

SPEECH LANGUAGE DISORDER IDENTIFICATION USING DEEP LEARNING TECHNIQUES

A PROJECT REPORT

Submitted by

S.SNEHAA

17CSR192

B. SWATHI

17CSR209

V. VENKATESH

17CSR220

in partial fulfillment of the requirements for the

award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638 060

APRIL 2021

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING KONGU
ENGINEERING COLLEGE**

(Autonomous)

PERUNDURAI ERODE – 638060

APRIL 2021

BONAFIDE CERTIFICATE

This is to certify that the Project Report entitled **SPEECH LANGUAGE DISORDER IDENTIFICATION USING DEEP LEARNING TECHNIQUES** is the bonafide record of project work done by **S.SNEHAA (Register no: 17CSR192), B.SWATHI (Register no: 17CSR209), V.VENKATESH (Register no: 17CSR220)** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in **Computer Science and Engineering** of Anna University, Chennai during the year 2020 - 2021.

SUPERVISOR

HEAD OF THE DEPARTMENT

(Signature with seal)

Date:

Submitted for the end semester viva voice examination held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
KONGU ENGINEERING COLLEGE
(Autonomous)
PERUNDURAI ERODE – 638060
APRIL 2021

DECLARATION

We affirm that the Project report titled **SPEECH LANGUAGE DISORDER IDENTIFICATION USING DEEP LEARNING TECHNIQUES** being submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering is the original work carried by us. It has not formed the part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

S.SNEHAA
(Reg.No.:17CSR192)

B.SWATHI
(Reg.No.:17CSR209)

V. VENKATESH
(Reg.No.:17CSR220)

Date:

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date:

Name & Signature of the Supervisor with seal

ABSTRACT

Communication is an important way of expressing yourself. But there are people who face communication disorders like stammering and stuttering. These days many people are significantly affected by communicative disorders. There are mainly three types of communicative disorders namely language disorders, speech production disorders and oral motor/swallowing/feeding disorders. These problems being unidentified, and not being treated may cause problems for the children, like emotional and behavioral problems. Identifying the problem earlier and giving intervention at right time is very important for children to improve in their personal and academic life. The sooner the problem is identified the better it is for the person.

Identification of the disorder is a complex decision making process which requires expertise in that field. Parents often hesitate to approach experts due to social stigma. Using deep learning techniques in identifying the disorder becomes necessary in such an environment. The application gets the voice of the person first, with the voice recording, one can predict whether the person has any speech disorders. The application can identify if the person has a stuttering problem, dysarthria and or whether he can talk normally. Hence, this proposal aims to include a novel and innovative deep learning based method to identify speech disorders by recording the person's voice.

ACKNOWLEDGEMENT

Owing deeply to the supreme, we extend our thanks to the Almighty, who has blessed us to come out successfully with our project. We take pleasure to express our deep sense of gratitude to our beloved parents.

We express our sincere thanks and gratitude to our beloved correspondent **Thiru.P.Sachithanandan** for giving us the opportunity to pursue this course.

We are extremely thankful with no words of formal nature to the dynamic Principal **Dr.V.Balusamy MTech, PhD** for providing the necessary facilities to complete our work.

We would like to express our sincere gratitude to our respected Head of the Department **Dr.N.Shanthi M.E., PhD.**, for providing necessary facilities.

We extend our thanks to **Dr.E.Gothai M.E., Ph.D** the project coordinator for her encouragement and valuable advice that made us to carry out the project work successfully.

We extend our gratitude to our supervisor **Dr.C.S.Kanimozhi Selvi M.E., Ph.D.**, for her valuable ideas and suggestions, which have been very helpful in the project. We are grateful to all the faculty members of the Computer Science and Engineering Department, for their support.

TABLE OF CONTENTS

CHAPTER No	TITILE	PAGE No.
I	ABSTRACT	iv
II	ACKNOWLEDGEMENT	v
III	LIST OF FIGURES	viii
IV	LIST OF TABLES	ix
V	LIST OF ABBEREVIATION	x
	INTRODUCTION	1
1.	1.1. DEEP LEARNING TECHNIQUES	2
	1.2. EXISTING SYSTEM	6
	1.3. OBJECTIVE	6
2.	LITERATURE REVIEW	7
3.	SYSTEM REQUIRMENTS	
	3.1. HARDWARE REQUIREMENTS	10
	3.2. SOFTWARE REQUIREMENTS	10
	3.3. SOFTWARE DESCRIPTIONS	11

4.	PROPOSED SYSTEM	13
	4.1. WORKING MODEL OF A PROPOSED SYSTEM	13
	4.2. MODULES DESCRIPTION	14
5.	RESULTS AND DISCUSSION	20
6.	CONCLUSION AND FUTURE WORK	26
7.	APPENDIX 1 - SAMPLE CODING	27
	APPENDIX 2 - SCREENSHOTS	41
8.	REFERENCES	43

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1.1	Convolutional Neural Networks	3
1.2	Convolutional Layer	4
4.1	Workflow of the application	13
5.1	Epoch vs Accuracy Graph	21
5.2	Epoch vs Error Graph	22
5.3	Website GUI - Audio Record	23
5.4	Website GUI - Audio Upload	23
5.5	Website GUI - Prediction	24
5.6	Website GUI - Prediction	24
5.7	Website GUI - Prediction	25
A1	Output Screenshot	41
A2	Output Screenshot	41
A3	Output Screenshot	42
A4	Output Screenshot	42

LIST OF TABLES

TABLE No.	TABLE NAME	PAGE No.
3.1	Hardware Requirements	10
3.2	Software Requirements	10
5.1	Performance Metrics	21

LIST OF ABBREVIATION

SLPs	-	Speech-Language Pathologists
GPL	-	General Public License
ML	-	Machine Learning
IDE	-	Integrated Development Environment
MFCC	-	Mel - Frequency Cepstral Coefficients
CNN	-	Convolutional Neural Network
ConvNet	-	Convolutional Neural Network

CHAPTER 1

INTRODUCTION

Communication is one's ability to express themselves to others in an effective way. Skillful communication is essential to succeed in all human activities. Speech disorders are most common problem seen in people these days. Individuals with speech disorders may be unable to participate fully and competently in everyday interpersonal, learning, and occupational situations. Limited participation in those life activities due to their problem leads to associated emotional and behavioral problems.

Speech disorders affect a person's ability to produce sounds that create words. They are not the same as language disorders, which make it more difficult for people to learn words or understand what others are saying to them. Types of speech disorder include stuttering, apraxia, and dysarthria.

Stuttering is a speech disorder characterized by repetition of sounds, syllables, or words; prolongation of sounds; and interruptions in speech known as blocks. Dysarthria often causes slurred or slow speech that can be difficult to understand. Common causes of dysarthria include nervous system disorders and conditions that cause facial paralysis or tongue or throat muscle weakness. Certain medications also can cause dysarthria.

Speech-language problems are the most common disability of childhood yet they are the least well detected, particularly in primary care settings. The goal of this proposal is to identify speech disorders using deep learning based web application. Web application is easy to use and available for free which makes it easy and efficient.

1.1. DEEP LEARNING TECHNIQUES

Deep learning is a subset of machine Learning. It is a concept which allows the machine to learn from examples and experience. Learning can be classified as supervised and unsupervised. Supervised learning trains a model on known input and output audio data so that it can predict future outputs. It was further divided into classification and clustering. Unsupervised learning finds hidden patterns or intrinsic structures in input data. Under unsupervised learning there is clustering. Deep learning plays a vital role in health care, because of its ability to process a large number of audio datasets beyond human's capability and convert the analyzed data into clinical insights.

Identification of these disorders at early stage can be really helpful for the person. Using deep learning techniques to do that is one creative idea. First get the dataset and train the model to identify the disorder. A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take an input image, assign importance to various aspects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

Since CNNs are hungry for images, it is necessary to transform the sound into an image. The audio signals with respect to time and frequency is plotted. The network is made to go through 30 epochs. The pixels of the image are fed in the form of arrays to the input layer of the neural network. The hidden layers carry out feature extraction by performing different calculations and manipulations. Finally, the fully connected layer identifies the word. The process in the CNN model: The pixels from the image are fed to the convolutional layer that performs the convolution operation and results in a convolved map. The convolved map is applied to a ReLU function to generate a rectified feature map. The image is processed with multiple convolutions and ReLU layers for locating the features.

