Identification of Autism Spectrum Disorder Markers in Spoken Conversations

Informationstechnik - Elektrotechnik - Mechatronik

Master Thesis

submitted by

Durgesh Nandan Sinha

Technische Hochschule Mittelhessen (THM), Friedberg

Thesis Supervisor: Prof. Dr.-Ing. Hartmut Weber
Co-supervisor: Dennis Pöpperl (M.Sc.)

Friedberg, 2022

Abstract

Autism spectrum disorder (ASD) is a disability that impacts the social behavior of a person. Diagnosis of ASD is a challenging task, as there is a spectrum of symptoms that can vary from person to person. One of the areas, which affect a person with autism is spoken conversation. This report focuses on recognizing autism markers in spoken conversation. Firstly the key symptoms with respect to spoken conversation will be discussed. To find a pattern in spoken conversation the audio needs to be digitized in form of text and with speaker identity. This report discusses various state-of-the-art machine learning models for speech-to-text translation or automatic speech recognition (ASR) and then the speaker diarization process. Autism detection videos can be very long as it's a long process so time complexity will also be determined for ASR and speaker diarization process. For pattern recognition, various metrics from the speech will be calculated. Lastly, a conclusion will be made and the future scope of this project will be discussed.

Acknowledgements

I would like to thank Professor Hartmut Weber for his guidance throughout my Master course at THM. I got opportunity to understand and work on many state pf the art technology such as computer vision and natural language processing.

Special thanks to Mr. Dennis Pöpperl for supervising this thesis. There were lot of challenges to complete this thesis but your guidance and support always helped me to overcome those challenges.

Statement of Authenticity
Selbstständigkeitserklärung

I hereby declare that I have completed this thesis work independently. I have not used sources or resources other than those specified in the references. I have not submitted this thesis to any other university or examination body in the same or similar form.

Friedberg, 31st March 2023

Durgesh Nandan Sinha

Contents

Abstract i

Acknowledgements ii

Statement of Authenticity Selbstständigkeitserklärung iii

Contents iv

List of Figures vi

List of Tables vii

Acronyms viii

1 Introduction 1

2 Theory 3
2.1 Related Work . . . 3
2.2 Autism Symptoms . . . 4
2.2.1 Natural Language Toolkit (NLTK) . . . 5
2.3 Natural Language Processing . . . 5
2.3.1 Brief History of NLP . . . 5
2.3.2 NLP Tasks . . . 6
Tokenization . . . 6
Stop Word Removal . . . 7
Stemming . . . 8
Part Of Speech (POS) . . . 8
Word Sense Disambiguation . . . 8
N-Grams . . . 9
2.4 NLP Approaches . . . 9
2.5 Naive Bayes Classifiers . . . 10

2.6 Automatic Speech Recognition . . . 11
2.6.1 Transformers . . . 11
2.7 Speaker Diarization . . . 12

3 State of the art **14**
3.1 Automatic Speech Recognition . . . 14
3.1.1 wav2letter++ . . . 15
3.1.2 wav2vec . . . 15
3.2 Speaker Diarization . . . 16
3.3 ChatGPT . . . 16

4 Implementation **18**
4.1 Tools . . . 18
4.1.1 Python . . . 18
4.1.2 PyCharm . . . 18
4.1.3 Google Colab . . . 18
4.2 Rule based approach . . . 19
4.2.1 Automatic Speech Recognition . . . 20
Example for ASR . . . 25
4.2.2 Speaker Diarization . . . 27
4.2.3 Change text to statements with speakers identifications 29
4.2.4 Identify question and predict type of question . . . 31
4.3 Machine learning approach . . . 32

5 Results **33**
5.1 ASR and Speaker Diarization . . . 33
5.2 Speech Metrics . . . 35

6 Conclusion **37**
6.1 Summary . . . 37
6.2 Future Scope . . . 38

A Naive Bayes classifier **39**

B Automatic Speech Recognition Classes **43**

Bibliography **44**

List of Figures

2.1 ELIZA Conversation 6
2.2 NLP Tasks 6
2.3 Transformer 12

3.1 Wav2vec model 15

4.1 CPU vs GPU 19
4.2 CPU vs GPU Time comparison 19
4.3 Pipeline blocks 21
4.4 ASR complete pipeline 21
4.5 Chunking with stride 25
4.6 ASR pipeline with Example 27
4.7 Speaker Diarization 28
4.8 Speech to Text 29

5.1 Graphical representation of ASR and SD processing time 34

List of Tables

5.1 Table to show time required by ASR and SD to process audio of different length. 34
5.2 Average sentence length for interviewer and respondent for different audio length 35
5.3 Average speech time for interviewer and respondent for different audio length 35
5.4 Average word time for interviewer and respondent for different audio length 36
5.5 Average thinking time of respondent for different audio length . . 36

A.1 Naive Bayes classification example 39
A.2 Probability of label|Weather 40
A.3 Probability of label|Striker 40
A.4 Probability of label|Defence 40
A.5 Probability of Goal 41

B.1 Classes to word representation mapping in automatic speech recognition decoder 43

Acronyms

ASD	Autism Spectrum Disorder
AI	Artificial Intelligence
NLP	Natural Language Processing
ML	Machine Learning
CNN	Convolutional Neural Network
fMRI	Functional Magnetic Resonance Imaging
DSM	Diagnostic and Statistical Manual
NLTK	Natural Language Toolkit
ASR	Automatic Speech Recognition
SD	Speaker Diarization
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
MFCC	Mel Frequency Cepstral Coefficients
SVM	Support Vector Machines
XLS-R	Cross-lingual Speech Representation
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
CPU	Central Processing Unit
TPU	Tensor Processing Unit
IDE	Integrated Development Environment
RAM	Random-Access Memory
CSV	Comma-Separated Values

Chapter 1

Introduction

Autism spectrum disorder (ASD) is a developmental disability or can be defined as a condition related to the development of the brain that impacts the way the person perceives the environment surrounding it [1]. The person who is suffering from autism is sometimes referred to as an "autistic person" or "person with autism" and has various social problems such as social communication, social interaction, repeated behaviors, etc. It is a spectrum disorder meaning this can have a wide range of symptoms and severity [2].

