IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Air-Writing Alphanumeric Character Recognition

Ms. M. Saranya
Computer Science
College of Engineering Guidny
Chennai, India
saranyamani18@gmail.com

Kariketi Tharun Reddy
Computer Science
College Of Engineering Guindy
Chennai, India
tarun7800@gmail.com

Madhumitha Raju
Computer Science
College Of Engineering Guindy
Chennai, India
madhumitharaju2001@gmail.com

Manoj Kutala
Computer Science
College Of Engineering Guindy
Chennai, India
kuthalamanoj@gmail.com

Abstract— Our project develops a system that can recognize the air-written words in 3D space, and then classify the recognized character into one of the possible classes. Air-writing is the new way of writing the linguistic characters or words in free area using hand or finger movements. Writing within the air may be a method to jot down one thing in an exceedingly 3D house with our finger-tip. This project could be a combination of computer vision object chase and handwriting recognition machine learning. The air writing recognition system uses the digital camera of a pc to trace character digits written within the air by the user and then uses a convolutional neural network to classify the character and digits into one of the possible classes. Several current systems use advanced and high-priced chase setups to realize gesture recognition, however, we tend to get to form a system that may attain a similar work with a far cheaper setup.

KEYWORDS: Hand writing recognition, hand writing classification, computer vision, machine learning, convolutional neural networks.

I. INTRODUCTION

Handwriting analysis could be a classic, well-explored downside within the realm of introductory machine learning that encompasses several of the necessary topics of neural networks. Air-writing is particularly helpful for user interfaces that don't permit the user to kind on a keyboard or pen a trackpad/touchscreen, or for text input for sensible system management, among several applications. Air-writing differs from typical hand writing; the one that performs air writing will solely use associate degree unreal coordinate to guide

the writing motion. The variability of motion knowledge represents a letter is so significantly broader in air-writing than in paper writing. Our project aims to use a mixture of computer vision and handwriting recognition to form a system that acts as a virtual whiteboard. Model users would be ready to write "air words" facing a web camera either real-time or in advance and have those gestures translated into letters or digits. The aim of this project is to further explore the task of classifying handwritten text and to convert handwritten text into a digital format and achieving a virtual whiteboard system at a cost that is accessible to the average user. We want to introduce alternative interfaces for communication that have high affordability, usability, and accessibility.

II. PROBLEM STATEMENT

Generally, in the field of 3D vision systems, the advanced techniques such as Leap motion and Kinect Cameras are used for writing the text in air. Tracking systems such as HTC Vive virtual reality (VR) had incorporated this mechanism but the system comes at a high cost that also requires an addition extensive (expensive) tracking system. Air-Writing allows for an entirely new text input interface that does not require much more than the computer itself. Our proposed method is done very easily by everyone and can be incorporated into their software. Here air-writing characters are recognised by using a library called OpenCV, that lets us detect the object we want to write with and then track its path. Any object like pen, pencil or a fingertip can be tracked by our system. We display to the user the character he/she has written in air by tracing the path of the object that is used to write in air. We then pass the image to the trained convolutional neural network for prediction of characters and digits.

III. LITERATURE SURVEY

An early notable attempt in the area of character recognition research is by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as the analysis-by-synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition.

K. Gaurav, Bhatia P. K. Et al,[6] this paper deals with the various pre-processing techniques involved in character recognition with different kinds of images ranges from simple handwritten form-based documents and documents containing coloured and complex backgrounds and varying intensities. In this, different pre-processing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed. It was concluded that using a single technique for pre-processing, we can't completely process the image. However, even after applying all the said techniques might not possible to achieve full accuracy in a pre-processing system.