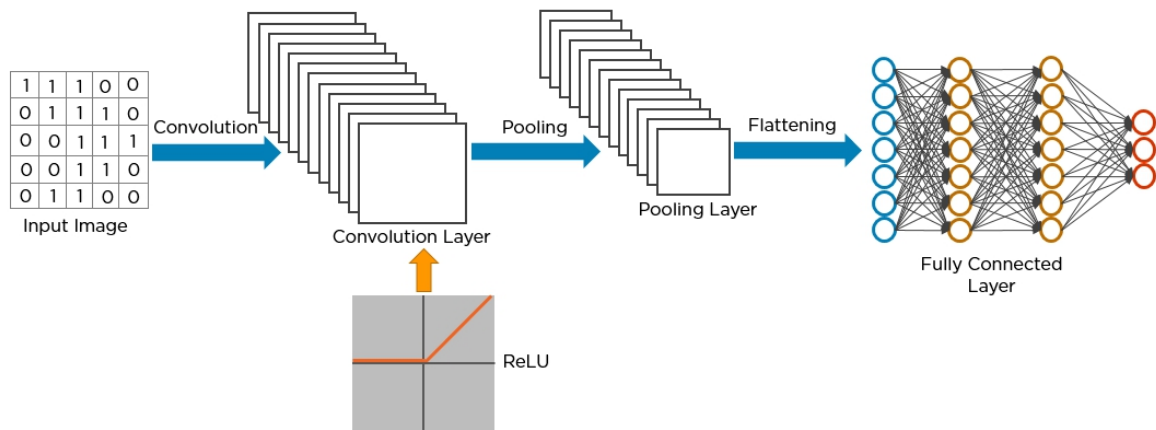


Figure 1.1. Convolutional Neural Networks

Different pooling layers with various filters are used to identify specific features. The pooled feature map is flattened and fed to a fully connected layer to get the final output.

Convolutional Neural Networks

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in a data through the application of relevant filters. The architecture performs a better fitting to the dataset due to the reduction in the number of parameters involved and re-usability of weights. In other words, the role of the ConvNet is to reduce the data into a form which is easier to process, without losing features which are critical for getting a good prediction. This is important while designing an architecture which is not only good at learning features but also is scalable to massive datasets.

Convolutional Layer — The Kernel

The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K . In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between K_n and stack $([K_1, I_1]; [K_2, I_2]; [K_3, I_3])$ and all the results are summed with the bias to give a squashed one-depth channel Convolved Feature

Output. The objective of the Convolution Operation is to extract the high-level features of a data. ConvNets need not be limited to only one ConvLayer. The first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features, giving a network which has the full understanding of the dataset.

There are two types of results to the operation - one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying Valid Padding in case of the former, or Same Padding in the case of the latter.

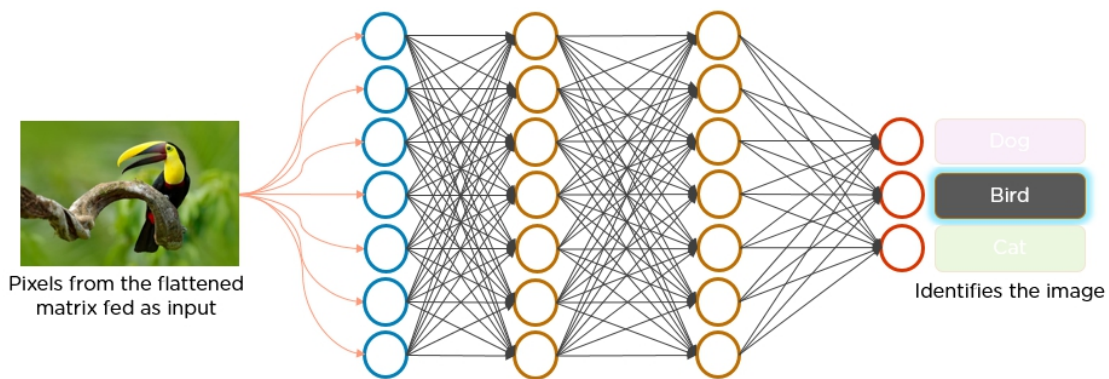


Figure 1.2. Convolutional Layer

ReLU Layer

The ReLU layer applies the function $f(x) = \max(0, x)$ to all of the values in the input volume. In basic terms, this layer just changes all the negative activations to 0. This layer increases the nonlinear properties of the model and the overall network without affecting the receptive fields of the conv layer. The input usually contains a lot of non-linear features. The rectifier serves to break up the linearity. In general, what the rectifier function does to an input is remove all the black elements from it, keeping only the pixels carrying a positive value. The essential difference between the non-rectified version and the rectified one is the progression of colors. The gradual change in color will no longer be there which indicates the disposed linearity.

Pooling Layer

The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. It is to decrease the computational power required to process the data through dimensionality reduction. Pooling layer is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling, Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Max Pooling performs as a Noise Suppressant. It discards the noisy activations and performs de-noising and dimensionality reduction. Max Pooling performs a lot better than Average Pooling.

The Convolutional Layer and the Pooling Layer, together form the i -th layer of a Convolutional Neural Network. Depending on the complexities in the input, the number of such layers may be increased for capturing low-levels details, but at the cost of more computational power. The model is enabled to understand the features. The final output is flattened and fed to a regular Neural Network for classification purposes.

Fully Connected Layer

Fully Connected Layer is fed forward to neural networks. Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened.

The output of convolution/pooling is flattened into a single vector of values, each representing a probability that a certain feature belongs to a label. The input values flow into the first layer of neurons. Then it is multiplied by weights and passed through an activation function (typically ReLu). It is then passed forward to the output layer, where every neuron represents a classification label. The fully connected part of the CNN network goes through its own backpropagation process to determine the most accurate weights. Each neuron

receives weights that prioritize the most appropriate label. Finally, the neurons “vote” on each of the labels, and the winner of that vote is the classification decision. This is how a CNN model works.

1.2. EXISTING SYSTEM

Speech disorder identification is performed by speech-language pathologists, who are often referred to as speech therapists. Therapists use a variety of tools and technologies to assist with evaluation, diagnosis and rehabilitation of individuals both young and old. These include articulation method, language intervention activities, and others depending on the type of speech or language disorder. There are techniques like Voice Synthesizer, Analytical Software and Tablet Computer are the frequently used tools by the physicians. To particularly focus on fluency improving, there is no proper therapy tools, or any tools available.

1.3. OBJECTIVE

The project aims to include a novel and innovative ML based method to identify speech disorders by recording the person’s voice. Some people may drag out syllables or may talk breathlessly, or seem tense while trying to speak. If you clutter, you often speak fast and merge some words together or cut off parts of them. So this project will help the person to identify their speech language disorder in speaking easily at an early stage, in an efficient way and fun way.

CHAPTER 2

LITERATURE REVIEW

Alan Preciado Grijalva and Ramon F. Brena [2018] have developed level assessment for foreign language students. This project works on audio processing system that classifies the level of fluency of non-native English speakers by using five different machine learning models. Here, in this own data set is taken that consists of labeled audio conversation in English between people which ranges in different classes like low, intermediate, high. Each model had the ability to get accuracy of 90%. The accuracy is high for the considered data set.

R.Gaines and C.Missiuna [2006] have proposed a method for early identification of speech toddlers. Developmental Coordination Disorder (DCD) is a movement skill disorder which impacts upon a child's ability to perform age-appropriate self-care and academic tasks. The project was conducted to determine whether children who had been identified with speech delays as toddlers demonstrated characteristics of DCD and speech problems at kindergarten age. As a result of 40 children, 18 showed evidence of significant motor impairment and two-thirds of these met diagnostic criteria for DCD at follow-up. Twelve children were identified as having persistent speech problems and of these nine presented with significant motor coordination difficulties. Clinical implications for early recognition of motor issues by speech/language pathologists and the potential use of parental reporting tools are addressed.

Bastanfard, et al. [2010] have developed a software system which facilitates the interaction and synchronization of language learning activities for Persian hearing-impaired children. Shortcomings in the current process of speech therapy motivated us to develop a software system for Persian children with articulation disorders. The main aim of the system is to facilitate the interaction and synchronization of language learning activities conducted both at home and in speech therapy centers during the therapy process. A creative, entertaining

combination of multimedia material is incorporated into the system. The methodology is practical and cost-effective and can be extended to other relatively uncommon languages. The preliminary test results show the successful integration of the designed system into the convenient method used by speech therapists in Iran.

Toki, E. I., & Pange [2010] introduced an e-learning system for improving articulation in Greek preschoolers. Over the past decade speech and language therapy has taken an interesting turn towards the use of Information Communication Technologies (ICTs) for diagnosis of disorders and delivery of therapy. In many cases ICTs have worked as assistive tools to therapists, while in others as sole providers of therapy, especially in remote areas. In this work authors provide a brief overview of the most representative articles for applications and assistive technologies used for assessment and intervention purposes in Speech Therapy according to the type of disorders. The results of the study on this software showed that children not only improved their articulation but they also increased on language activities success by acquiring new vocabulary.