Diagnosis of autism spectrum disorder is a challenging task as there are much of psychological behaviors involved, unlike other diseases where medical tests determine the disease. People with autism can lead a normal life if they are diagnosed and can be trained to learn in a different way.

In the era of artificial intelligence (AI), everything can be done with the help of computers. AI has bright scope, as per the beliefs of a group of "The Beneficial AI Movement" [3] it has the potential to treat a wide range of medical conditions and help to live a better quality of life. As an example, Neuralink [4] a company that works on neural engineering is researching on treating a wide range of neurological disorders such as restoring sensory and motor function, the way humans perceive the world, and social interaction [4].

Natural language processing (NLP) is the branch of AI that primarily deals with human interaction with machines. It can be better stated as "NLP is a subset of AI". As stated above, one of the important aspects of autism is that autistic people are that they often face difficulties in social interaction. Ideally, psychiatrists are needed to conduct interviews and find the pattern in behavior to diagnose autism. But as technology is evolving, new-age technologies like NLP can help to find the same pattern using interviews with autistic person.

Applications of NLP are now everywhere from chatbots that help to book flight tickets to waking up in the morning and checking schedules with "alexa" [5] or "siri" [6]. This technology has the potential to diagnose certain mental disorders. For example, there are many chatbots [7] available for diagnosis of depression - a mental disorder that changes the way a person feels about the world.

A recent development in the field of NLP is ChatGPT [8], It is a chatbot that can automate a lot of tasks such as summarizing text, answering questions, providing ideas for business, writing stories, and many more. It works on a transformer model to learn from text and generate new text. It is a revolutionary change as it can generate new things which people have never thought of such as writing new stories. Research can benefit a lot from this as this can give precise answers to many research in an understandable way.

This paper aims to discuss the possibilities of diagnosis of autism spectrum disorder using natural language processing (NLP) and machine learning (ML). NLP and ML both are subsets of AI and when these technologies come together they can open paths for many new applications. These technologies can be effective tools in diagnosing the exact symptoms and severity of autism. Several symptoms will be discussed in further sections which involve the role of language development and social behavior.

Chapter 2

Theory

Autism is a neurological disorder, it affects a person's ability to interact with others. It is important to detect autism in the early stage so that education patterns and social learning can be optimized as per the need of the person [9]. There have been several research going on to diagnose autism using artificial intelligence technology.

2.1 Related Work

A lot of prior research has been made to automate the process of autism diagnosis using AI technologies. This literature survey has been conducted to explore current trends in research on ASD diagnosis. Recently, Ibrahim A. A. et al. presented a paper in Eye tracking based autism diagnosis using machine learning and deep learning models [10]. Mahmoud Elbattah et al. have developed NLP based approach for ASD using saccadic eye movement [11], the team focuses on early diagnosis of ASD using saccadic eye movement. The eye movement data is translated into textual strings and NLP-based methods are applied to produce features for training classification models. Both of the works use eye movement as the basis of their algorithm.

Another study conducted by Zeinab Sherkatghanad et al. on automatically detecting autism using a convolutional neural network (CNN) [12] uses functional magnetic resonance imaging (fMRI) data and tries to find out patterns to classify as autism. Another research conducted by Mohammed Isam Al-Hiyali et al. uses the same fMRI signals to identify autism based on CNN [13]. Above work is based entirely on fMRI signals from the brain to detect autism.

This paper aims to make use of these technologies, but instead of eye movement or fMRI data, it will use the text spoken by an autistic person during the interview to find patterns in autism. Similar work is done by Clark et al. on diagnosing autism using NLP [14]. Clark et. al. has filed patent for work in finding one or more symptoms of autism using conversation. His paper focus on relating conversation meaning one of the symptoms of autism is that person with autism tends to provide an inappropriate response. This paper on the other hand focus on converting audio to text and then using statistics to find a pattern in a person with autism.

2.2 Autism Symptoms

Considering only linguistic features as the parameter of autism, currently autism can be diagnosed in 3 ways firstly, medical diagnosis where a psychologist perform assessment based on symptoms which is in accordance with Diagnostic and Statistical Manual (DSM) [15]. Another way to diagnose is by by educational determination where evaluation results are used as a measure to determine autism. Autism can also be diagnosed by screening test which is done at fixed interval after birth [16]. These ways can help to determine autism in a child. This paper mostly focus on development of social interaction therefore below is the list of symptoms which can use social communication parameters and help to detect autism.

- Children with autism often confuse WH questions. They often respond to a given WH question as though a different question were asked. For example, a child may answer a, "what" question when a, "where" question is asked. [17]
- Repeating the same phrases. [18]
- Taking things very literally – for example, they may not understand phrases like "break a leg". [18]
- Finding it hard to say how they feel. [18]
- Children with ASD may have difficulty developing language skills and understanding what others say to them. [19]
- Other children may use stock phrases to start a conversation. [19]

These symptoms forms the basic of this research.

2.2.1 Natural Language Toolkit (NLTK)

Natural Language Toolkit is an open source python library that has predefined methods for NLP tasks. It has a collection of over 100 Corpora and lexical resources which comes in handy during NLP operations [20]. The dataset such as punkt, and stopwords. wordnet has been extensively used during the experiment related to this paper.

2.3 Natural Language Processing

Natural language processing is a subsection of artificial intelligence that deals with interactions between humans and machines using natural language. Suppose two people are talking using a common language referred to as their natural language, then their brains will process this language and extract information to understand each other. But computers, are not capable of understanding this language. To make computer understands natural language a technology named natural language processing can be used. It helps the computer to process the information in the same way a human can process. In brief, NLP enables computers to derive meaning from natural language.