In [7], Renata F. P. Neves have proposed SVM based offline handwritten digit recognition. Authors claim that SVM outperforms the Multilayer perceptron classifier. Experiment is carried out on NIST SD19 standard dataset. Advantage of MLP is that it is able to segment non-linearly separable classes. However, MLP can easily fall into a region of local minimum, where the training will stop assuming it has achieved an optimal point in the error surface. Another hindrance is defining the best network architecture to solve the problem, considering the number of layers and the number of perceptron in each hidden layer. Because of these disadvantages, a digit recognizer using the MLP structure may not produce the desired low error rate.

In [12] the system proposed used the depth and colour information from Kinect sensor to detect the hand shape. As for the gesture recognition, even with the Kinect sensor. It is still a very challenging problem. The resolution of this Kinect sensor is only 640×480. It works well to track a large object, e.g. the human body. But tracking a very small object like finger is complex.

IV. PROPOSED SOLUTION

The proposed system uses the camera for Hand gesture recognition. In our prototype, two basic techniques are used that includes processing of a video input frame and then pattern recognition of processed frame using CNN.

Figure 4 depicts the architecture design of our system which has four modules that helps us to detect alphabets and numbers drawn by motion of any object in air. The basic working is that a video input stream from camera will be captured and pre-processed using background subtraction, Gaussian blurring and thresholding algorithms to separate foreground objects of user from background and correctly detect it in any background condition. Object motion of user is then traced and the frame with tracked path is pre-processed and then given to the trained convolutional neural network which consists of large enough training dataset and our trained CNN model performs recognition of written text.

V. DATASETS

This project builds two models one for digits and the other for letters as the use of one model for recognising both letters and digits leads to mis-predictions and flaws. Our system uses [13] EMNIST dataset for handwritten classification. The two datasets have 784 columns in pixel format including the labels and around 4 lakh samples in total. We split the dataset into two for training and testing by 8:2 ratio for letters dataset and 9:1 ratio for digits dataset respectively. The input to the convolutional neural network is a 2D array of 28x28 pixels.

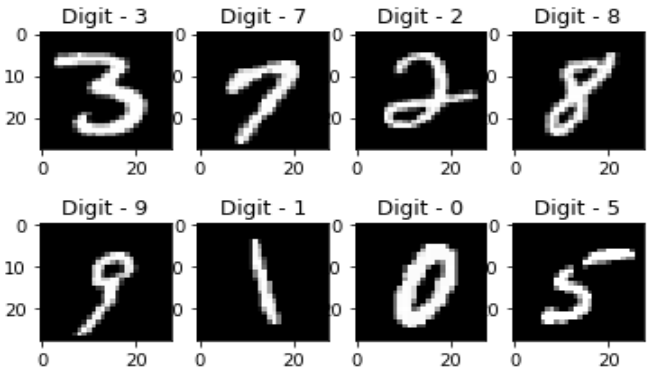

Figure 1. Dataset for digits.

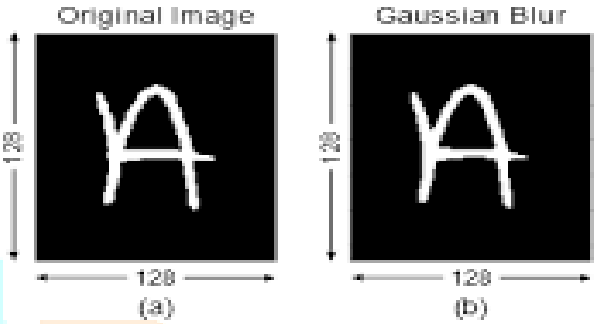

Figure 2. Dataset for Letters

VI. SYSTEM DESIGN

A. Air-Writing

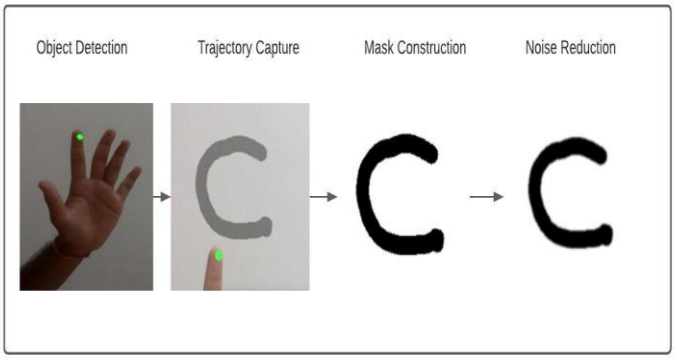

Figure 3 The Air-writing process for object detection and data acquisition along with noise reduction. Here we have used fingertip as our object.