Amandeep Kaur and Jubilee Padmanabhan [2017] have highlighted the need and importance of early identification of the students with specific learning disorder, they also focus on the various tools and techniques for the screening of SpLD; national and international level programs and policies and school based interventions that can facilitate the learning. For better understanding of every aspect of SpLD is very essential for the teachers, as he/ she has the responsibility towards such students being specially able children and it is necessary to guide and train them in proper direction. While highlighting the need and importance of early identification of the students with specific learning disorder, this work will focus on the various tools and techniques for the screening of SpLD.

Kanwajit Kaur¹ & S. Pany [2017] have reviewed the computer based interventions which were used to improve social skills of autism spectrum disorder children. This work review addresses two systematic research questions: How the computer based intervention is used or developed and the effectiveness of computer based intervention for autism spectrum disorder children in improvement of social skills. Therefore, the specific objectives of this work are described as; to review the computer based interventions which were used to

improve social skills of autism spectrum disorder children; and to analyse the findings of the previous work. The analysis of different studies revealed that computer based games are popularly used to improve the social skills of the ASD children and it is also observed that computer based interventions proved to be the useful interventions to improve the social skills of autism spectrum disorder children.

Sai Aishwarya Ramani and Amudhu Sankar [2017] focused on the development of a prototype “ISpeak” for Augmentative and Alternative Communication (AAC) which includes all forms of communication (other than speech) that are used to express thoughts, needs and ideas. People with severe speech or language difficulties rely on AAC to supplement existing speech. People with severe speech or language difficulties rely on AAC to supplement existing speech. This increases social interaction, school performance, and feeling of self esteem. Studies in the past have mentioned greater results in the usage of AAC. This study focused on the development of a prototype AAC “ISpeak” which is cost efficient and can be easily maintained. This increases social interaction, school performance, and feeling of self esteem. Studies in the past have mentioned greater results in the usage of AAC.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

Table 3.1 Hardware Requirements

Processor	DELL
Processor Speed	1.80 GHz
Hard Disk	1 TB
RAM	4GB

3.2 SOFTWARE REQUIREMENTS

Table 3.2 Software Requirements

Programming language	Python 3.5
Software	Anaconda Navigator 1.9.7 (Jupyter Notebook 6.0.3, Spyder 4.1.5)
Operating system	Window 10

3.3 SOFTWARE DESCRIPTION

Python

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python source code is also available under the GNU General Public License (GPL). It provides constructs that enable clear programming on both small and large scales. It features a dynamic type system and automatic memory management, supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python is open source software and has a community-based development model. Python and C Python are managed by the non-profit Python Software Foundation.

Following are the features of Python,

Easy-to-learn – Python has few keywords, simple structure, and a clearly defined syntax. This allows the developers to learn the language quickly.

Easy-to-read – Python code is more clearly defined and visible to eyes.

Easy-to-maintain – Python's source code is fairly easy to maintain.

A broad standard library – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

Interactive Mode – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Databases – Python provides interfaces tall major commercial databases.

GUI Programming – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable – Python provides a better structure and support for large programs than shell scripting.

Anaconda Framework

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda distribution is used by over 15 million users and includes more than 1500 popular data-science packages suitable for Windows, Linux, and MacOS.

Jupyter Notebook

The Jupyter Notebook is an interactive computing environment that enables users to author notebook documents that include: - Live code - Interactive widgets - Plots - Narrative text - Equations - Images – Video. These documents provide a complete and self-contained record of a computation that can be converted to various formats and shared.

The Jupyter notebook combines three components,

- 1.The notebook web application:** An interactive web application for writing and running code interactively and authoring notebook documents.
- 2.Kernels:** Separate processes started by the notebook web application that runs users' code in a given language and returns output back to the notebook web application. The kernel also handles things like computations for interactive widgets, tab completion and introspection.
- 3.Notebook documents:** Self-contained documents that contain a representation of all content visible in the notebook web application, including inputs and outputs of the computations, narrative text, equations, images, and rich media representations of objects. Each notebook document has its own kernel.

Spyder

Spyder, the Scientific Python Development Environment, is a free integrated development environment (IDE) that is pre-installed Anaconda. It is a very powerful editor that provides all necessary features starting from code framing to its deployment. It includes editing, interactive testing, debugging, and introspection features. This means the difference between Anaconda and Spyder is – we can use Spyder just like Jupyter that is running separate code blocks at a time to avoid any kind of errors.

CHAPTER 4

PROPOSED SYSTEM

4.1.WORKING MODEL OF PROPOSED SYSTEM

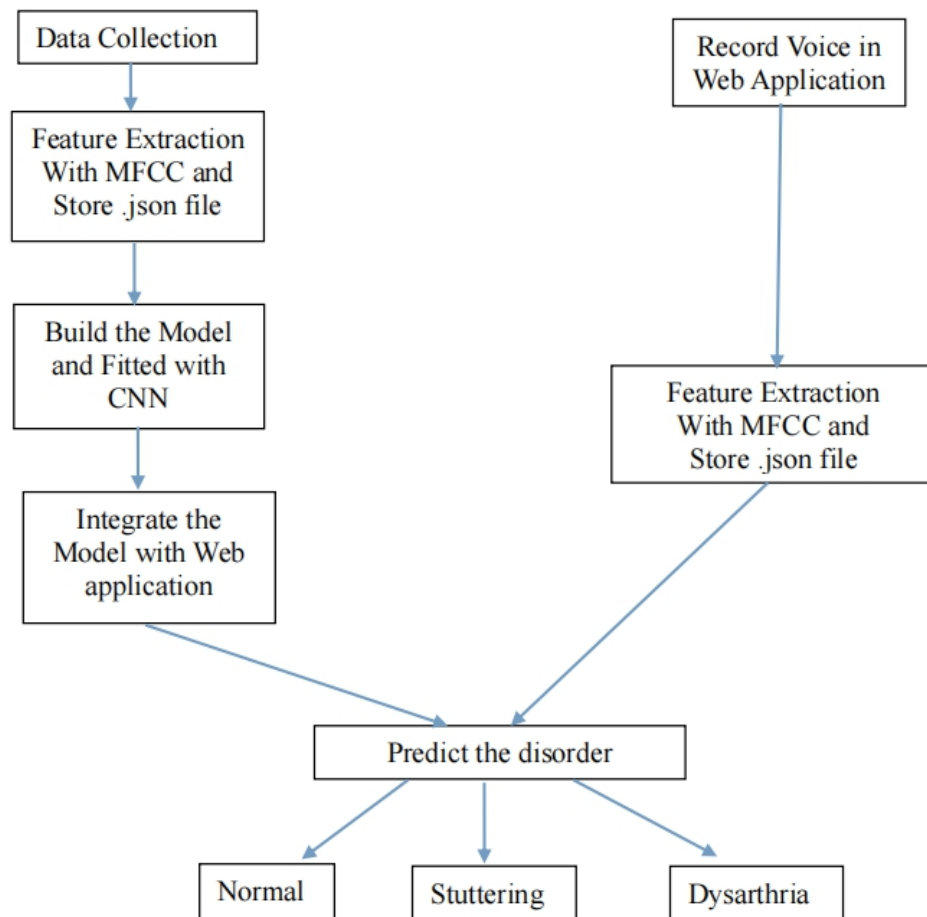


Figure 4.1. Workflow of the Application

4.2. MODULE DESCRIPTION

The aim of the proposed work is to predict the amount of correctness in the speech signal. Machine learning algorithms have been applied to classify the correctness percentage on the basis of audio taken from the observation from the audio recordings. The modules in the proposed system are

1. Data collection
 - 1.1. Data Augmentation
2. Feature Extraction
3. Model Fitting with CNN
4. Website Development

Data collection

Voice recording of people is collected. Recordings consist of people saying small sentence. The dataset is generally of different data formats. So, it is required to convert it to .json format before proceeding. Each audio data that are collected has to be labeled so that the machine learning algorithm can be applied and both input & output data is an important part for supervised learning. To load training dataset from json file,

```
def load_data(data_path):
    with open(data_path, "r") as fp:
        data = json.load(fp)
        X = np.array(data["mfcc"])
        y = np.array(data["labels"])
    return X, y
```

About 1955 audio recordings were collected from a website [research.google](https://research.google.com/tamil-audio-dataset/). It is a tamil audio dataset. The dataset recordings consists only male voice recordings. Recordings consists of small tamil sentences. There are many small sentences. Each time the website is loaded, it displays a different sentence. This audio is then sent for data augmentation to get disordered dataset.

