IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

A Mid-Air Word gesture and Voice Alert System for physically challenged using Machine Learning

[1]Anand M, [2]Namana Prasanna, [3]Sona Ganesh G [4]Sushmashree B S

[1]Assistant Professor [2,3,4]Student

Department of Information Science and Engineering

[1,2,3,4]GSSS Institute of Engineering and Technology for Women, Karnataka, India.

Abstract: Automatic detection of anomalies in space-and time-varying measurements is an important tool in several fields, e.g., fraud detection, climate analysis, crime detection or healthcare monitoring. We present an algorithm for detecting anomalous regions in multivariate spatio-temporal time-series, which allows for spotting the interesting parts in large amounts of data, in video frame data. In opposition to existing techniques for detecting isolated anomalous data points, we propose the "Maximally Divergent Intervals" (MDI) framework for unsupervised detection of coherent spatial regions and time intervals characterized r divergence compared with all other data given. In this regard, we define divergence that allows for ranking regions of different size and show how to enable the algorithm to run on large-scale data sets in reasonable time using an interval proposal technique. Experiments on both synthetic and real data from various domains, such as climate analysis, video surveillance, and text forensics, demonstrate that our method is widely applicable and a valuable tool for finding interesting events in different types of data.

I. INTRODUCTION

Motion gestures provide a complimentary modality for general human–computer interaction. Motion gestures are meant to be simple so that a user can easily memorize and perform them. However, motion gestures themselves are not expressive enough to input text for motion-based control. We define "air-writing" as writing letters or words in a free space. Air-writing is especially useful for user interfaces that do not allow the user to type on a keyboard. To develop a Machine Learning based system that uses gestures to perform functions. There are a lot of gesture-based applications existing in today's world but every time we want to use a gesture-controlled application, we need to learn the predefined gestures and the functionality is also limited to the defaults provided with it. For this project, we aim to build a system which can be trained to recognize the gestures we make and perform the dedicated function we decide for it. We will be demonstrating this by training the system to recognize letters by the gestures we make in air. To build this system, we will be using an Arduino board interfaced with an accelerometer. The device can be attached to the user's hand. The accelerometer will provide input to the microcontroller about the hand's coordinates. The algorithm will pick up this data and maintain a database to recognize each gesture differently. Once we train the system with the same gesture multiple times it will gather enough data to have an estimate of what the gesture should look like. This gesture can then be assigned to perform a task on the computer. Today Machine Learning is taking over everything. This is due to the fact that the machine can be made to learn some amazing things. Classification plays a vital role in many information administration and retrieval tasks. Document classification, also known as document categorization, is the process of assigning a class to one or more predefined category labels. Classification is often posed as a supervised learning problem in which a set of labeled data is used to train a classifier which can be applied to label future examples. Gesture classification includes different parts such as data processing, feature extraction, feature vector construction and final classification. Thus, improvement in each part should lead to better results in document classification. we apply machine learning methods for gesture recognition using classification techniques and algorithms available.

II. LITRTURE REVIEW

Before you begin to format your paper, first write and save the content as a separate text file. Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper Do not number text heads—the template will do that for you. Finally, complete content and organizational editing before formatting. Please take note of the following items when proof reading spelling and grammar returns of the shares and estimated betas.[1] "Hand gesture keyboard using machine learning" Number 2020 By Sam sunny, Sindhu v the development of hand gesture recognition using three different methods based universal keyboard that can be used under any circumstances. Gesture could be used as a tool which determines the user intent and

communicate between human and computer. In past few years' different methods has being evolved which can be seen in the field gesture recognition-based technology. [2] "Investigating Multi-Finger Gestures for Mid-Air Text Entry" September By Anna Maria the Sridhar, Christian Theobald, Antti Olivera the proposed prototype system for tracking the hand with the Leap Motion sensor1 and recognizing hand gestures as defined by Fast Type. It uses a computational optimization method to identify gesture sets that allow for high-throughput. Building on prior work in keyboard optimization. [3] "Arduino Based Motion Tracking Keyboard Using Machine Learning" 2019 By Aditya Priyadarshi, Bharath V, Aptha KS, Sanjit S, Sahana V This device is a gesture tracking device which uses machine learning's support vector machine model to convert accelerometer data to a sequence of alphabets. The main aim of the project is to build a device using an Arduino pro micro that translates gestures into words wirelessly [4] "Implementation of Gesture Keyboard Using Machine Learning" March 2019 By Bharathi M, Bhavani R, Keerthana Priya B, R.Paavaikarasi M.E This project propose a handy keyboard that is used to type in character by recognizing hand gesture keyboard are currently the most universally accepted computer input device .They may be wired, wireless ,or virtual , but the chance are that you are within a few centimetre of a keyboard right now.