NOTE: Any object can be used to write in air.

B. Overall Architecture Diagram

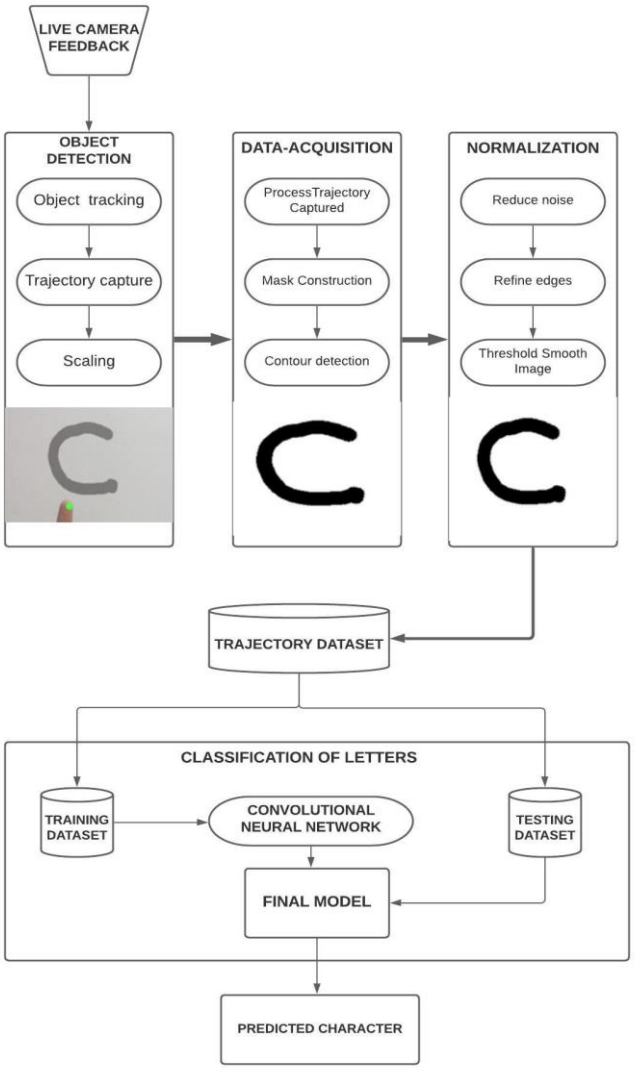

Figure. 4. Overall block diagram of the system along with Air-writing recognition and handwritten character classification.

VII. MODULES

A. Object Detection

In our model, object tracking is realized with the detection of the target object through image analysis of each video frame OpenCV -an open-source computer vision and machine learning library provides enough functions and facilities so that we can process our image with various computer vision algorithms hence the OpenCV library is used to detect, track, and save the trajectory of the target object as its position shifts throughout the video. The digital representation of each frame is pre-processed. The digital image is nothing more than a matrix of scalar or vector values (depending on whether the image is monochromatic or coloured) and it can be processed in numerous ways. That is where the OpenCV library may be used since it is optimized for such operations. We use optical flow function. Motion of image objects between two consecutive frames which is caused by the motion of cameras that creates a pattern and it is called as Optical flow [14]. This is 2D vector field where each vector is a displacement vector showing the movement of points from first frame to second.

Optical Flow -Lucas Kanade Method

Assumptions:

The pixel intensities of an object do not change between consecutive frames.

Neighbouring pixels have similar motion.