Before proceeding with the model fitting, the train, the test and the validation set needs to be splitted. The data needs to be loaded and then split. It is split as train/test set and train/validation set for both X and Y.

```
X, y = load_data(DATA_PATH)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size)
X_train, X_validation, y_train, y_validation = train_test_split(X_train, y_train,
test_size=validation_size)
```

One thing we want to do is make sure our data is balanced. In the case of this dataset, it can be seen that the dataset started off as being balanced. Balanced, mean there are the same number of examples for each class. Also, if you have a dataset that is too large to fit into your ram, you can batch-load in your data.

Data Augmentation

Data augmentation is a strategy that enables practitioners to significantly increase the diversity of data available for training models, without actually collecting new data. Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks. Having a large dataset is crucial for the performance of the deep learning model. However, we can improve the performance of the model by augmenting the data we already have. It also helps the model to generalize on different types of images. In data augmentation, we add different filters or slightly change the images we already have for example add a random zoom in, zoom out, rotate the image by a random angle, blur the image, etc.

There is about 1955 normal audio recordings. The audio dataset has been manipulated many times to get many data for training. For this dataset the data has been augmented in 2 types. Small silent audio recordings were taken and inserted between the normal audio to make it look like stuttering audio dataset. Since all the normal audios were manipulated, there will 1955 audio datas for stuttering. For dysarthria, the audio volume is reduced and one silent audio is insert fro each normal audio. 1955 normal audio data manipulated to get 1955 dysarthria audio dataset. So in total there is 1955x3 i.e. 5865 audio data for the model.

Feature Extraction

Feature extraction is a type of dimensionality reduction where a large number of pixels of the image are efficiently represented in such a way that interesting parts of the image are captured effectively. Mel-frequency cepstral coefficients(MFCC) is used for extracting features in this project. The MFCC feature extraction technique basically includes windowing the signal, applying the DFT, taking the log of the magnitude, and then warping the frequencies on a Mel scale, followed by applying the inverse DCT.

Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip. The mel frequency cepstral coefficients (MFCCs) of a signal are a small set of features (usually about 10-20) which concisely describe the overall shape of a spectral envelope.

MFCC alone can be used as the feature for speech recognition. The recorded speech signals are sampled and stored using Audacity. The sampling is done at a rate of 16000 samples per second. Each speech signal is divided into windows of 16 ms each and hence, 256 samples each. Feature extraction involves reducing the number of resources required to describe a large set of data. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

Building a ML model

Machine learning consists of algorithms that can automate analytical model building. ML model iteratively learn from data, ML models facilitate computers to find hidden insights from Big Data without being explicitly programmed. To create a successful machine learning model, some steps need to be followed, gather and clean the data, visualize the data, train the algorithm, evaluate the result based on the requirements. A ML model has been trained to recognize certain types of patterns. Train a model over a set of data, providing it an algorithm

that it can use to reason over and learn from those data. The three stages to build the hypotheses in machine learning are model building, model testing and applying model.

To build a CNN net, it necessary to build a network topology first. Create the model, then finalize the number of layers the model will have.

```
def build_model(input_shape):

    model = keras.Sequential()

    model.add(keras.layers.Conv2D(128,(3,3),activation='relu',input_shape=input_
                                shape))

    model.add(keras.layers.MaxPooling2D((3, 3), strides=(2, 2), padding='same'))

    model.add(keras.layers.BatchNormalization())
```

Here the proposed model has 3 conv layers. Each layer will have maxpooling and BatchNormalization done. Since the model will give 2-D array, and 1-D array is what is required, flattening is done. The 1-D array is then passed on to the dense layer to get the final model ready.

```
    model.add(keras.layers.Flatten())

    model.add(keras.layers.Dense(64, activation='relu'))

    model.add(keras.layers.Dropout(0.3))

    model.add(keras.layers.Dense(26, activation='softmax'))

    return model
```

In the proposed model, 70% of the data is used as the training dataset and in which 22% records are Dysarthria, 22% records are Normal and 30% records are Stuttering and 22% of the data is used as the testing dataset and in which 10% records are Dysarthria, 10% records are Normal and 10% records are Stuttering.

Model Fitting with CNN

Before training a model, there is a need to configure the learning process, which is done via the compile method. It receives three arguments:

```
optimiser = keras.optimizers.Adam(learning_rate=0.0001)  
model.compile(optimizer=optimiser, loss='sparse_categorical_crossentropy',  
metrics=['accuracy'])
```

The model is then compiled with the compile function. This creates the neural network model by specifying the details of the learning process. The model hasn't been trained yet. Only the optimizer to use and the loss function to minimize are declared. Actually model training is done using the model fit function. The arguments are as follows:

x: The input data, we defined it as X above. It contains the x and y coordinates of the input points.

y: Not to be confused with the y coordinate of the input points. In all ML tutorials y refers to the labels, in our case the class we're trying to predict: 0 or 1.

verbose: Prints out the loss and accuracy, set it to 1 to see the output.

epochs: Number of times to go over the entire training data. When training models we pass through the training data not just once but multiple times.

```
history = model.fit(X_train, y_train, validation_data=(X_validation, y_validation),  
batch_size=32, epochs=30)
```

The output of the fit method is the loss and accuracy at every epoch. We then plot it using our custom function, and see that the loss goes down to almost 0 over time, and the accuracy goes up to almost 1. The model is then predicted with the predict function. The accuracy is calculated.

```
test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)  
print('\nTest accuracy:', test_acc)
```

Website Development

The main objective of the website is to bring the user an interactive application. The website was developed using HTML and Python . The welcome screen prompts the user to select the preferred language, only Tamil so far, which then proceeds to the recording session. The page has a record button, pause button, and a stop button.

When the website is loaded you need to click the record button and start reading the tamil sentence. Once it is done, the recording will display in the page, with save to disk and upload buttons. If recording will get saved in the downloads folder in your personnel computer if the save to disk button is clicked. If the upload button is clicked then the recording is stored in a folder inside the running program, in the local disk. Once the recording is uploaded and ready for prediction, click the predict button in the page.

The predict button when pressed, will call a function. Then the audio is passed to the mfcc function and converted to .json file. The model is run once the predict is pressed. The website finally after loading, the file stored in the local disk is passed to the model and it is predicted. In the model if the output is 0, 1 and 2 in the website it is displayed as dysarthria, normal and stuttering respectively. This website was developed to be user friendly.

CHAPTER 5

RESULTS AND DISCUSSIONS

The website is a speech disorder recognition application written in HTML. It uses the persons voice recording to identify whether they have a disorder. Depending on the user's recorded voice it predicts whether normal, stuttering and dysarthria. Therefore, the disorder is predicted from the model.

Accuracy

Accuracy is a metric for evaluating classification models. Equation 1 refers to the formula to find accuracy. Mathematically, accuracy may be defined as the ratio of the number of correct predictions to the total number of predictions. The model yield a quite fair percentage (95%) of accuracy which reveals that most of the predictions for the type of disorder are correctly shown out.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where ,

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

$$\text{Confusion Matrix} = \begin{bmatrix} 590 & 1 & 0 \\ 0 & 541 & 30 \\ 0 & 48 & 550 \end{bmatrix}$$

Table 5.1. Performance Metrics

Class	Precision	Recall	F1-Score
Dysarthria	0.99	1.0	0.99
Normal	0.94	0.91	0.93
Stuttering	0.91	0.94	0.99

Accuracy for our model is calculated to be 95%.

Epoch vs Accuracy

The Figure 5.1 shows the training accuracy and epoch of the CNN model. The accuracy of a model is usually determined after the model parameters are learned and fixed and no learning is taking place. Then the test samples are fed to the model and the number of mistakes the model makes are recorded, after comparison to the true targets. Then the percentage of misclassification is calculated. There is a gradual increase in the accuracy throughout the epochs.

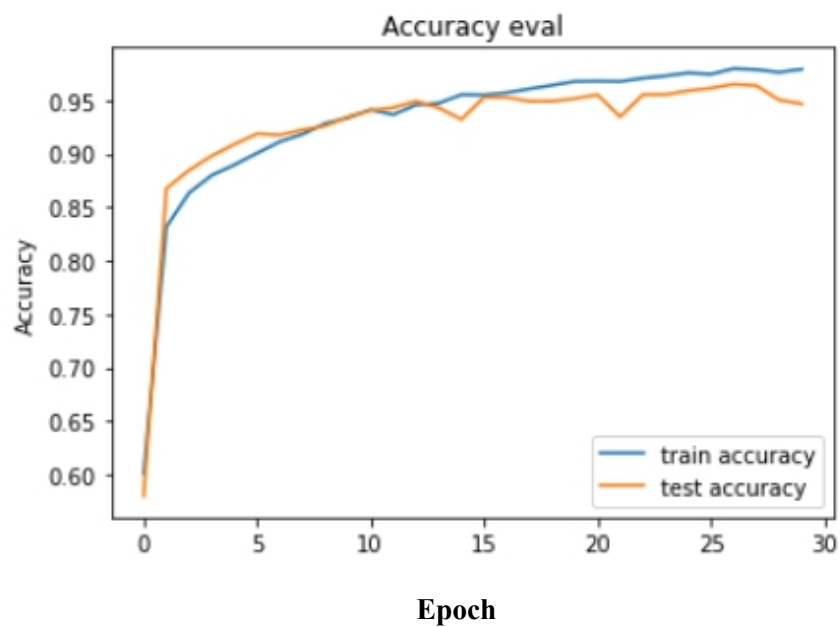


Figure 5.1 Epoch vs Accuracy

Epoch vs loss

Minimum loss gives a better model. The Figure 5.2 shows the training loss and epoch of the CNN model. The loss is calculated on training and validation and its interpretation is how well the model is doing for the two sets. Loss is a summation of the errors made for each example in training or validation sets. Loss value implies how well or poorly a certain model behaves after each iteration of optimization. Loss gets decreased as the epoch increases.