2.3.1 Brief History of NLP

Researchers at MIT in 1964 were trying to make a system that can talk like humans. Joseph Weizenbaum a German-American scientist made the first program ELIZA [21] which can talk like humans it was based on pattern matching and substitution methodology. A set of scripts were given as directives that ELIZA can process and provides pre-stored results. This was able to give an illusion of a natural language processing system [21]. ELIZA is known as the first attempt of a program to mimic human understanding of language. Below figure 2.1 represents the implementation of the very first ELIZA program.

```
  Welcome to
                 EEEEEE  LL      IIII   ZZZZZZ   AAAAA
                 EE      LL       II       ZZ   AA   AA
                 EEEEE   LL       II     ZZZ    AAAAAAA
                 EE      LL       II    ZZ      AA   AA
                 EEEEEE  LLLLLL  IIII ZZZZZZ    AA   AA

  Eliza is a mock Rogerian psychotherapist.
  The original program was described by Joseph Weizenbaum in 1966.
  This implementation by Norbert Landsteiner 2005.


ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:   Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU:   He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:   It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:
```

Figure 2.1: ELIZA Conversation [21], which is known as the first program to mimic human understandings.

2.3.2 NLP Tasks

Humans communicate in natural language in many ways. This language is made up of things such as sarcasm, idioms, phrases, metaphors, etc. To make computer understands natural language there are standard set of task that needs to be done. Each task adds its own advantages to make computer understand the language in a better way. There are various tasks that are performed during NLP, these are Tokenization, Stop Word Removal, Stemming, and more. This is also represented in figure 2.2 The following are the tasks:

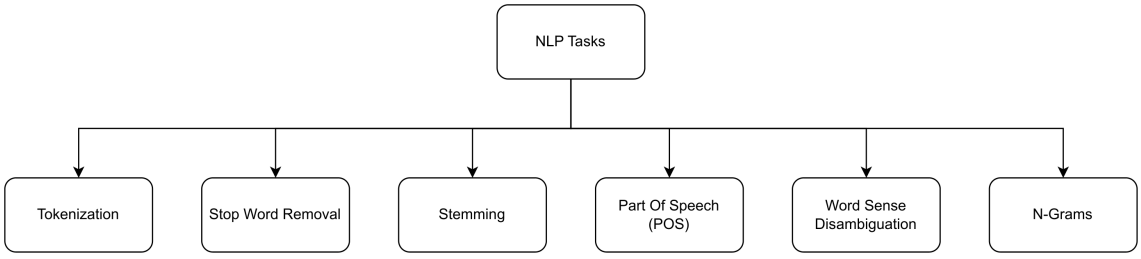

Figure 2.2: Various NLP Task

Tokenization

Tokenization is the process in which a given sentence is broken down into words and sentences called tokens [22]. It can be done in two ways firstly it

can be broken down into multiple sentences and secondly, sentences can be broken down into words. In programming, this can be achieved by splitting the sentences using a split method based on white space characters. In this paper, nltk which is a python library will be used to achieve this work. This library has predefined functions to split into sentences or into words. The library exposes "sent_tokenize" and "word_tokenize" methods which are used for breaking down into sentences and words respectively.

Input : "This is an example sentence. This is another example sentence."
Output of "sent_tokenize" : ['This is an example sentence.', 'This is another example sentence.']
Output of "word_tokenize" : ['This', 'is', 'an', 'example', 'sentence', '.', 'This', 'is', 'another', 'example', 'sentence', '.']

As in the above example it can be seen that the input sentence is just simply converted into array of strings based on the operations. The first output translates the sentence into two sentences. and the next output translated into array of words. Interestingly, the punctuation mark are treated as individual tokens so the period is part of sentences and as a separate tokens.

Stop Word Removal

Once the tokens are created, the tokens contain unnecessary words such as punctuation marks, articles (a, an, and the), and helping verbs like is, am, are, etc. These words are called stop words [22] and do not add any information to the sentences, hence removing these words does not lose any information from the text. These stop words are from a list called stop list [23]. This list contains stop words from all the popular languages and is used as a reference for removing stop words. The nltk library contains stop words defined in nltk corpus, this set has all the stop words defined in different languages. Additionally to remove punctuation one can make use of string library of python which contains the entire list of punctuations. Now, the stop word removal task can be done by filtering out stop words from the array of words.

Input is output of "word_tokenize" : ['This', 'is', 'an', 'example', 'sentence', '.', 'This', 'is', 'another', 'example', 'sentence', '.']

Output of stop word removal function : ['This', 'example', 'sentence', 'This', 'another', 'example', 'sentence']

From the above example, It can be noted that words like is, an and period is removed from the output.

Stemming

Sometimes a sentence contains similar words but with different morphological forms. For example, words like open, opening, and opened. If they are tokenized they can be treated as different words but all three words have the same meaning so they must be represented as a single entity for further calculations. To achieve this goal, these words must be reduced in their root form which can then be used as a single word. This process to reduce the word to its root form is called stemming. Using nltk this can be achieved using a built-in method "stem". The most common algorithm used for stemming in English is the Porter algorithm [24] developed in the year 1980.

Part Of Speech (POS)

In a sentence, sometimes a word has a different meaning based on the position of the word in a sentence. Therefore, to carry out analysis it becomes important to know what is the "part of speech" each word signifies. It can be a noun, pronoun, verb, adverb and so on. the nltk function which can identify the parts of speech is "pos_tag".

Input is output of "word_tokenize" : ['This', 'is', 'an', 'example', 'sentence', '.']

Output of part of speech function : [('This', 'DT'), ('is', 'VBZ'), ('an', 'DT'), ('example', 'NN'), ('sentence', 'NN')]

In the above output example it can be seen that the different parts of sentence is represent in acronym. For example: NN is "Common Noun", DT is "Determiners" and so on. The complete list of acronym can be found in google website [25]

Word Sense Disambiguation

Disambiguation in simple English means the way of interpreting the intended use of a word by the author. A word can have multiple meanings when used with different words. Word Sense Disambiguation is the process of getting the

meaning of a word based on the context in which it occurs.