Consider a pixel **I(x,y,t)** in first frame. It moves by distance **(dx,dy)** in next frame taken after **dt** time. So since those pixels are the same and intensity does not change, we can say,

$$I(x,y,t)=I(x+dx,y+dy,t+dt)$$

Then take taylor series approximation of right-hand side, remove common terms and divide by dt to get the following equation:

$$fxu+fyv+ft=0$$

where: $x=\partial f\partial x;\ fy=\partial f\partial y$

$$u=dxdt;\ v=dydt$$

Above equation is called Optical Flow equation, from which we can find fx and fy, they are image gradients. Similarly, ft is the gradient along time. But (u,v) is unknown. We cannot solve this one equation with two unknown variables. So several methods are provided to solve this problem and one of them is Lucas-Kanade. Lucas-Kanade method [14] takes a 3x3 patch around the point, hence, all the 9 points have the same motion. We can find (fx,fy,ft) for these 9 points. So now our problem becomes solving 9 equations with two unknown variables which are over-determined. A better solution is obtained with least square fit method. Below is the final solution which is two equation-two unknown problem and solve to get the solution.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i f_{x_i}{}^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & \sum_i f_{y_i}{}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix}$$

Figure.5 Lucas-Kanade Algorithm

It denotes that corners are better points to be tracked. So from the user point of view, the idea is simple, we give some points to track; we receive the optical flow vectors of those points.

B. Data Acquisition

The creation of a mask or subtraction of background depends on several circumstances. Whether there is a separate image of background available, whether there is a lot of noise in the image, variability in lights, and others. Not all objects have strong features which can be immediately recognized. There was the conversion of colour space performed before actual sample selection. The reason for conversion is that RGB colour space (default colour space for OpenCV) defines colour using 3 values, where the HSV (or HSB) colour space defines colour with only one value.

We pre-process each frame by resizing and reducing Gaussian noise. Then, we construct a binary mask around the object and perform morphological transformations to clean up. We find the contours using an OpenCV algorithm which calculates the hierarchy of contours in the image and compresses it.

C. Normalization

Image normalization [15] is a typical process in image processing that changes the range of pixel intensity values. Its normal purpose is to convert an input image into a range of pixel values that are more familiar or normal to the senses, we normalize strokes and remove variations that would otherwise complicate recognition and reduce the recognition rate. The video frames captured from the camera may have noise especially when the ambient light around the sensor is low. The Gaussian blurring process reduces the sensor noise and reduces the effects of low light condition on input frame to some extent, resulting in less number of false contours and regions in our subsequent operation. Blurring of image is done by convolving the image [16] with a low-pass filter kernel. The kernel here is a simple Gaussian kernel matrix. Each pixel in the input frame matrix gets multiplied by the Gaussian kernel. The output values of these multiplications are summed up and that result is used to set value at the destination pixel.

D. Classification of Letters

After obtaining the pre-processed, 28x28 sized image of the written character, our next step is to run it through a trained convolutional. While training with pre-processed images, the input to CNN is a fixed-size 28x28 grayscale image. T The batch size was set to 256 and the number of epoch was 25. The image is passed through a stack of convolutional layers, each of which has 3x3 filters.

When an input image is presented to the system, its features are extracted and given as input to the trained classifier like an artificial neural network or support vector machine. Classifiers compare the input feature with the stored pattern and find out the best matching class for input.

Two activation functions are used for training the CNN are ReLU and Softmax. Relu (Rectified Linear Unit) Activation function has output 0 if the input is less than 0, and raw output otherwise f(x) = max(x,0) It results in much faster training for large network. The Softmax Activation function's input is a set of values and it operates on it to make their sum equal to one, which will give probabilities for each value, i.e 0<softmax<1.