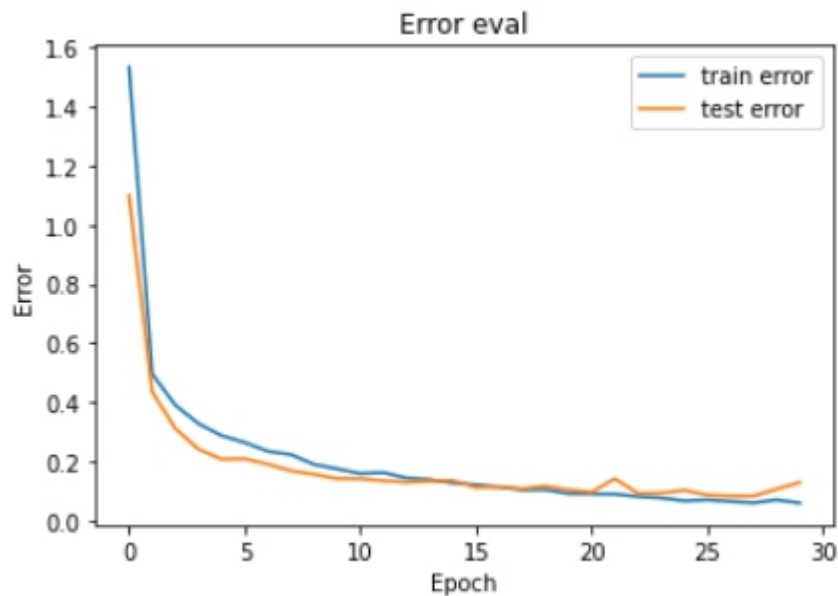


Figure 5.2. Epoch vs Error

Result

This website is able to predict disorder using simple recorded sentences depending on the user's speech by using a voice classification model under the hood. Figure 5.3 shows the GUI of the website. Once the website is opened, the user can record a sentence and immediately gets the prediction, about what their disorder is. Hence, by pressing the play button shown in the Figure 5.3 b user can hear their recorded voice again.

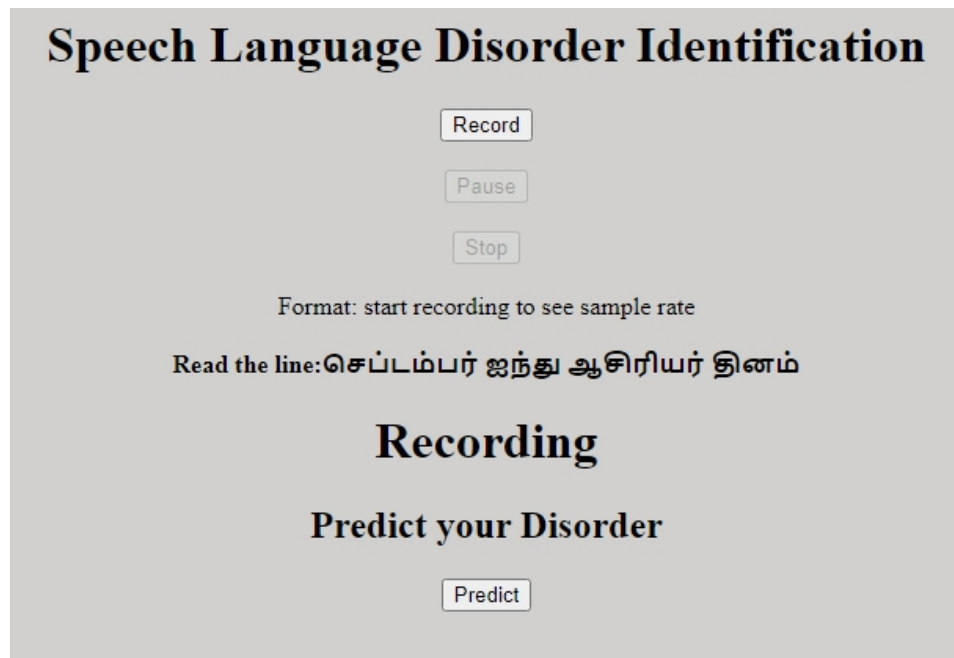


Fig 5.3. Website GUI - Audio Record

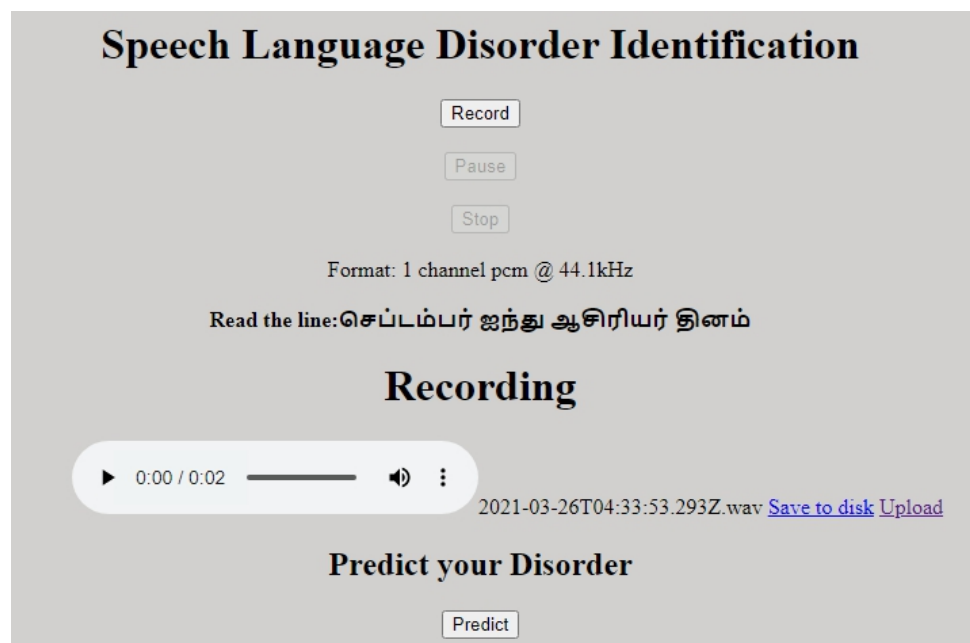


Fig 5.4. Website GUI - Audio Upload

Speech Language Disorder Identification

Record

Pause

Stop

Format: start recording to see sample rate

Read the line: செப்டம்பர் ஐந்து ஆசிரியர் தினம்

Recording

Predict your Disorder

Predict

Your speech is Normal

Fig 5.5. Website GUI - Prediction

Speech Language Disorder Identification

Record

Pause

Stop

Format: start recording to see sample rate

Read the line: செப்டம்பர் ஐந்து ஆசிரியர் தினம்

Recording

Predict your Disorder

Predict

You have stuttering

Fig 5.6. Website GUI - Prediction

Speech Language Disorder Identification

Record

Pause

Stop

Format: start recording to see sample rate

Read the line: செட்டம்பர் ஐந்து ஆசிரியர் தினம்

Recording

Predict your Disorder

Predict

You have Dysarthria

Fig 5.7. Website GUI - Prediction

CHAPTER 6

CONCLUSION AND FUTURE WORK

The web application was developed for the welfare of people. Machine Learning based Convolutinal Neural Networks is created. Dataset consists of recordings of people saying few sentences, the dataset consists of both normal and disordered data. The data is then augmented and dataset for stuttering and dyenthria. Validated/Test data is given finally to check the accuracy and loss of the constructed model. The accuracy is found out to be 95%. It will prompt the person in early identification of the disorder so that treating it as early as possible, with low cost.