For example, the word "racket" has two meaning one is the racket used to play games such as badminton or tennis, and another is the group of people who run illegal or dishonest businesses to earn money. Now if the word racket is used in sentences then to understand what the word signifies, Word Sense Disambiguation process can help.

Python library nltk has a lexicon called Wordnet. Lexicon can also be understood as a thesaurus. It has words, with their meaning relative to the relation in sentences. Each word with meaning is called "Synset". So to figure out which word has what meaning in the sentence Word Sense Disambiguation is carried out using the function "lesk". It takes two parameters one is the sentence and another the word whose meaning needs to be found.

N-Grams

Language is a well-developed tool and there are certain rules by which sentences are formed. For example, in a sentence "he is so hard working that" the next word can easily be guessed. The probability of the next word being "he" has more probability than any word [26]. This is the basic concept of N-Grams. It keeps a track of previous words and this helps to predict the next word. The term N in N-Grams refers to the number of previous words it considers to predict the next word. If N is 2 then it considers the previous 2 words to predict the next word. similarly, it can be 3-Grams, 4-Grams, and so on.

2.4 NLP Approaches

Solution of any problem in NLP can be found using two approaches. The first method is rule based approach. In rule based approach the rule is designed based on the historical data and patterns and based on these rule NLP algorithm is designed.

Another way is machine leaning approach, when the pattern is dynamic and its hard to setup rules then machine learning approach is used. The algorithm will look for the historical data and continuously train itself and also keep on updating the rules.

2.5 Naive Bayes Classifiers

One of the key features of this project is to identify the question and answer in a speech. To know whether the given statement is a question or not, and if it is a question whether it is a "WH" question or a "Yes/No" question can be known with the help of a classifier. Classifier is a machine learning program that labels statements, based on the probability of their features.

Naive Bayes classifier is one of the classifiers used to label text. This algorithm works in two parts, first, the Bayes' theorem to express P(label|features) in terms of P(label) and P(features|label) [27]. Another part is naive assumption is to assume all the features are independent. This can be expressed mathematically by equations 2.1 and 2.2 respectively.

$$P(label|features) = \frac{P(label) * P(features|label)}{P(features)} \tag{2.1}$$

$$P(label|features) = \frac{P(label) * P(f1|label) * ... * P(fn|label)}{P(features)} \tag{2.2}$$

This formula can be described as the probability of a label given features is the ratio of the product of the probability of a label and the probability of a feature given label to the probability of features. This is the case of the probability of dependent events. where the probability of the label is dependent on the feature. It is also important to note that the numerator portion is only calculated to know the probability of the label because the denominator part can be considered constant and will be later discarded from the calculation to determine probability.

To better understand the conditional probability concept, let's assume there is a bag of 5 marbles, 3 of them are red in color and 2 are blue in color. So the probability of picking a red marble from the bag is 3/5, but after this event, if the probability of picking up a blue marble is calculated the probability will be 2/4 or 1/2 because there are only 4 marbles left in the bag and 2 are blue marbles. This scenario is known as a dependent event and the probability of drawing blue marble after drawing red marble is known as conditional probability. This forms the basis of Bayes's theorem.

Naive Bayes theorem is explained with a generic example in appendix A. The same concept can be later applied to text classification problems.

This theorem is applied to statements to classify or label them based on the probability of their features. Here features are words or tokens which occur in

various statements. This approach comes under the supervised learning model of machine learning algorithm.

2.6 Automatic Speech Recognition

Automatic Speech Recognition (ASR) is also known as speech-to-text (STT). In this process, audio signals are transcribed into text. A speech-to-text system can be explained as a block of several components, a spoken input which is a raw waveform, a signal processing module to digitize the waveform, a feature extraction module which extracts relevant features from the digital signal, and a natural language processing module to convert the features into text. The system can be made better by using a language model to improve the accuracy of the transcription by taking into account the context and grammar of the spoken input.

The early work dates back in the year 1952 when the machine named "Audrey" by Bell Labs in the US was invented to recognize numbers between 1 to 9 [28]. Recent development in this field is done with the help of machine learning techniques where machine learning models are used to recognize words and then bring out a sentence. One of the popular model, which is used for speech-to-text translation is developed by Facebook is called "wav2vec2-xls-r" and will be discussed in detail in further sections 3.1.

2.6.1 Transformers

Transformers can be explained in short as the alternative of Recurrent neural networks (RNN) and long short-term memory (LSTM) which solves the problem of sequential computation by doing parallel computation of the attention mechanism. RNN and LSTM were using a sequential approach where each encoder output is fed into the next encoder and finally, the last encoder will be used to feed into the decoder. To translate a long sentence, this model fails as it loses the position of words for long sentences. Thus, transformers are introduced. The architecture of transformers is based on attention, which means the model pays attention to the words as well as to their sequence. This is achieved by using serial encoders which take the input and this input is then fed to parallel decoders to get the translation [29].

As an example, a sentence in English has to be translated into German. In the conventional technique i.e. by using RNN and LSTM each word of English will be translated into German and then the output will be clubbed to get a

sentence. It can work for small sentences as these models have a memory to retain a few words. But in the case of large sentences these models will not work as by the time the last word is processed the first word will be lost by the model hence it can not efficiently translate. This problem can be solved by using transformers.

Below figure 2.3 shows the implementation of transforms it is mostly used in translating from one language to another. As an example, the encoder inputs a sentence "I like science" and encodes it into numeric representations and then the numbers are decoded into the desired language (German is this example). The decoders work in parallel to arrange the words to make a meaningful full sentence.

