] Dietterich, T., Pratt, L., and Sebastian T., (Eds.), 1997, “Machine Learning: Special Issue on Inductive Transfer,” Kluwer Academic Publishers, Hingham, MA, USA, 28(1).

[47] Ballard, D., and Brown, C., 1982, Computer Vision, Prentice Hall, Inc. Englewood Cliffs, NJ.

[48] Reinhard, K., 2014, Concise Computer Vision, Springer, London.

[49] Morris, T., 2004, Computer Vision and Image Processing, Palgrave Macmillan, London.

[50] “CNN for Visual Recognition,” accessed April 24, 2018, http://cs231n.github.io/ classification.

[51] Victor, W., and Thomas, L., 2018, “Computer Vision and Image Processing: A Paper Review,” International Journal of Artificial Intelligence Research, 2(1), pp. 22-31.

[52] Altman, N. S., 1992, “An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression,” The American Statistician, 46 (3), pp. 175-185.

[53] “A Quick Introduction to K-Nearest Neighbors Algorithm,” accessed May 04, 2018, https:// medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7.

[54] Samworth, R. J., 2012, "Optimal Weighted Nearest Neighbor Classifiers," The Annals of Statistics, 40(5), 2733-2763.

[55] Baker, J. M., 2009, "Developments and Directions in Speech Recognition and Understanding, Part 1 (DSP Education)," IEEE Signal Processing Magazine, 26 (3), pp. 75-80.

[56] Beigi, H., 2011, Fundamentals of Speaker Recognition, Springer, New York.

[57] Gevaert, W., Tsenov, G., and Mladenov, V., 2010, “Neural Networks used for Speech Recognition,” Journal of Automatic Control, 20(1), pp. 1-7.

[58] Ghahramani, Z., 2001, “An Introduction to Hidden Markov Models and Bayesian Networks,” International Journal of Pattern Recognition and Artificial Intelligence, 15, 9-42

[59] Zegers, P., 1998, “Speech Recognition Using Neural Networks,” Master’s Thesis, University of Arizona, AZ, USA.

[60] Connolly, J. H., Edmonds, E. A., Guzy, J. J., Johnson, S. R., and Woodcock, A., 1986, “Automatic Speech Recognition based on Spectrogram Reading,” International Journal of Man-Machine Studies, 24 (6), pp. 611-621.

[61] Das, S., Bakis, R., Nadas, A., Nahamoo, D., and M. Picheny, M., 1993, "Influence of Background Noise and Microphone on the Performance of the IBM Tangora Speech Recognition System," IEEE International Conference on Acoustics, Speech, and Signal Processing, Minneapolis, MN, USA, April 27-30, pp. 71-74.

[62] Tebelskis, J., 1995, “Speech Recognition Using Neural Networks,” PhD Dissertation, Carnegie Mellon University, Pittsburgh, PA, USA.

[63] Furui, S., 1989, “Digital Speech Processing, Synthesis and Recognition,” Marcel Dekker Inc., New York.

[64] Jebara, T., Eyster, C., Weaver, J., Starner, T., and Pentland, A., 1997, “Stochasticks: Augmenting the Billiards Experience with Probabilistic Vision and Wearable Computer,” Proceedings of the IEEE International Symposium on Wearable computers, Cambridge, MA, USA, October 13-14, pp. 138–145.

[65] Larsen, L. B., Jensen, M. D., and Vodzi, W. K., 2002, “Multi Modal User Interaction in an Automatic Pool Trainer,” Proceedings of the Fourth IEEE International Conference on Multimodal interfaces (ICMI’02), Pittsburgh, USA, October 14-16, pp. 361–366.

[66] Smith, M., 2007, “PickPocket: A Computer Billiards Shark,” Artificial Intelligence, 171(16–17), pp. 1069–1091.

[67] Leckie, W., and Greenspan, M., 2006, “An Event-Based Pool Physics Simulator,” Lecture Notes Computer Science, Springer, Berlin, Heidelberg, 4250, 247–262.

[68] Nadler, D., 2005, “Mathematical Theory of Spin, Friction, and Collision in the Game of Billiards: An English Translation of Coriolis’ 1835 Book,” http://www.coriolis billiards.com/

[69] “Physics of Billiards - Real World Physics Problems,” accessed on March 20, 2018, http:// www.real-world-physics-problems.com/physics-of-billiards.html.

[70] “The Math and Physics of Billiards,” accessed on March 23, 2018, http://archive.ncsa. illinois.edu/Classes/MATH198/townsend/math.html.

[71] Chun, H.-T., 2008, “A Mathematical Analysis of Billiards Games: Amplification Factor, Stability of Ball Path, and Player’s Pattern,” http://citeseerx.ist.psu.edu/viewdoc /download?doi=10.1.1.575.4989&rep=rep1&type=pdf .

[72] Laumond, J.-P., 1998, Robot Motion Planning and Control, Springer-Verlag, Berlin.

[73] “Kinova JACO Robotic Arm,” accessed March 16, 2018, https://www.kinovarobotics .com/en/products/robotic-arm-series/jaco-prosthetic-robotic-arm/.

[74] Warden, P., 2017, "Speech Commands: A Public Dataset for Single-Word Speech Recognition," retrieved on April 01, 2018, http://download.tensorflow.org/data/speech _commands_v0.01. tar.gz, Copyright Google.

[75] Mannell R.H., 2002, "The Perception of Speech Processed with Non-Overlapping and Overlapping Filters in a Bark-Scaled Channel Vocoder," Proceedings of the Ninth Australian International Conference on Speech Science and Technology, Melbourne, Australia, December 3-5.

[76] Wilamowski, B. M., 2009, "Neural Network Architectures and Learning Algorithms," IEEE Industrial Electronics Magazine, 3 (4), pp. 56-63, doi: 10.1109/MIE. 2009.934790.

[77] Abdel-Hamid, O., Deng, L., and Yu, D., 2013, “Exploring Convolutional Neural Network Structures and Optimization Techniques for Speech Recognition,” Proceedings of 14th Annual Conference of the International Speech Communication Association, INTERSPEECH 2013, Lyon, France, August 25-29, pp. 3366–3370.

[78] Visa, S., Ramsay, B., Ralescu, A., and Knaap, E., 2011, “Confusion Matrix-Based Feature Selection,” Proceedings of The 22nd Midwest Artificial Intelligence and Cognitive Science Conference, CEUR Workshop Proceedings, 710, Cincinnati, Ohio, USA, April 16-17, pp. 120-127.

[79] Chaudhari, D. K., 2017, “Smart Robotics Prosthesis Using Deep Learning and Musculoskeletal Modeling,” Dissertations & Theses at California State University, Long Beach 2009347715, retrieved April 4, 2018, http://csulb.idm.oclc.org/login?url=https://search-proquest-com.csulb.idm.oclc.org/docview/2009347715?accountid=10351.

[80] Abadi, M., 2016, “TensorFlow: A System for Large-Scale Machine Learning,” Proceedings of 12th USENIX Symposium on Operating System Design and Implementation, Google Brain, Savannah, GA, USA, November 2–4, pp. 265-283.

[81] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Feifei, L., 2009, “ImageNet: A large-scale hierarchical image database,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, FL, USA, June 20-25, accessed April 24, 2018, http://image-net.org/.