III. METHODOLOGY

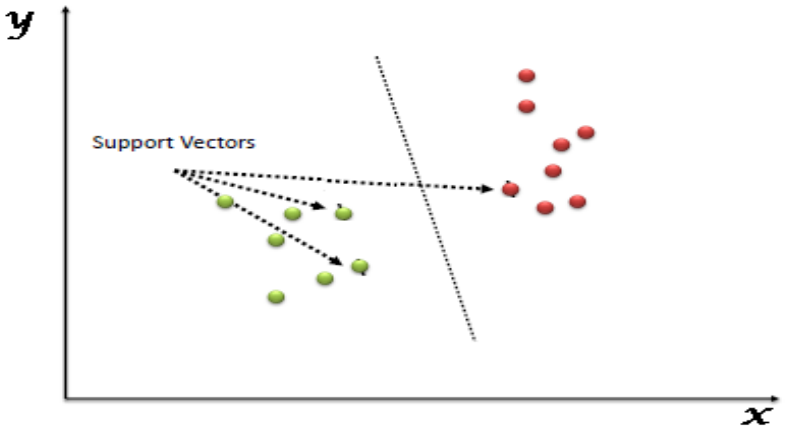
SUPPORT VECTOR MACHINES (SVMS)

Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset.

SVMs revolve around the notion of a "margin"—either side of a hyperplane that separates two data classes. Maximizing the margin and thereby creating the largest possible distance between the separating hyper plane and the instances on either side of it has been proven to reduce an upper bound on the expected generalization error. Support Vector Machine (SVM) is a popular classification technique discovered in 1995. A classification task usually involves separating data into training and testing sets. Each instance in the training set contains one "target value" and several "attributes". The final aim of the support vector machines is to make a model on the basis of training data which predicts the target values of test data by only giving out the attributes. Given a training set of instance-label pairs (xi, yi), i = 1,…,l, this requires the solution of the optimization problem. This algorithm needs that each data instance is represented as a vector of the real numbers. In case there are categorical attributes, it must be first converted into numeric data. Performing scaling before Support Vector Machine Algorithms is a very good practice. The very main advantage of scaling is to avoid all attributes in greater numeric ranges as to dominating those of them present in the smaller numeric ranges.

WORKING

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).



Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

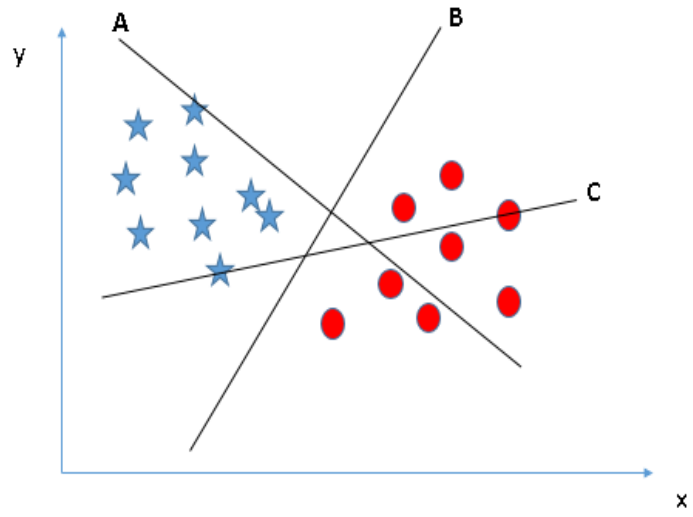
You can look at support vector machines and a few examples of its working here.

How does it work?

Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is "How can we identify the right hyper-plane?". Don't worry, it's not as hard as you think!

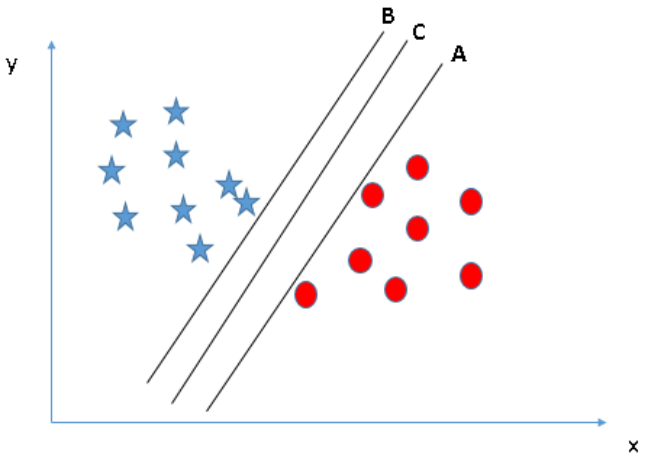
Let's understand:

Identify the right hyper-plane (Scenario-1): Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle

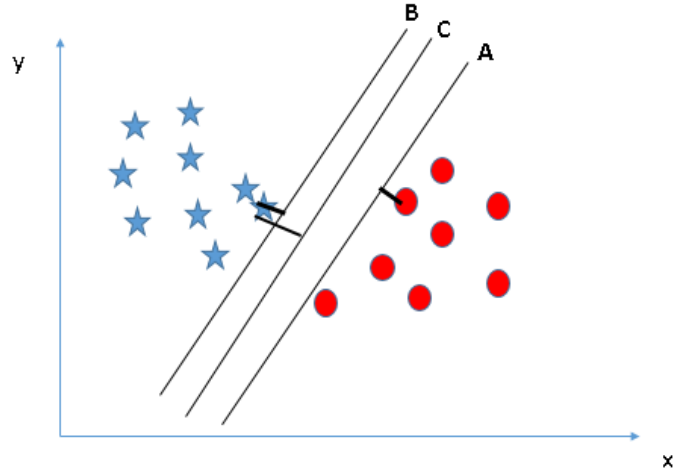


You need to remember a thumb rule to identify the right hyper-plane: "Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job.