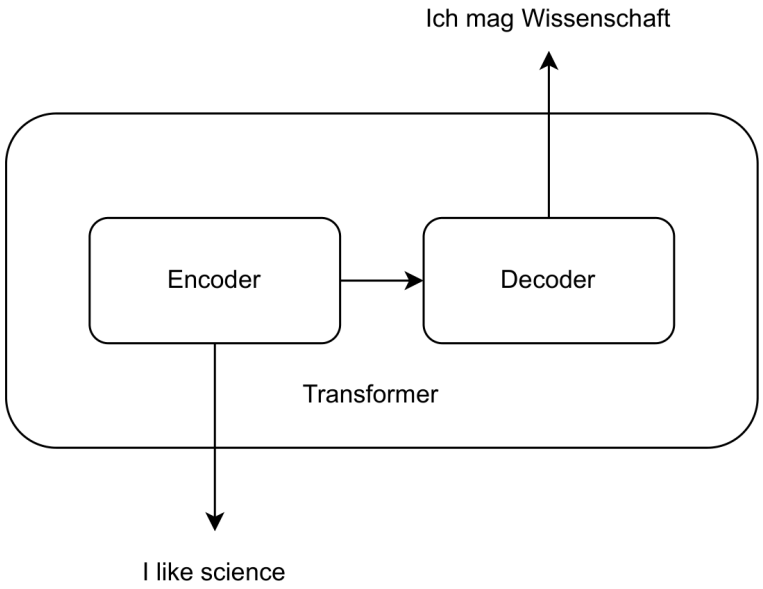

Figure 2.3: Transformer block with example

2.7 Speaker Diarization

"Diarize" means entering an event in the diary. Speaker diarization does the same thing, it creates an entry in the memory for *who spoke when*. In multi-speaker audio, it becomes very important to recognize speakers by their voices. In such a situation, speaker diarization is used. The goal of this process is to identify and label audio as per speaker and their timestamps.

Speaker diarization is a complex process. It involves dealing with various scenarios such as different speech patterns among speakers, background noise,

and overlapping speech. The most common approach for speaker diarization is with the help of machine learning models.

Some of the most common models used in speaker diarization include:

1. Clustering based models: These models operate on unsupervised clustering algorithm, for example K-Means [30]. K-Means refers to number of clustered needs to be formed. For example, If K = 2, this means 2 clustered will be formed.

   It groups segments of the audio according to the identity of the speaker. These algorithms uses voice features and represent them using Mel Frequency Cepstral Coefficients (MFCC) for segmentation [31].

   MFCC are represented as sequence of vectors for audio input signal. It is widely used in speaker diarization process. Its vectors represent some import speech information such as vocal tract and speech sounds.

2. Support Vector Machines (SVMs): SVMs are used in supervised learning models. This model works on prediction of speaker label for each speech segment [32].

Chapter 3

State of the art

Natural language processing is been a hot topic in the industry since last many years. Many researchers are trying to build state-of-the-art solutions to achieve minimal error and maximum output for a particular problem. This chapter will discuss state-of-the-art technology used during this project.

3.1 Automatic Speech Recognition

Some of the latest research in the field of automatic speech recognition is about creating a model that is capable of learning from one language and able to provide output with the same accuracy for another language. One of the recent models that is developed in 2021 is "wav2vec2-xls-r" [33]. The meaning of this model lies in its naming convention, the model name can be separated into two terms "wav2vec2" and "xls-r". The term term "wav2vec2" means "wave to vector" whose literal meaning is waveform to vector or waveform to digital representation. Another term xls-r stands for Cross-lingual Speech Representation meaning it is used for different languages. This model was ideally developed to solve the problem of translating low-resource languages (languages for which it is difficult to obtain training data) such as Sinhala, and Zulu using high-resource languages such as English or German. It uses unlabeled data and self-supervised to learn from the data. To understand the theory behind this model there is a need to understand the chronological order of development of these models. The first which can be discussed in this flow is "wav2letter++" [34] which was published in the year 2018. In the next year, 2019 some improvisation was made to this model, and a new model "wav2vec" [35] was made which forms the basics of "wav2vec2" model. In the following sections "wav2letter++" and "wav2vec" will be discussed.

3.1.1 wav2letter++

In the year 2018 wav2letter++ model was published to recognise the stream of letters from audio signals. wav2letter++ acts as the basic model for speech translation models such as wav2vec. wav2letter++ which is written in C++ to classify raw audio input to a sequence of letters [34]. The principle of this model is based on CNN. It does so by extracting audio representations from the raw audio (using CNN layers), processing the sequence of audio representations with a stack of transformer layers, and classifying the processed audio representations into a sequence of output letters.

3.1.2 wav2vec

Wav2vec translates into literal meaning of waveform to vector. It translates waveform, a audio signal to vectors. Vectors can be explained as digital representation of audio signals. The input of this model is raw audio signal waveform and the output is the words representations. It is based on Convolutional Neural networks (CNN). The model can be separated into two networks.

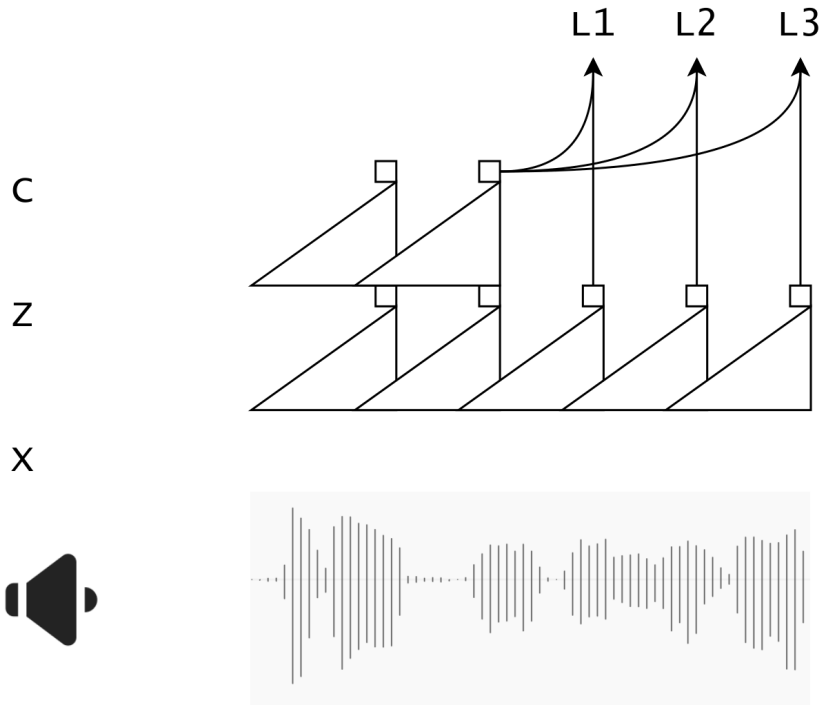

Figure 3.1: Illustration of Wav2vec model

In the above illustration 3.1 it can be seen as the model has two layers of the network. The first network is the "encoder network" which embeds the