The work will be further extended by developing with many more languages and many sentences. The next project will let the people to speak many small paragraphs, and small sentences. ML model will be developed and trained with audio dataset. People can record long paragraphs and the correctness can be predicted accurately. The application will also have different language options for the person to speak. This application will help people and will bring better improvement their life

APPENDIX 1

SAMPLE CODING

.ipynb FILE - Code for Model Building

```
import json

import os

import math

import pickle

import numpy as np

from sklearn.model_selection import train_test_split

import tensorflow.keras as keras

import tensorflow as tf

import structlog

import librosa

import matplotlib.pyplot as plt

import sounddevice as sd

from scipy.io.wavfile import write

DATASET_PATH = "Tamil_dataset"

JSON_PATH = "python_files1/data_new_2.json"

json_path1 = "python_files1/data_2_new_2.json"

SAMPLE_RATE = 16000

TRACK_DURATION = 1
```

```

SAMPLES_PER_TRACK = SAMPLE_RATE * TRACK_DURATION

DATA_PATH = "python_files1/data_new_2.json"

def predict_1(model, X):

    X = X[np.newaxis, ...]

    prediction = model.predict(X)

    predicted_index = np.argmax(prediction, axis=1)

    if predicted_index == 0:

        print(" Your are affecting Dysarthria ")

    if predicted_index == 1:

        print(" Your are voice is Normal Level ")

    if predicted_index == 2:

        print(" Your are voice Not_audiable")

    if predicted_index == 3:

        print(" Your are affectings stuttering ")

def save_mfcc(dataset_path, json_path, num_mfcc=13, n_fft=2048, hop_length=512,
num_segments=5):

    data = {

        "mapping": [],

        "labels": [],

        "mfcc": []

    }

    samples_per_segment = int(SAMPLES_PER_TRACK / num_segments)

    num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)

```

```

for i, (dirpath, dirnames, filenames) in enumerate(os.walk(dataset_path)):

    if dirpath is not dataset_path:

        semantic_label = dirpath.split("\\")[-1]

        data["mapping"].append(semantic_label)

        print("\nProcessing: {}".format(semantic_label))

        for f in filenames:

            file_path = os.path.join(dirpath, f)

            signal, sample_rate = librosa.load(file_path, sr=SAMPLE_RATE)

            if len(signal) >= SAMPLE_RATE:

                signals = signal

            else:

                signal = np.pad(

                    signal,

                    pad_width=(SAMPLE_RATE - len(signal), 0),

                    mode="constant",

                    constant_values=(0, 0),

                )

            for d in range(num_segments):

                start = samples_per_segment * d

                finish = start + samples_per_segment

                mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate,

n_mfcc=num_mfcc, n_fft=n_fft, hop_length=hop_length)

                mfcc = mfcc.T

```

```

        if len(mfcc) == num_mfcc_vectors_per_segment:

            data["mfcc"].append(mfcc.tolist())

            data["labels"].append(i-1)

            print("{} , segment: {}".format(file_path, d+1))

    with open(json_path, "w") as fp:

        json.dump(data, fp, indent=4)

def load_data(data_path):

    with open(data_path, "r") as fp:

        data = json.load(fp)

    X = np.array(data["mfcc"])

    y = np.array(data["labels"])

    return X, y

def plot_history(history):

    fig, axs = plt.subplots(1)

    axs.plot(history.history["accuracy"], label="train accuracy")

    axs.plot(history.history["val_accuracy"], label="test accuracy")

    axs.set_ylabel("Accuracy")

    axs.legend(loc="lower right")

    axs.set_title("Accuracy eval")

    plt.show()

    fig, axs = plt.subplots(1)

    axs.plot(history.history["loss"], label="train error")

    axs.plot(history.history["val_loss"], label="test error")

```



```

    axs.set_ylabel("Error")

    axs.set_xlabel("Epoch")

    axs.legend(loc="upper right")

    axs.set_title("Error eval")

    plt.show()

def prepare_datasets(test_size, validation_size):

    X, y = load_data(DATA_PATH)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size)

    X_train, X_validation, y_train, y_validation = train_test_split(X_train, y_train,
test_size=validation_size)

    X_train = X_train[..., np.newaxis]

    X_validation = X_validation[..., np.newaxis]

    X_test = X_test[..., np.newaxis]

    return X_train, X_validation, X_test, y_train, y_validation, y_test

def build_model(input_shape):

    model = keras.Sequential()

    model.add(keras.layers.Conv2D(128,(3,3),activation='relu',input_shape=input_shape))

    model.add(keras.layers.MaxPooling2D((3, 3), strides=(2, 2), padding='same'))

    model.add(keras.layers.BatchNormalization())

    model.add(keras.layers.Conv2D(32,(2,2),                                activation='relu',
input_shape=input_shape))

    model.add(keras.layers.MaxPooling2D((2, 2), strides=(2, 2), padding='same'))

    model.add(keras.layers.BatchNormalization())

```

```

    model.add(keras.layers.Flatten())

    model.add(keras.layers.Dense(64, activation='relu'))

    model.add(keras.layers.Dropout(0.3))

    model.add(keras.layers.Dense(26, activation='softmax'))

    return model

def predict(model, X, y):

    X = X[np.newaxis, ...]

    prediction = model.predict(X)

    predicted_index = np.argmax(prediction, axis=1)

    print("Target: {}, Predicted label: {}".format(y, predicted_index))

if __name__ == "__main__":

    save_mfcc(DATASET_PATH, JSON_PATH, num_segments=1)

    X_train, X_validation, X_test, y_train, y_validation, y_test = prepare_datasets(0.30,
0.20)

    input_shape = (X_train.shape[1], X_train.shape[2], 1)

    model = build_model(input_shape)

    optimiser = keras.optimizers.Adam(learning_rate=0.0001)

    model.compile(optimizer=optimiser, loss='sparse_categorical_crossentropy',

                    metrics=['accuracy'])

    history = model.fit(X_train, y_train, validation_data=(X_validation, y_validation),

                        batch_size=32, epochs=30)

    test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)

    print("\nTest accuracy:!", test_acc)

```

```

def test_data_analyze(file_path):

    data = {

        "mfcc": []

    }

    num_mfcc=13

    n_fft=2048

    hop_length=512

    num_segments=1

    c = 0

    k = 0

    samples_per_segment = int(SAMPLES_PER_TRACK / num_segments)

    num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)

    signal, sample_rate = librosa.load(file_path,sr=SAMPLE_RATE)

    if len(signal) >= SAMPLE_RATE:

        signals = signal

    else:

        signal = np.pad(

            signal,

            pad_width=(SAMPLE_RATE - len(signal), 0),

            mode="constant",

            constant_values=(0, 0),

        )

    start = samples_per_segment

```

```

    finish = start + samples_per_segment

    mfcc = librosa.feature.mfcc(signal[start:finish], sample_rate, n_mfcc=num_mfcc,
n_fft=n_fft, hop_length=hop_length)

    mfcc = mfcc.T

    if len(mfcc) == num_mfcc_vectors_per_segment:

        data["mfcc"].append(mfcc.tolist())

    with open(json_path1, "w") as fp:

        json.dump(data, fp, indent=4)

    with open(json_path1, "r") as fp:

        data = json.load(fp)

    X = np.array(data["mfcc"])

    X = X[..., np.newaxis]

    y = X[0]

    y = y[np.newaxis, ...]

    prediction = model.predict(y)

    predicted_index = np.argmax(prediction, axis=1)

    return predicted_index

def vvvv():

    l=0

    h=0

    dataset_path='test'

    for i, (dirpath, dirnames, filenames) in enumerate(os.walk(dataset_path)):

        for f in filenames:

```

```

file_path = os.path.join(dirpath, f)

z=test_data_analyze(file_path)

print(" target={} predicted_index= {} number={} ".format((h-1), z,h))

if z == h-1:

    #print(" target={} predicted_index= {} number={} ".format((h-1), z,h))

    l = l + 1

h += 1

print(" *****target={} ".format(l))

print("Test_data_accuracy={} ".format(l/3))

vvvv()

```

.html FILE - Code for Website Designing

```

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Simple Recorder.js demo with record, stop and pause - addpipe.com</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

</head>

<body bgcolor='#D1D0CE'><center>

<h1><center>Speech Language Disorder Identification</h1></center>

<div id="controls" align=center>

```

```

<button id="recordButton">Record</button><br><br>

<button id="pauseButton" disabled>Pause</button><br><br>

<button id="stopButton" disabled>Stop</button><br><br>

</div>

<div id="formats" align=center>Format: start recording to see sample rate</div>

<p><strong><center>Read the line:செப்டம்பர் ஐந்து ஆசிரியர்
தினம்</strong></p></center>

<p><strong><h1><center>Recording</h1></strong></p>

<center><ol id="recordingsList" align=center></ol></center>

<!-- inserting these scripts at the end to be able to use all the elements in the DOM -->

<center><script
src="https://cdn.rawgit.com/mattdiamond/Recorderjs/08e7abd9/dist/recorder.js"></center>
</script>

<center><script>

    URL = window.URL || window.webkitURL;

    var gumStream;

    var rec;

    var input;

    var AudioContext = window.AudioContext || window.webkitAudioContext;

    var audioContext //audio context to help us record

    var recordButton = document.getElementById("recordButton");

    var stopButton = document.getElementById("stopButton");

    var pauseButton = document.getElementById("pauseButton");