Identify the right hyper-plane (Scenario-2): Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

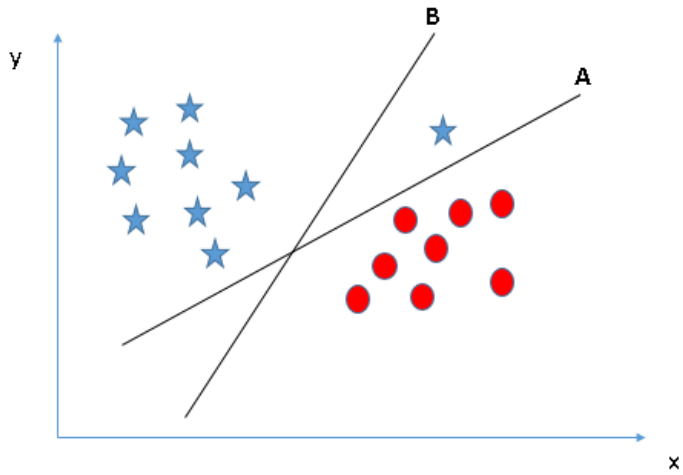


Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot:



Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

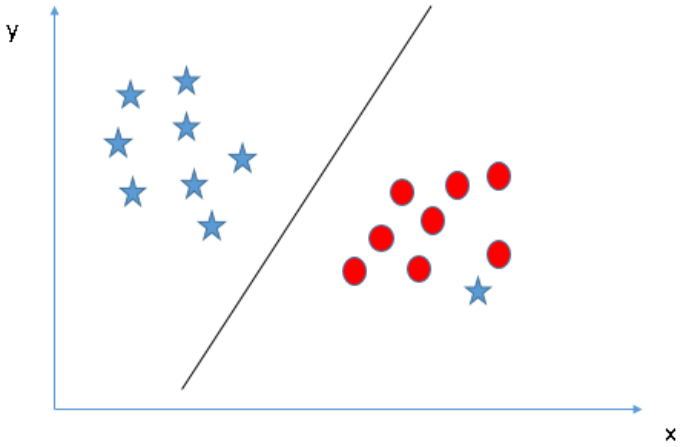
Identify the right hyper-plane (Scenario-3): Hint: Use the rules as discussed in previous section to identify the right hyper-plane



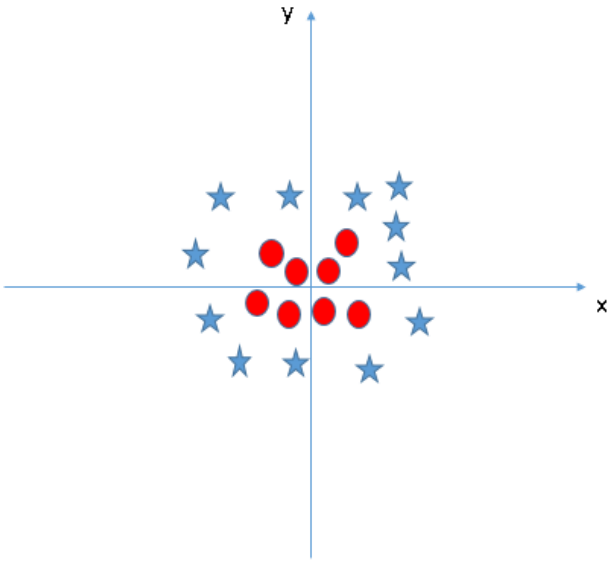
Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

Can we classify two classes (Scenario-4): Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as anoutlier.

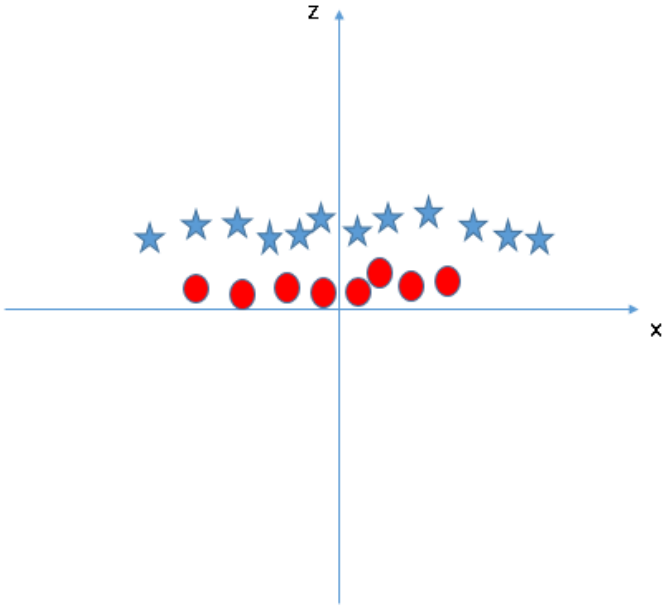
As I have already mentioned, one star at another end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



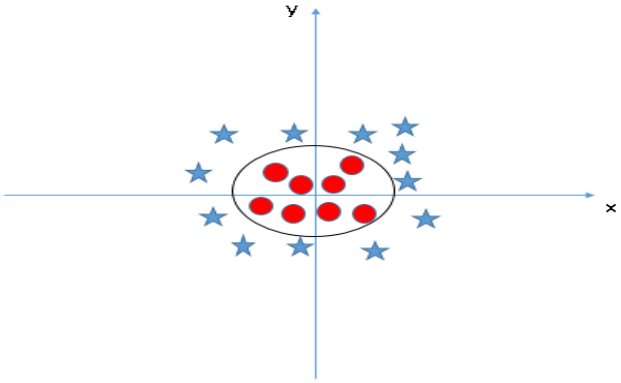
Find the hyper-plane to segregate to classes (Scenario-5): In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.



SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let's plot the data points on axis x and z:



In above plot, points to consider are:
All values for z would be positive always because z is the squared sum of both x and y
In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.
In SVM, it is easy to have a linear hyper-plane between these two classes. But another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the kernel trick. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e., it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs you've defined. When we look at the hyper-plane in original space it looks like a circle:



COMPARISON BETWEEN KNN AND SVM

KNN classifies data based on the distance metric whereas SVM need a proper phase of training. Due to the optimal nature of SVM, it is guaranteed that the separated data would be optimally separated. Generally, KNN is used as multi-class classifiers whereas standard SVM separate binary data belonging to either of one class. For a multiclass SVM, One-vs-One and One-vs-All approach is used. In One-vs-one approach, we have to train n*(n-1)/2 SVMs: for each pair of classes, one SVM. We feed the pattern which is unknown to the entity and the final verdict on the type of data is decided by majority result among all results of all SVMs. This approach is used mostly used in multiclass classification. When it comes to One-vs-All approach, we have to train as many SVMs as there are classes of unlabelled data. As in the other approach, we give the unknown pattern to the system and the final result if given to the SVM with largest decision value. Although, SVMs look more computationally intensive, once training of data is done, that model can be used to predict classes even when we come across new unlabelled data. However, in KNN, the distance metric is calculated each time we come across a set of new unlabelled data. Hence, in KNN we always have to define the distance metric. SVMs have two major cases in which classes might be linearly separable or non -linearly separable. When the classes are non-linearly separable, we use kernel function such as Gaussian basis function or polynomials. Hence, we only have to set the K parameter and select the distance metric suitable for classification in KNN whereas in SVMs we have to select the R parameter (Regularization term) and also the parameters for kernel if the classes are not linearly separable.

When we talk about accuracy of both of the classifiers, SVMs usually have higher accuracy than KNN

Classifier	Training Set	Test Set	Accuracy rate (in %)
SVM	10,000	10,000	98.9
KNN	10,000	10,000	96.47

The results and observations show that SVMs are a more reliable more of classifiers. However, KNN is less computationally intensive than SVM. Since, KNN is easy to implement, the classification of Multi-class data should be done with kNN. The algorithm that guarantees reliable detection in unpredictable situations depends upon the data. If the data points are heterogeneously distributed, both should work well. If data is homogenous to look at, one might be able to classify better by putting in a kernel into the SVM. For most practical problems, KNN is a bad choice because it scales badly - if there are a million labelled examples, it would take a long time (linear to the number of examples) to find K nearest neighbors.

DESIGN METHODOLOGY

This chapter provides a brief explanation about the working principle and methodology of Gesture Keyboard Type letters by moving in the air. It is quite evident by now that the whole project comprises of two phases. First phase determines hand gesture by building the motion tracking device with the help of MPU6050 sensor and Arduino Uno. Whereas, in the second phase gesture recognition software is created using python programming language.

6.1 Circuit Diagram

The Figure 3.1 shows the circuit diagram for gesture keyboard. Arduino will work for this project; we will also need MPU-6050 accelerometer and a push button switch.

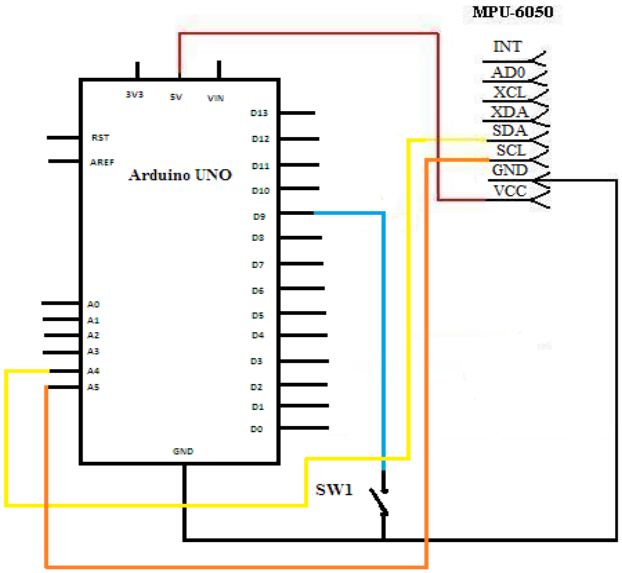


Fig 3.1 Circuit Diagram of Gesture Keyboard

Gesture Keyboard is a library used to convert accelerometer data to a sequence of characters and sentences. Gesture recognition typically involves a limited vocabulary set. It is relatively easy to collect sufficient data of each gesture and straight forward to model each gesture directly from its own recordings. However, the vocabulary of air-writing can easily be thousands of words, and it is difficult to collect enough data for every word in the vocabulary. In order to get the accelerometer data, build a module using an Arduino, an MPU-6050 as accelerometer, the module starts to send accelerometer data to the project; the library is written in Python and uses Scikit-learn “Support Vector Machine algorithm” to classify the signals into letters.