TABLE 1: CNN MODEL DESCRIPTION

LAYERS	PARAMETERS
Input layer	(28,28,1)
2D convolutional layer	Activation=Relu filters=32 kernal_size=(3,3)
Max pooling layer	pool_size=(2,2) strides=2
2D convolutional layer	Activation='relu', filters=64

	kernel_size=(3, 3) padding = 'same'
Max pooling layer	pool_size=(2,2) strides=2
2D convolutional layer	activation='relu' filters=128 kernel_size=(3, 3) padding = 'valid'
Max pooling layer	pool_size=(2,2) strides=2
Flatten	
Dense	nodes=64 activation ="relu"
Dense	nodes=128 activation ="relu"
Dense	node =64 activation=” softmax”

TABLE 2: MODULE WISE INPUT AND OUTPUT

Module Name	**Input**	**Output**
Object detection	Live video feed	Trajectory Path
Data-Acquisition	Trajectory Path	Masked Image
Normalization	Masked Image	Smooth Image
Classification Of Characters	Pre-processed Image	Predicted Character

VIII. RESULTS AND ANALYSIS

When we initially trained our neural network on 414,450 samples (42,000 samples digit dataset and 372,450 alphabet dataset) of EMNIST data in which we split 8:2 between testing and evaluation for alphabets datasets and 9:1 split for digits dataset. We had trained the both datasets for 10 epochs for digits the accuracy is 98.71% and validation accuracy is 99% (Figure 6). We were also at a training loss percentage of 4.62% and validation loss percentage of 4.24% (Figure 7). Here we can see that the training accuracy and validation accuracy are almost same. The training data, and when tested on the original 10% test data, we were able to achieve 99% test accuracy and 4.24% test loss.

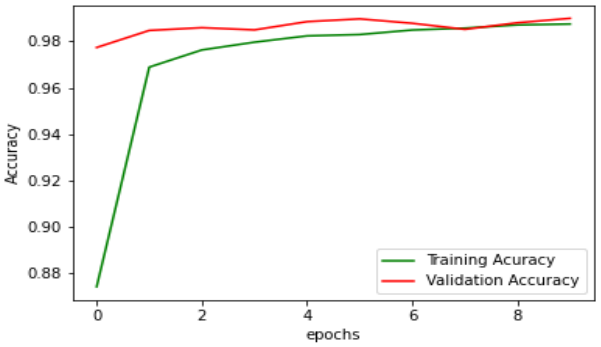

Figure 6. Digit Model Accuracy

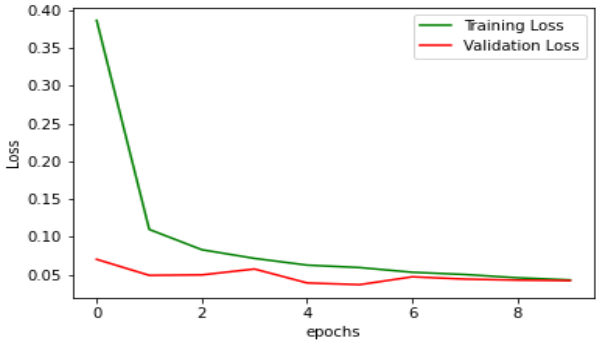

Figure 7. Digit Model Loss

Now for Alphabets the accuracy is 98.83% and validation accuracy is 98.76% (Figure 8). We were also at a training loss percentage of 4.78% and validation loss percentage of 5.83% (Figure 9). Here also we can see that the training accuracy and validation accuracy are almost same. The training data, and when tested on the original 10% test data, we were able to achieve 98.77% test accuracy and 5.84% test loss.