audio signal in a latent space and the other network is the "context network" which combines multiple time steps of the encoder to obtain contextualized representations [35].

Encoder network: Latent space is also known as embedding space or feature space meaning it has representations based on the features for the purpose of finding [35]. Raw audio samples xi X is fed into five layer convolutional network which is also the encoder network f : X → Z. The encoder layers have kernel sizes (10, 8, 4, 4, 4). The output of the encoder is a low sampling feature representation zi Z, each zi represents encodes of about 30 ms of 16 kHz of audio.

Context network: Next, the context network g : Z → C is a nine layer convolutional network. It inputs the output of the encoder network to mix multiple latent representations z1, z2, z3. . . . zi into a single contextualized tensor ci = g(z1 . . . zi) for a receptive field. The low sampling representations are now considered together to provide a sound and it also predicts further sounds which is called contextualised representations.This representation is then used to classify words and get the speech representations [35].

3.2 Speaker Diarization

Speaker diarization is the task of segmenting audio, based on the identity of the speakers. There are many companies working on building models for speaker diarization such as AssemblyAI [36] and Hugging Face. Since Hugging Face is open source so it is used for this project. Hugging Face provides the pre-trained model for speaker diarization process "pyannote/speaker-diarization" [37]. This library follows a clustering based algorithm where the voice of speaker is clustered together based on physical properties of a voice such as pitch and tone. There is also an option to set the maximum number of speakers to obtain better results. In case the maximum number of speaker is unknown then this field can be left blank and depending upon the clusters it will show the results.

3.3 ChatGPT

ChatGPT is a chatbot [8] developed by OpenAI organization and launched in November 2022. It is funded by Microsoft. ChatGPT is trained on Microsoft Azure supercomputer. The supercomputer used by OpenAI has more than 285,000 CPU cores, 10,000 GPUs, and 400 Gbps of network connectivity for

each GPU server [38]. The interaction with ChatGPT happens in a conversational way. The term ChatGPT can be broken down into two words Chat and GPT. Chat translates to its way of interaction for humans and another term GPT (Generative Pre-trained Transformer), which is the generative language model that works on transformer architecture discussed in section **??**. ChatGPT was initially launched with GPT 3.5 model but recently there is a new version release GPT4. Since the introduction of ChatGPT technology it has been a topic of discussion in the technology world. This model is capable of processing a large amount of data and learning from it to perform natural language processing tasks. It can perform a wide range of applications such as answering questions, summarizing the text, translating from one language to another, and many more. ChatGPT has trained on data until September 2021, meaning it does not have knowledge of the events that occur afterward.

Chapter 4

Implementation

4.1 Tools

Various tools and technology are used during the project. This project focuses mostly on using NLP to detect autism markers. Python is the programming language to perform NLP tasks and many supporting libraries are used which will be discussed in the following sections.

4.1.1 Python

Python is one of the popular languages used by AI developers because of its platform-independent interpreter and fast operational capability to carry out scientific and mathematical calculations. During this research python version, 3.11 will be used. It supports many libraries such as nltk, etc.

4.1.2 PyCharm

PyCharm is a Python Integrated Development Environment (IDE). It provides many useful functions such as compiling and debugging. It is across platform applications and is available for many popular operating systems such as windows, macOS, Linux, etc.

4.1.3 Google Colab

Google offers a runtime environment for machine learning and data science project. It is called Colaboratory [39] or commonly known as "Colab". It is a IaaS (Infrastructure as a service) platform to run Juypter notebook. It has free as well as paid tiers. For this project, the free tier is used. The free tier offers three different runtime environments as standard, and two hardware

accelerators GPU (Graphics Processing Unit) and TPU (Tensor Processing Unit).

In the free tier, the CPU allocates 12.7 GB of RAM and 78.2 GB of hard disk space. Whereas, GPU uses NVIDIA K80 [40] processor and allocates 15 GB of RAM for a free tier account. It is illustrated in figure4.1. To run a program for a typical convolutional neural network layer over a random image CPU takes 6.62 seconds whereas GPU computes in 0.21 seconds which makes GPU 30X faster than standard CPU which can also be seen in fig 4.2 below.

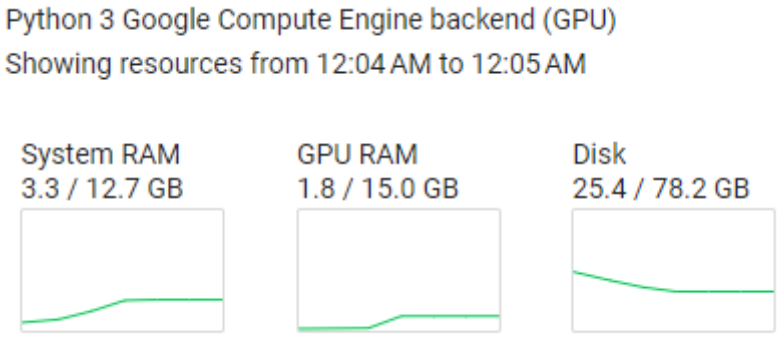

Figure 4.1: Comparison of memory consumption by CPU and GPU in free tier account of Google Colab

```
Time (s) to convolve 32x7x7x3 filter over random 100x100x100x3 images (batch x height x width x channel). Sum of ten runs.
CPU (s):
6.621239546000027
GPU (s):
0.2160315279999736
GPU speedup over CPU: 30x
```

Figure 4.2: Comparison of time consumption by CPU and GPU in free tier account of Google Colab

TPU on the other hand is a specialized hardware accelerator developed by google to support its TensorFlow software. In general, if there is a model which uses TensorFlow then running on TPU will be faster as compared to GPU.