Arduino UNO

Arduino is an open-source platform that offers both hardware and software platform functionalities for interactive projects that have the control and sensing capability providing an extension to connect to other such controls. Arduino board uses a micro-controller board along with 14 digital input and output configuration pins and 6 analog inputs are used. The output of the accelerometer sensor is given to three analog inputs of the Arduino board. The inbuilt ADC of the board converts the values to digital and sends the data to the Wireless or Bluetooth module through Serial Port. Arduino can be programmed through its IDE software USB cable. It can be powered by a USB cable. In our project this plays an important role which provides a platform to run a program.

Accelerometer module-MPU 6050

An accelerometer is a micro electromechanical sensor (MEMS) that measures acceleration. It is also helpful in detecting tilt or orientation with respect to the earth of a device it is attached to. This application uses its property to detect the tilt of the hand to move the mouse pointer. The continuous data streams are divided into individual gesture according to the button pressing label. An accelerometer can sense keenly the acceleration data of three spatially placed orthogonal axes in each gesture in a given sampling frequency. A gesture can be denoted as:

G = (Ax, Ay, Az)

Here, Ax, Ay, Az is the acceleration vector of an axis and *L* is the length of the temporal sequence. To describe the whole gesture but distinguish the periods from each other, we divide a gesture into *N*+1 segment identical in length, and then every two adjunct segments make up a frame. For the feature extraction of each gesture, firstly, it is divided into *N* frames. The feature vector is eventually put into a classifier in order to train a classification model or retrieve a recognized gesture type. More specifically, a gesture can be represented as: Intuitively, more frames a gesture is broken up into, more details we know about the gesture. However, it may lead to the over-fitting problem if the frame number *N* becomes large. It will also increase the dimension of the feature space, which increases computational complexity. We will conduct an experiment to determine the optimal frame number *N* later.

Algorithm for gesture recognition with accelerometer

Step1: Start the Transmitter by providing power supply to the module.
Step2: Once the supply is given, Arduino uno starts communication with MPU-6050 by sending the slave address.
Step3: After the slave address has been sent, some registers of MPU 6050 have to be programmed.
Step4: Now, the registers of Accelerometer (X, Y, Z-axis registers) are read.
Step5: These values are sent to the Processor through USB.

Switch

The switch is used to control device operation for toggling activation of the sensor. When someone wants to signal and record gestures, we first press the button, and hence the module starts to send accelerometer data to the computer connected. At the moment when the button is released, the transmission stops immediately.

Arduino uno and Accelerometer Connection

The circuit connection between MPU-6050 and Arduino Uno is connected as shown in the figure below. The operating range of the MPU-6050 is 2.375-3.4V is as stated in the datasheet. Thus, we will directly use the ready 5V voltage source that the Arduino is able to provide. The Ground (GND) and AD0 are connected to both the ground pin on Arduino. The AD0 pin functions to defining the default I2C address and thus we need to ground it for the default evaluation purposes. SDA and SCL on the sensor module are then connected to the dedicated I2C communication on the Arduino which is the A4 and A5 analogue pin. As for obtaining the raw sensor values, the MPU-6050 outputs a digital signal from the ADC for each of the sensors. The sensors values are represented by 16-bit 2's complement format. Take example for a ±2g full scale range accelerometer the bits that are able to represent the scale are 2^{16}-1 which is a total of 65535 bits. The coding of Arduino to use MPU-6050 is as referred from open source. The main processing of all sensor data happens in the microcontroller which will be the main components of the project. The below fig 3.2 shows Arduino uno and mpu 6050 connection.