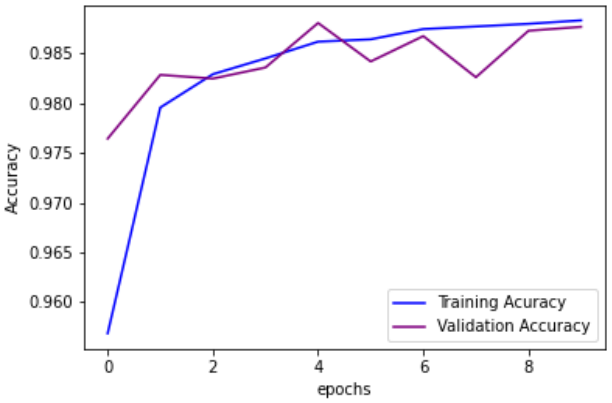

Figure 8. Alphabet Model Accuracy

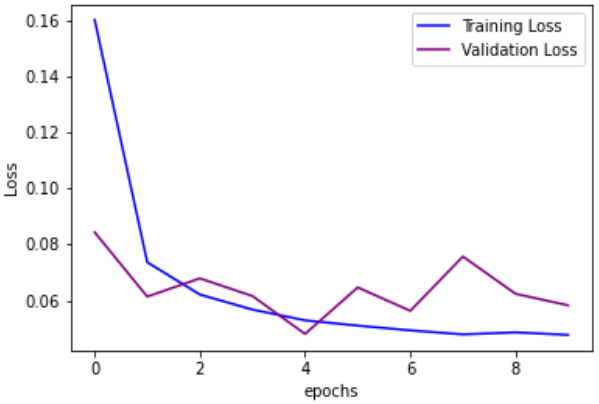

Figure 9. Alphabet Model Accuracy

Here why we had used two datasets is for better prediction accuracy and also to avoid misconceptions between digits and characters like O and 0 which are same by this way we can differ them both (Figure 10).

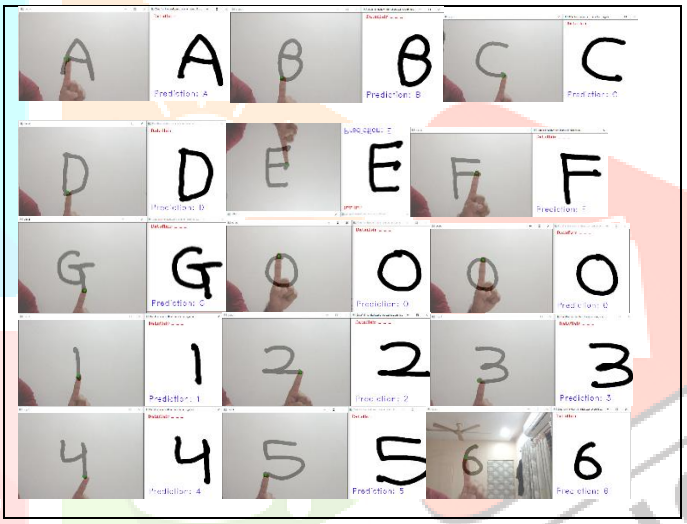

Figure 10. Air Written Characters/digits

IX. CONCLUSION

Our work took a classic machine learning introduction project, handwriting analysis, and extended it to new interfaces so that it may be used in different ways. We challenged ourselves by implementing a novel solution to the misconception problem because we believed that our solution provides a more intuitive interface for end users. Although the accuracy is not 100% (few things are), we hope our small contribution draws attention toward the nuances of different implementations of the same machine learning, computer vision problems and encourages other researches to think more deeply about the way users use the technologies they create.

X. FUTURE WORK

In future, we would be interested in increasing both the usability. One, we would continue working to give the input as continues and predict each and every word. Currently, the program can predict alphabets and digits in a single input format only also depth checking to determine whether the gesture would be tracked or untracked, which often leads to rough edges and small skips in a single stroke. Although we already use

blurring techniques to reduce noise, we would like to implement a dynamic, weighted line that increases or decreases based on the depth of the tracked object. This may help with both usability and accuracy as users can more easily recognize the depth of which they are writing, and the output images will not have the harsh skips that happen when the object exits the tracked range. We would also like to conduct more testing with different variations of our CNN. In the future, we may implement more hidden layers and convolutional layers to try to capture more of the unique features of the 35 classes.

Beyond bugs, we would want to extend this project to recognize cursive and air-written words. If the accuracy of the current program were to increase, it would be interesting to continue extending the air-writing recognition to more complex gestures.