```

```

recordButton.addEventListener("click", startRecording);

stopButton.addEventListener("click", stopRecording);

pauseButton.addEventListener("click", pauseRecording);

function startRecording() {

    console.log("recordButton clicked");

    var constraints = { audio: true, video:false }

    recordButton.disabled = true;

    stopButton.disabled = false;

    pauseButton.disabled = false

    navigator.mediaDevices.getUserMedia(constraints).then(function(stream) {

        console.log("getUserMedia() success, stream created, initializing Recorder.js ...");

        audioContext = new AudioContext();

        document.getElementById("formats").innerHTML="Format: 1 channel pcm @@"
        "+audioContext.sampleRate/1000+"kHz"

        gumStream = stream;

        input = audioContext.createMediaStreamSource(stream);

        rec = new Recorder(input, {numChannels:1})

        rec.record()

        console.log("Recording started");

    }).catch(function(err) {

        recordButton.disabled = false;

        stopButton.disabled = true;

        pauseButton.disabled = true

```

```

    });
}

function pauseRecording() {
    console.log("pauseButton clicked rec.recording=",rec.recording );
    if (rec.recording){
        rec.stop();
        pauseButton.innerHTML="Resume";
    }else{
        rec.record()
        pauseButton.innerHTML="Pause";
    }
}

function stopRecording() {
    console.log("stopButton clicked");
    stopButton.disabled = true;
    recordButton.disabled = false;
    pauseButton.disabled = true;
    pauseButton.innerHTML="Pause";
    rec.stop();
    gumStream.getAudioTracks()[0].stop();
    rec.exportWAV(createDownloadLink);
}

function createDownloadLink(blob) {

```



```

var url = URL.createObjectURL(blob);

var au = document.createElement('audio');

var li = document.createElement('li');

var link = document.createElement('a');

var filename = new Date().toISOString();

au.controls = true;

au.src = url;

link.href = url;

link.download = filename+".wav"; //download forces the browser to donwload the file
using the filename

link.innerHTML = "Save to disk";

li.appendChild(au);

li.appendChild(document.createTextNode(filename+".wav "))

li.appendChild(link);

var upload = document.createElement('a');

upload.href="#";

upload.innerHTML = "Upload";

upload.addEventListener("click", function(event){

    var xhr=new XMLHttpRequest();

    xhr.onload=function(e) {

        if(this.readyState === 4) {

            console.log("Server returned: ",e.target.responseText);

        }
    }
});

```

```

    };

    var fd=new FormData();

    fd.append("audio_data",blob, filename);

    xhr.open("POST","/",true);

    xhr.send(fd);

  })

  li.appendChild(document.createTextNode (" "))

  li.appendChild(upload)

  recordingsList.appendChild(li); }

</script></center>

<div class="login" align=center >

    <h2>Predict your Disorder</h2>

    <form action="{{ url_for('predict')}}"method="post">

        <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>

    </form><br><br>

    {{ prediction_text }}

</div></center>

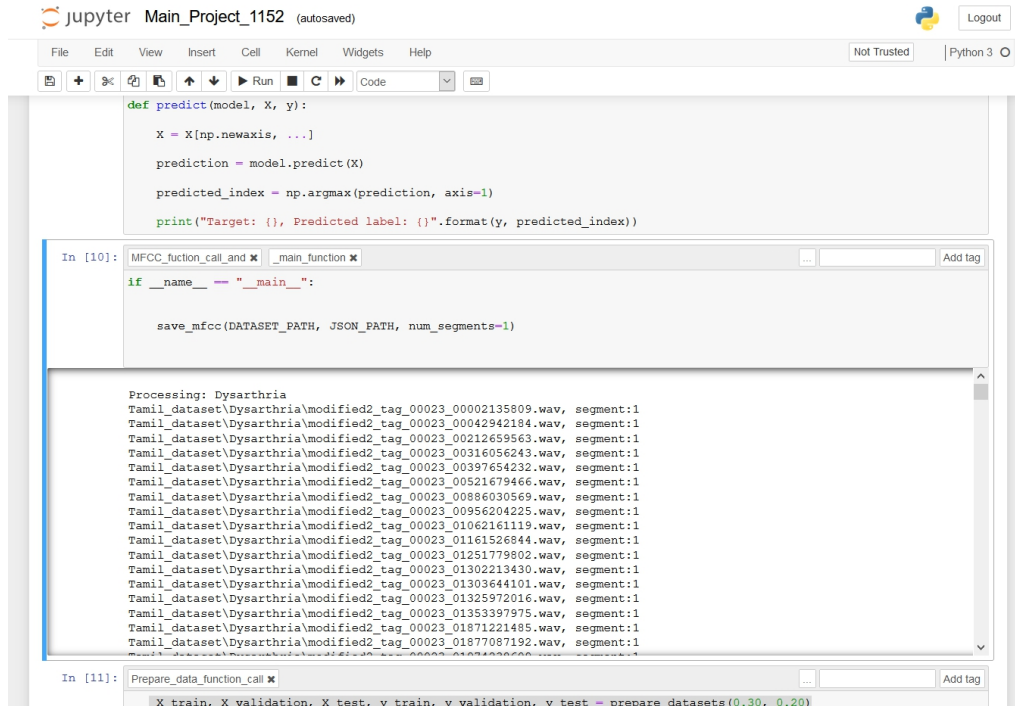
</body>

</html>

```

APPENDIX 2

SAMPLE SCREENSHOT



The screenshot shows a Jupyter Notebook interface for 'Main_Project_1152'. The code cell contains a `predict` function that takes a model and data `X` as input, predicts the class, and prints the target and predicted label. The output cell shows the execution of the `save_mfcc` function for the 'Dysarthria' dataset, listing 20 audio files with their paths and segment numbers. The next cell shows the execution of the `prepare_data_function_call` function, which prepares the datasets for training and validation.

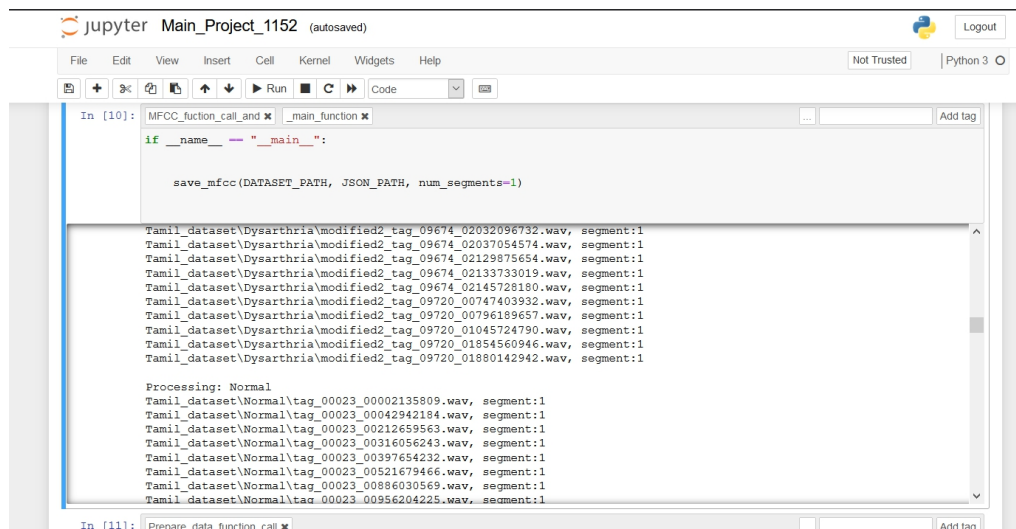
```
def predict(model, X, y):
    X = X[np.newaxis, ...]
    prediction = model.predict(X)
    predicted_index = np.argmax(prediction, axis=1)
    print("Target: {}, Predicted label: {}".format(y, predicted_index))
```

```
In [10]: MFCC_fuction_call_and _main_function
if __name__ == "__main__":
    save_mfcc(DATASET_PATH, JSON_PATH, num_segments=1)
```

```
Processing: Dysarthria
Tamil_dataset\Dysarthria\modified2_tag_00023_00002135809.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00042942184.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00212659563.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00316056243.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00397654232.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00521679466.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_0086030569.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_00956204225.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01062161119.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01161526844.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01251779802.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01302213430.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01303644101.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01325972016.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01353397975.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01871221485.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_00023_01877087192.wav, segment:1
```

```
In [11]: Prepare_data_function_call
X train, X validation, X test, v train, v validation, v test - prepare datasets(0.30, 0.20)
```