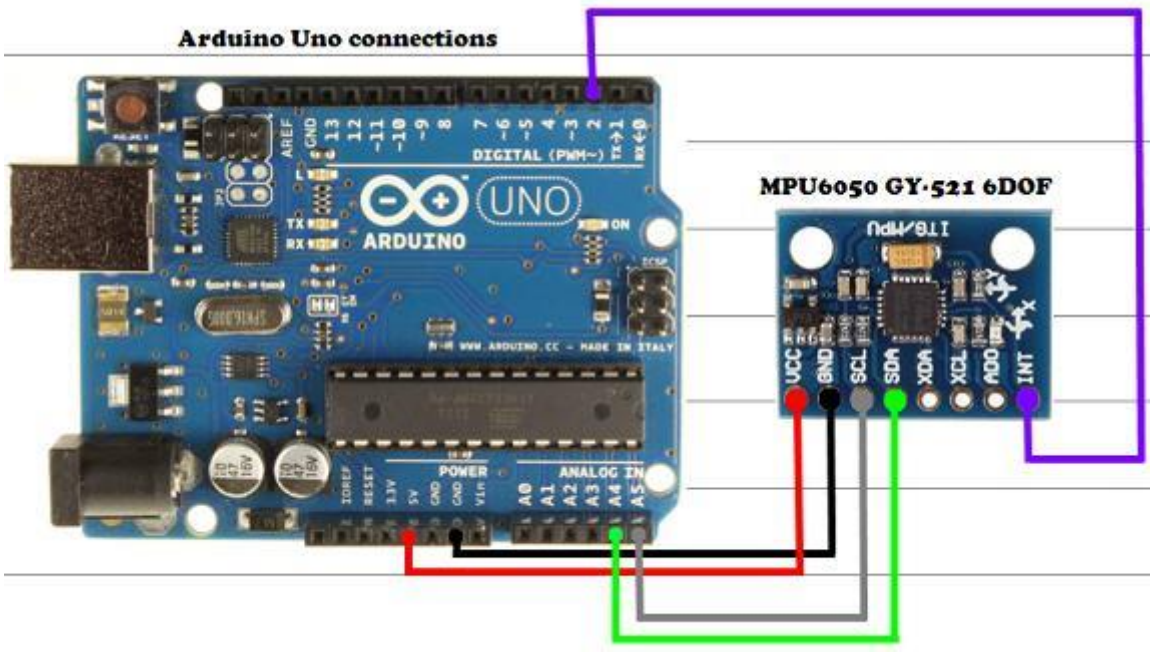


Fig 3.2 Arduino uno MPU 6050 connection

The initial result as from the Arduino sketch. The sensor is place on a table with no movements to test the validity of the sensors. The baud rate is set to 9600 which can be monitored on Arduino.

Creation of gesture recognition software

Scikit-learn is a programming library in machine learning for the programming in Python. It highlights distinctive characterization, backslide and clustering calculations counting reinforce vector machines, self-assertive forests, slant boosting and is aiming to interoperate with the Python numerical and coherent libraries. Gesture Recognition or any other machine learning tasks that uses supervised learning goes through a process of stages before the final classifier produces an efficient output. All the stages are indecent of each other and play a unique role in contributing to the accuracy of classification. The fig 3.3 Flow diagram for machine learning algorithm.

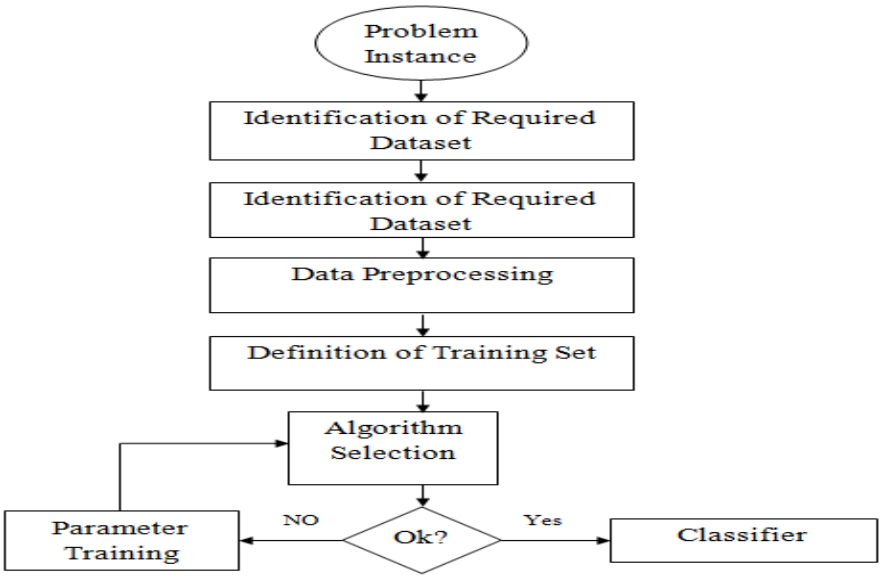


Fig 3.3 Flow diagram for machine learning algorithm

Required dataset

The first step is to fabricate the data attributes that are crucial in determining the goal of the output. All values must have a constraint or an upper and lower bound limit to initiate remission of boundary limits. The dataset is then made available to be identified in terms of different parameters that can further extracted. The set is then made available to be identified in terms of different parameters that can further extracted.

Data Pre-Processing

Depending on the type of problem, the researchers have concluded that the pre-processing of data may vary on disparate elements of attributes. For example, instance selection is not only used to handle noise but to cope with the infeasibility of learning efficiently from heavy datasets. Instance selection in such datasets is an optimization problem that aims to maintain the mining quality while attempts to degrade the sample size. It reduces data and enables a data classification algorithm to function and work effectively with very large or heavy datasets. There is a variety of procedures for sampling instances from a large dataset.

Training Set

Once a clear input variable is made available on a global scope with logistic and constraint satisfying training inputs can be fetched at every input cycle, the similar categories of elements are classified together as a distinct entity. This is called as the training set and serves as the basis of classification space to make experiments and observe trends. The set consists of attributed key-value pairs of abundance different instances that cumulatively make the set.

Algorithm Selection

In this step, the documents are split into training and testing documents, the training documents are used to train the system (i.e., learn the system) to recognize different patterns of categories, the testing documents are used to evaluate the system, the process of categorization depends on the algorithm used. The choice of which specific learning algorithm we should use is a critical step. Once we complete the preliminary testing and if achieved as to be satisfactory, the classifier that maps the unlabeled instances into classes is available for the routine to use. The classifier's evaluation is most often based on prediction accuracy (the percentage of correct prediction divided by the total number of predictions). There are three techniques used to calculate a classifier's accuracy which is essential One technique is to split all the training set by using two-thirds of it for training and the other third of it for estimating their performance. The average of the error rate of each subset is roughly an estimate of the error rate of the classifier.