XI. ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our project guide, Mrs. Lalitha Devi K, Teaching Fellow, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for her constant source of inspiration. We thank her for the continuous support and guidance which was instrumental in taking the project to successful completion. We are grateful to Dr.S.Valli , Professor and Head, Department of Computer Science and Engineering, College of Engineering Guindy, Chennai for her support and for providing necessary facilities to carry out for our the project. We would also like to thank our friends and family for their encouragement and continued support. We would also like to thank the Almighty for giving us the moral strength to accomplish our task.

REFERENCES

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M. and Ghemawat, S., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems.

[2] Bradski, G., 2000. The openCV library. Dr. Dobb's Journal: Software Tools for the Professional Programmer, 25(11), pp.120-123.

[3] Ganger, M., Duryea, E. and Hu, W., 2016. Double Sarsa and double expected Sarsa with shallow and deep learning. Journal of Data Analysis and Information Processing, 4(4), pp.159-176.

[4] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: an extension of MNIST to handwritten letters. arXiv preprint arXiv:1702.05373 (2017).

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and [6] E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12 (2011), 2825–2830.

[7] K. Gaurav and Bhatia P. K., “Analytical Review of Preprocessing Techniques for Offline Handwritten Character Recognition”, 2nd International Conference on Emerging Trends in Engineering & Management,

ICETEM, 2013.

[8] Renata F. P. Neves, Alberto N. G. Lopes Filho, Carlos A.B.Mello, CleberZanchettin, "A SVM Based Off-Line Handwritten Digit Recognizer", International conference on Systems, Man and Cybernetics, IEEE Xplore, pp. 510-515, 9-12 Oct, 2011, Brazil.

[9] Air-writing Recognition, Part 1: Modeling and Recognition of Characters, Words and Connecting Motions Mingyu Chen, Ghassan AlRegib, Senior Member, IEEE, and Biing-Hwang Juang, Fellow, IEEE.

[10] C.-M. Karat, C. Halverson, D. Horn, and J. Karat, "Patterns of entry and correction in large vocabulary continuous speech recognition systems," in Proc. of the SIGCHI Conference on Human Factors in Computing Systems, 1999, pp. 568–575.

[11] Yuan-Hsiang Chang, Chen-Ming Chang, "Automatic Hand-Pose Trajectory Tracking System Using Video Sequences", INTECH, pp. 132- 152, Croatia, 2010.

[12] Erik B. Sudderth, Michael I. Mandel, William T. Freeman, Alan S. Willsky, "Visual Hand Tracking Using Nonparametric Belief Propagation", MIT Laboratory For Information & Decision Systems Technical Report P-2603, Presented at IEEE CVPR Workshop On Generative Model-Based Vision, Pp. 1-9, 2004.

[13] EshedOhn-Bar,Mohan ManubhaiTrivedi, "Hand Gesture Recognition In Real Time For Automotive Interfaces", IEEE Transactions on Intelligent Transportation Systems, VOL. 15, NO. 6, December 2014,pp 2368-2377

[14] Cohen, G., Afshar, S., Tapson, J. and Van Schaik, A., 2017, May. EMNIST: Extending MNIST to handwritten letters. In 2017 international joint conference on neural networks (IJCNN) (pp. 2921-2926). IEEE.

[15] Siong, L.Y., Mokri, S.S., Hussain, A., Ibrahim, N. and Mustafa, M.M., 2009, August. Motion detection using Lucas Kanade algorithm and application enhancement. In 2009 International Conference on Electrical Engineering and Informatics (Vol. 2, pp. 537-542). IEEE.

[16] Abu-Mostafa, Y.S. and Psaltis, D., 1985. Image normalization by complex moments. IEEE Transactions on Pattern Analysis and Machine Intelligence, (1), pp.46-55.

[17] Zhong, L., Cho, S., Metaxas, D., Paris, S. and Wang, J., 2013. Handling noise in single image deblurring using directional filters. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 612-619).