Figure A1. Output Screenshot

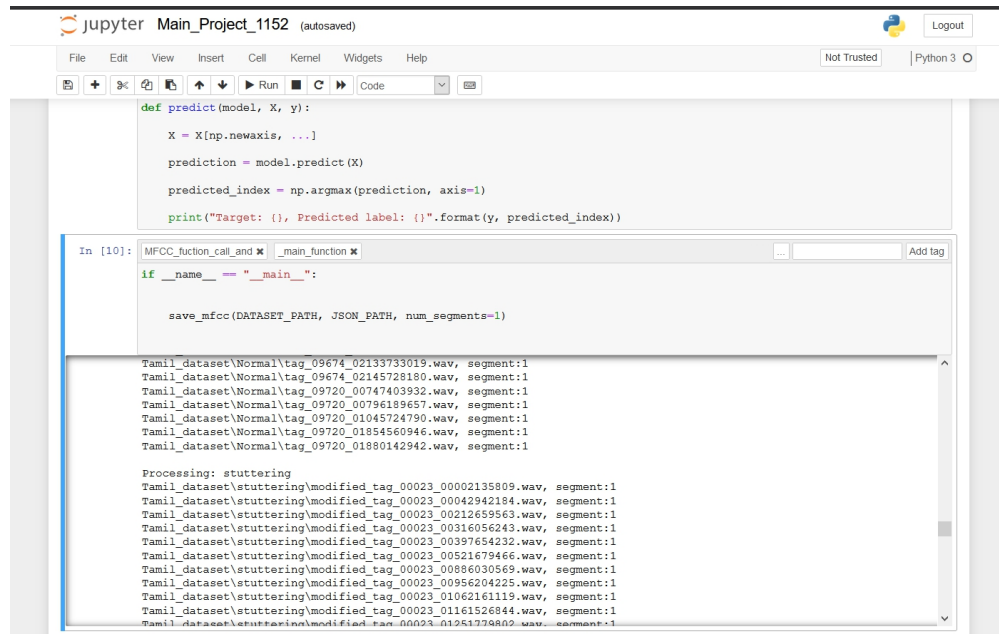


The screenshot shows the continuation of the Jupyter Notebook. The output of the `prepare_data_function_call` function is displayed, showing the processing of the 'Normal' dataset. It lists 10 audio files with their paths and segment numbers, followed by the processing of the 'Normal' dataset files.

```
Tamil_dataset\Dysarthria\modified2_tag_09674_02032096732.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09674_02037054574.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09674_02129875654.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09674_02133733019.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09674_02145728180.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09720_00747403932.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09720_00796189657.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09720_01045724790.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09720_01854560946.wav, segment:1
Tamil_dataset\Dysarthria\modified2_tag_09720_01880142942.wav, segment:1
```

```
Processing: Normal
Tamil_dataset\Normal\tag_00023_00002135809.wav, segment:1
Tamil_dataset\Normal\tag_00023_00042942184.wav, segment:1
Tamil_dataset\Normal\tag_00023_00212659563.wav, segment:1
Tamil_dataset\Normal\tag_00023_00316056243.wav, segment:1
Tamil_dataset\Normal\tag_00023_00397654232.wav, segment:1
Tamil_dataset\Normal\tag_00023_00521679466.wav, segment:1
Tamil_dataset\Normal\tag_00023_0086030569.wav, segment:1
Tamil_dataset\Normal\tag_00023_00956204225.wav, segment:1
```

Figure A2. Output Screenshot



The screenshot shows a Jupyter Notebook interface with the title 'Main_Project_1152 (autosaved)'. The top bar includes a 'Logout' button and a 'Python 3' indicator. The notebook contains a function definition for 'predict' and its execution output.

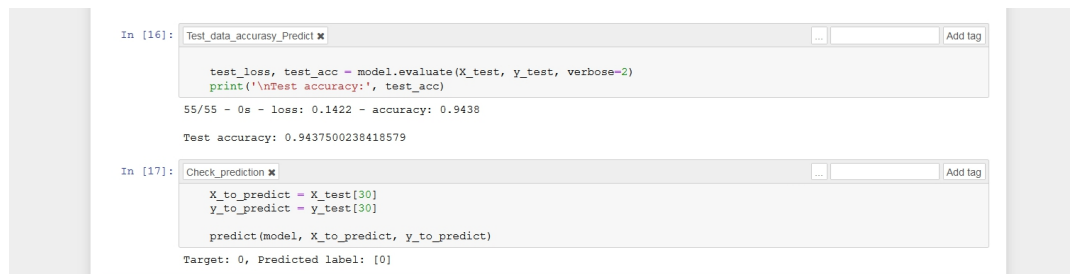
```
def predict(model, X, y):
    X = X[np.newaxis, ...]
    prediction = model.predict(X)
    predicted_index = np.argmax(prediction, axis=1)
    print("Target: {}, Predicted label: {}".format(y, predicted_index))
```

The output of the function is displayed in a scrollable box, showing a list of file paths and their corresponding predicted labels. The output is as follows:

```
Tamil_dataset\Normal\tag_09674_02133733019.wav, segment:1
Tamil_dataset\Normal\tag_09674_02145728180.wav, segment:1
Tamil_dataset\Normal\tag_09720_00747403932.wav, segment:1
Tamil_dataset\Normal\tag_09720_00796189657.wav, segment:1
Tamil_dataset\Normal\tag_09720_01045724790.wav, segment:1
Tamil_dataset\Normal\tag_09720_01894560946.wav, segment:1
Tamil_dataset\Normal\tag_09720_01880142942.wav, segment:1

Processing: stuttering
Tamil_dataset\stuttering\modified_tag_00023_00002135809.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00042942184.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00212659563.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00316056243.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00397654232.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00521679466.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00886030569.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_00956204225.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_01062161119.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_01161526844.wav, segment:1
Tamil_dataset\stuttering\modified_tag_00023_01541338802.wav, segment:1
```

Figure A3. Output Screenshot



The screenshot shows two code cells in a Jupyter Notebook. The first cell, labeled 'In [16]:', contains code to evaluate the model and print the test accuracy. The second cell, labeled 'In [17]:', contains code to make a prediction on a specific input and print the target and predicted labels.

```
In [16]: test_data_accuracy_Predict
test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)
print('\nTest accuracy:', test_acc)

55/55 - 0s - loss: 0.1422 - accuracy: 0.9438

Test accuracy: 0.9437500238418579
```

```
In [17]: Check_prediction
X_to_predict = X_test[30]
y_to_predict = y_test[30]

predict(model, X_to_predict, y_to_predict)

Target: 0, Predicted label: [0]
```

Figure A4. Output Screenshot

REFERENCES

- [1] Alan Preciado-Grijalva (2018) has developed a system capable of classifying the level of fluency of non-native English speakers using five different machine learning models.
- [2] Amandeep Kaur and Jubilee Padmanabhan [2017] have featured the need and importance of early identification of the students with specific learning disorder . This work focus on the various tools and techniques for the screening of SpLD.
- [3] Bastanfard [2010] have developed a software system which facilitates the interaction and synchronisation of language learning activities for Persian hearing-impaired children. The results show the successful integration of the designed system into the convenient method used by speech therapist in Iran.
- [4] David Millen (2017) has trained a machine by a machine learning technique for recognizing speech utterance to determine language fluency level of a user.
- [5] Janet M.Beilby (2012) has described about acceptance and commitment therapy for adults who stutter: Psychosocial adjustment and speech fluency.
- [6] Kanwajit Kaur and S. Pany [2017] have reviewed the computer based intervention which were used to improve social skills of autism spectrum disorder children. The analysis shows that computer based games are popularly used to improve the social skills.
- [7] Robles-Bykbaev et al(2014) have developed an ecosystem of intelligent ICT tools consists of an expert system to support speech and language pathologists, doctors, students, patients and their relatives.

[8] Sai Aishwarya Ramani and Amudhu Sankar [2017] have developed the prototype "ISpeak" for Augmentative and Alternative Communication (AAC) that increase social interaction, school performance and feeling of self esteem.

[9] Toki, E. I., & Pange(2010) introduced an e-learning system for improving articulation in Greek preschoolers. The results of the study on this software showed that children not only improved their articulation but they also increased on language activities success by acquiring new vocabulary.

[10] Witsawakiti et al (2006) has described an e-training tool designed to help hearing impaired children learn and practice words in Thai language more correctly. The tool uses speech to overcome the limitations of the traditional face-to-face speech therapy.