Classification Algorithms for Gesture Recognition

In this section, we first elaborate on our data set and target values and then begin to evaluate the performances of algorithm for training of gestures.

Gesture Dataset

To improve precision, we will create a new data set and evaluate their trends. Each time an accelerometer position is moved or a different accelerometer is used, the device becomes less accurate. So, it is important to keep adding the training data to learning set in added time sequences. The algorithm can feature a maximum of 40different gestures, such that each gesture is associated with a single letter of the English alphabet. The first step is to begin recording a new batch of data inputs for gestures by pressing the online button, the library records the data from movements of accelerometer using the following markup Target: Argument tells module it wants to record new gesture samples

a: It is the one that characterizes the unique gesture of length 1

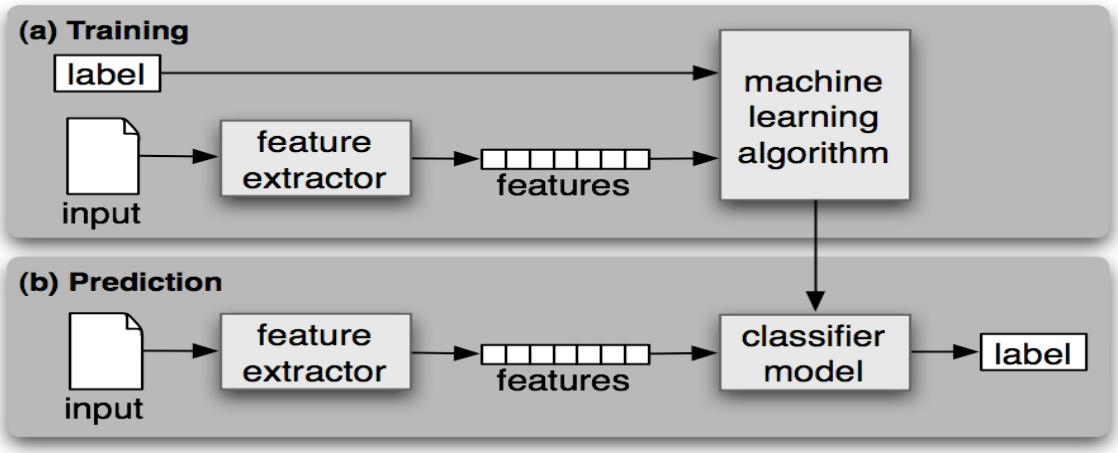
0: Batch Number; It must be different each time we begin to register a new batch in order to avoid overriding

Port: The port represents the serial port connection to the Micro-controller/Arduino

Ideally, each sample must be recorded and saved as a different file in the data folder and is later evaluated. Once the dataset is ready a suitable model is selected to train these inputs. The choice of a perfect algorithms can be a daunting and challenging task; hence we aim to compare 2 of the leading classification techniques on our dataset and evaluate the best model.

Classification Algorithms

Classification algorithms use a set of variables such as "feature variable" or "independent variable" or "predictor variable" for making predictions and "class variable or dependent variable" for holding outcome of prediction. These algorithms are driven by "class variable" for the given data that can be either "categorical" or "continuous". "Classification" techniques are used in cases where the class label is categorical and "Regression" techniques are used in cases where the class label is continuous. The classifier selection is based on prediction accuracy of algorithm which is measured by a parameter called "Confusion matrix". Confusion matrix is used to assess percentage of test data that has been correctly labeled by the classifier.

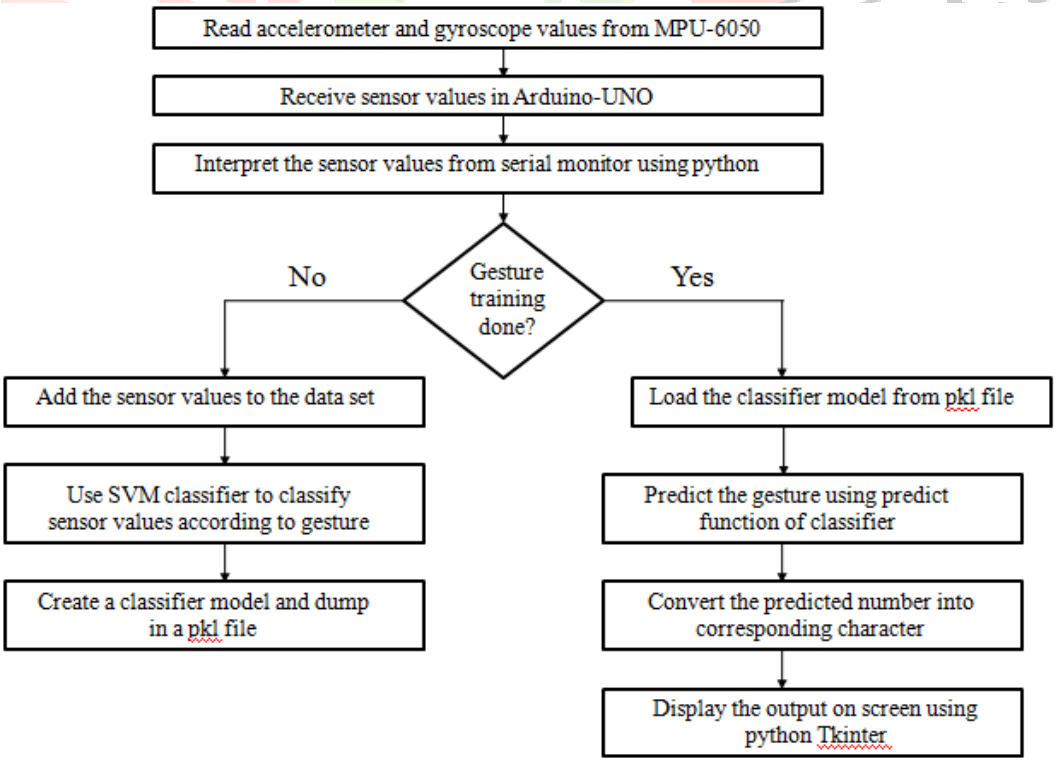


Supervised Classification of data

IMPLEMENTATION

In the transmitter side an Arduino Uno is used as the controlling unit, the accelerometer is connected to Arduino through I2C communication that measures the gravitational acceleration in X, Y and Z direction. In the receiver side Arduino receives the data from the transmitter 6050 module has a 5V pin connect it to Arduino's 5V pin, Next, the GND of the Arduino is connected to the GND of the MPU 6050. To setup the I2C lines, connect the pin labelled SDA on the MPU 6050 to the Arduino's analog pin 4(SDA), and the pin labelled as SCL on the MPU 6050 to the Arduino's analog pin 5(SCL).The accelerometer data from MPU 6050 is send to Pectin PC the library is written in Python and uses Scikit-learn "Support Vector Machine algorithm" to classify the signals into Pectin-learn is to an extraordinary degree composed in Python, with a few center calculations composed in Notepad++ to achieve execution. Support vector machines are executed by a Notepad++ wrapper around LIBSVM.

Gesture Keyboard Work Flow



Flow Diagram of Gesture Keyboard.

Working of the system initially the machine won't know anything. The machine has to read the gestures which are made by the hardware. The gestures are read through the MPU 6050 sensor attached to the Arduino. The sensor values provide the continuous location of the hand used to make the gesture. A combination of the sensor values will be used to estimate the path taken by the hand while making the gesture. The sensor values will be used to form a dataset. The sensor values will be mapped to the corresponding character which was specified by the user while training the machine. This mapping will be used to predict the character from the gesture made in thin air. For example, making an 'a' in the air with our hand can be mapped to print the letter 'a' on the computer screen. Initially the user has to feed the preferred gesture to the program and after doing that the user has to train the machine by testing it continuously. This way the machine comes to
know the gestures required for a particular operation.

Collecting Data from Gestures

The accelerometer data from MPU 6050 is send to PC. In PC the library is written in Python and uses Scikit-learn "Support Vector Machine algorithm" to classify the signals into Pectin-learn is to an extraordinary degree composed in Python, with a few center calculations composed in Notepad++ to achieve execution. Support vector machines are executed by a Notepad++ wrapper around LIBSVM. The fig 3.9 shows the Python Machine Learning Script.

IV. CONCLUSION AND FUTURE SCOPE

There is a need for more efficient, user-friendly procedure for interaction with devices, touch screens, and applications. This can be done by using video-based noncontact interaction techniques. Our proposed system is currently being developed for text writing and invoking other applications by these gestures drawn in air.

Gesture based writing alphabets into the text pad and writing numbers into the text pad. Machine learning approach, teach once and start predicting for gesture inputs. Fast gesture recognition. Better performance. More accuracy to predict right gesture input.

V. REFERENCES

[1]Saria Beg, M. Fahad Khan and Faisal Bag, "Text Writing in Air", Journal of Information Display Volume 14, Issue 4, 2013

[2]Maryam Khosravi Nahouji,"2D Finger Motion Tracking, Implementation For Android Based Smartphones",Master's Thesis, CHALMERS Applied Information Technology, ,2012,pp 1-48

[3]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

[4]P. Ramasamy, G. Prabhu and R. Srinivasan, "An economical air writing system converting finger movements to text using web camera," 2016 International Conference on Recent Trends in Information Technology (ICRTIT), Chennai, 2016, pp. 1-6.

[5]Kenji Oka ,Yoichi Sato and Hideki Koike, "Real-Time Fingertip Tracking and Gesture Recognition", IEEE Computer Graphics and Applications, 2002, pp.64-71.

[6]Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters.

[7] M.D. Zeiler, ADADELTA: An Adaptive Learning Rate Method. CoRR 2012.

[8]Vladimir I. Pavlovic ,Rajeev Sharma and Thomas S. Huang, "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review", IEEE Transactions On Pattern Analysis and Machine Intelligence, VOL. 19, NO. 7, JULY 1997,pp.677-695

[9]Guo-Zhen Wang, Yi-Pai Huang, Tian-Sheuan Chang, and Tsu-Han Chen, "Bare Finger 3D Air-Touch System Using an Embedded Optical Sensor Array for Mobile Displays",Journal Of Display Technology, VOL. 10, NO. 1, JANUARY 2014, pp.13-18

[10]Napa Sae-Bae,Kowsar Ahmed,Katherine Isbister, NasirMemon, "Biometric-rich gestures: a novel approach to authentication on multi-touch devices", Proc. SIGCHI Conference On Human Fcators in Computing System,2005, pp.977-986