4.2 Rule based approach

As one of the symptoms of autism is known to be slow response to questions. The idea is to find the various speech metrics like the average time taken by

the autistic person to respond to questions. This could help to find various other statistical data such as the average length of the statement.

Open-source videos are found of autistic persons on freely available websites such as YouTube. Videos contain talks between the interviewer and the autistic person. The first task is to convert the speech to text so that it can be used for analysis. This can be done with the help of a python project which can provide timestamp data of speech to text. Here timestamp is an important parameter, once the speech is converted into text. The obtained text is a raw file without speaker identifications. To calculate metrics such as average speech time and average thinking time the text obtained from speech needs to be converted to sentences with the speaker identifications tag.

4.2.1 Automatic Speech Recognition

In the theory section, ASR technique and its models have been discussed. To implement this technology open-source libraries are used. Hugging Face is an American company that provides an open-source implementation for the latest machine learning models. The wav2vec XLS-R model is also available in Hugging Face. During the implementation, "wav2vec2-xls-r-1b-english" [41] model for English and "wav2vec2-xls-r-1b-german" [42] model for German are used. This model is based on Facebook's developed XLS-R model.

To make the implementation easier Hugging Face provides a library called Transformers [43]. Transformers word here can be confusing with the transformers is also a model which is discussed earlier 2.6.1. In the context of Hugging Face transformers is a library that provides API and tools to download and train the pre-trained model. This helps to perform NLP tasks without the need for many hours of training as the models come with pre-trained weights.

Transformers have many APIs and one of the most used APIs is pipeline. Pipeline is a function that defines all the steps required to find the desired solution in the setup phase. A pipeline can be understood as a 3 step process which is pre-processing, model, and post-processing. Pre-processing is responsible for modifying data and making it suitable for model input, the model acts as the brain to do the required task and the post-processing step is responsible for decoding the information after model processing. The figure below 4.3 depicts pipeline blocks. This pipeline internally is made up of various sub-blocks figure 4.4 illustrates the whole process of automatic speech translation.

Figure 4.3: Illustration of Pipeline blocks

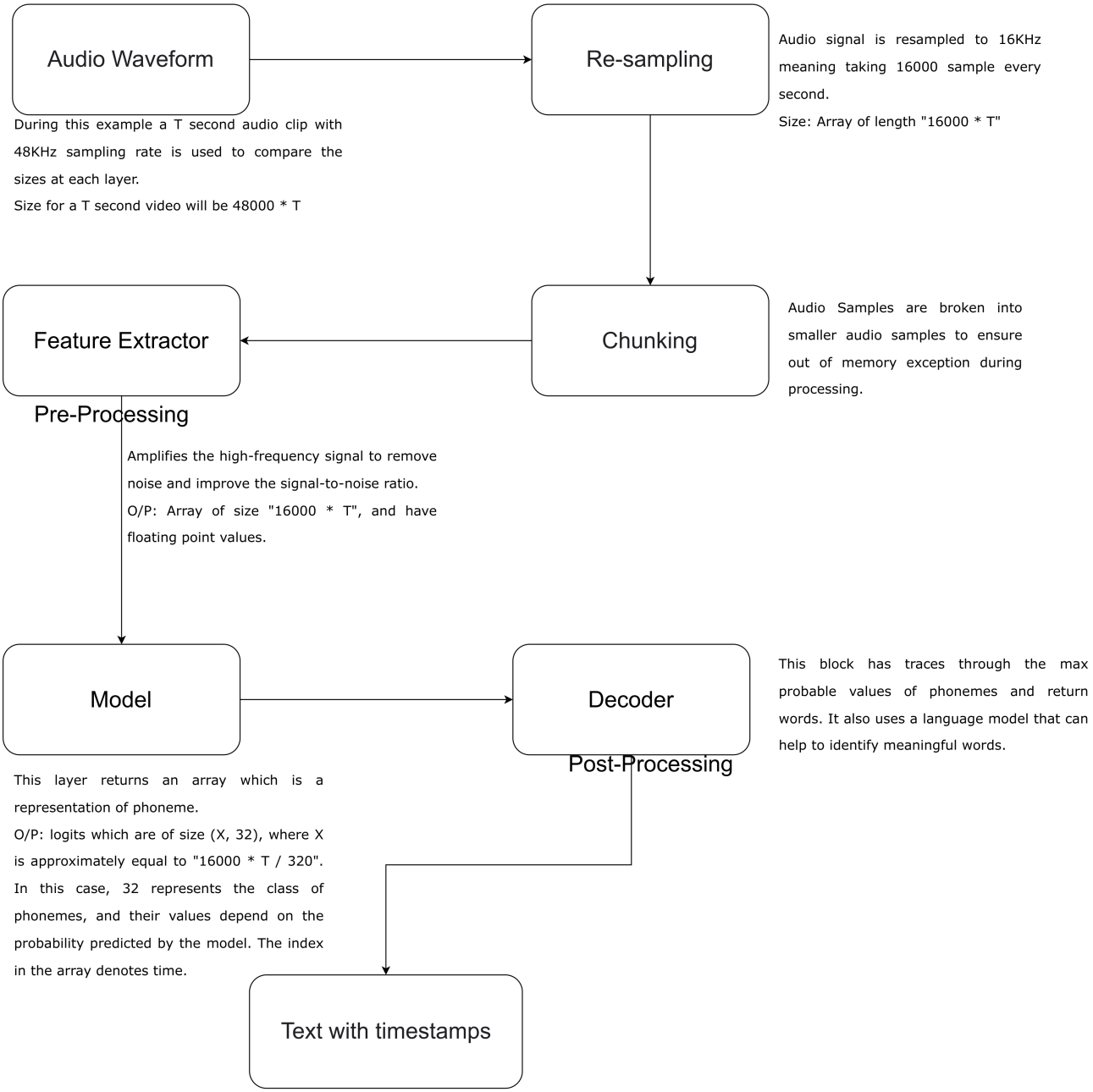

Figure 4.4: Illustration of complete automatic speech translation process

Pipeline has following parameters which is required to find the desired solution:

1. Feature Extractor: A feature extractor is responsible for preparing input features for audio models. It is used to encode waveform for the model. Audio feature extraction is a necessary step in audio signal processing, which is a subfield of signal processing. It uses a high-pass filter to amplify the higher frequencies and removes unwanted noise [44]. This phase is also known as the pre-emphasizig of audio signals. It focuses on computational methods for altering sounds. Different features capture different aspects of sound. Generally, audio features are categorized with regard to the following aspects:

   a) Level of Abstraction: High-level, mid-level and low-level features of musical signals.
   b) Musical Aspect: Acoustic properties that include beat, rhythm, timbre (colour of sound), pitch, harmony, melody, etc.
   c) Signal Domain: Signals from time domain is converted in to frequency domain to extract information from frequencies.
   d) ML Approach: Hand-picked features for traditional ML modeling or automatic feature extraction for deep learning modeling.

   The feature extractor returns the array of size "16000 * T" where T is the time in seconds. Features in the audio signals usually represented in the form of floating points numbers. These features are the phonemes of audio. Phonemes are unit of sounds which can help to distinguish between two words.

2. Model: The model that will be used by the pipeline to make predictions. For this project we have used "wav2vec2-xls-r-1b-english" and "wav2vec2-xls-r-1b-german".

   The output of model is called logits. Logits in general are the logarithmic representation of probabilities. The output of model is a score for a particular class, this score can range from 0 to 1. This score can be mapped into real numbers from -∞ to ∞. Scores can be converted int o logits by using the equation 4.1. Here p is the score and the logarithm ln has base e.

$$logit = ln\frac{p}{1-p} \quad , p \in (0,1) \tag{4.1}$$

In this context, logits are array of size (X, 32) where X is the number of number of logits in the audio. Usually it is approximately calculated by using equation 4.2. The index of this array indicates time of the signal. 32 are the number of classes for each of sounds such as A, B, C, and so on. The model outputs probabilities for each sound as per phonemes. Also these 32 classes represents unique character given in B.

$$numberoflogits = ln\frac{16000 * T}{320} \tag{4.2}$$

Only the maximum probability scores are used for each array to predict the text. This argmax value is calculated for each tensor. argmax is a pyTorch function which return the index of maximum value in of an array. Once the maximum value index of logits are carried out the data reduces to an array of X size.

3. Processor: This processor is also an API of transformers just like pipeline which defines the model to be used for finding the solution. Once this processor is defined tokenizers and feature extractors can be easily set as the processor contain all such model definitions. For speech translation Wav2Vec2ProcessorWithLM process is used.

4. Tokenizer: Tokenizer in general means a method that prepares inputs for a model or decodes the list of tokens into strings. Tokenizer in this context is used to decode the logits in to a sequence of texts. For each model there is a tokenizer available in Hugging Face library. In this project "Wav2Vec2CTCTokenizer" is used.

5. Decoder: The output signals of feature extractor are fed into model which gives the context output which is then fed into decoder to get the text. This is the post-processing part of pipeline. "BeamSearchDecoderCTC" decoder is used for the current project.

   The Decoder again uses tokenizer to decode the logits maximum probable values. The output of decoder will be a sequence of text. If required the timestamps can also be carried out with text. As the decoder also outputs the start and end offset of each word using these offsets time can be calculated using the below formula.

   First the total audio length time is calculated which is the ratio of total number of samples and sampling rate, this can be seen in equation 4.3. In this case the sampling rate is fixed to 16000 or 16K Hz. Now this

total time is distributed into number of logits so each logit time is equal to total time divided by total number of logits, illustrated in equation 4.4. So this time for each logit can be further multiplied by start and end offset of decoder output to know the timing of the word.

$$totalTime = \frac{totalNumberOfSamples}{samplingRate} \quad seconds \tag{4.3}$$

For the wav2vec2 model, sampling rate is fixed to 16KHz or 16000

$$timeforeachlogit = \frac{totaltime}{totalnumberoflogits} \quad seconds \tag{4.4}$$

6. Chunking: In order to process long audio files, more computation space is required but due to limitation of such processor it's a best practice to chunk (seperate) the audio signals into multiple audio files and then do the processing. It is very beneficial to avoid out of memory exception for long waveforms [45]. Chunking can be done in many ways:

   a) Simple Chunking: Simple chunking is the process of breaking audio into smaller audio samples based on silence in the audio, but sometimes it can be difficult to find silence in a long waveform or another way is to simply chunk after every 10 seconds. The problem here comes is when audio is broken at the time between a word then it's difficult to translate the word.

   b) Chunking with stride: This is another method of chunking where chunks can be created for every 10 seconds of audio but it can be added with a stride. Strides are the overlap chunks with the length of the stride smaller than the chunk size. This helps to determine the word which is said during the inference of two chunks. As an example below 4.5 represents chunking with stride. It can be seen the overlapped letters are dropped and the actual word is constructed. This also used Language models such as n-grams which are helpful in determining which letter can come next to the current letter.

      The figure 4.5 represents a long audio waveform of lets assume "Hello World". The below blocks shows an example how the chunks are created for every 10 seconds for example. there will be stride which is overlapped for every chunk for lets assume every 2 seconds.

This will be in terms of milliseconds in practice but just for this example it is considered in seconds.

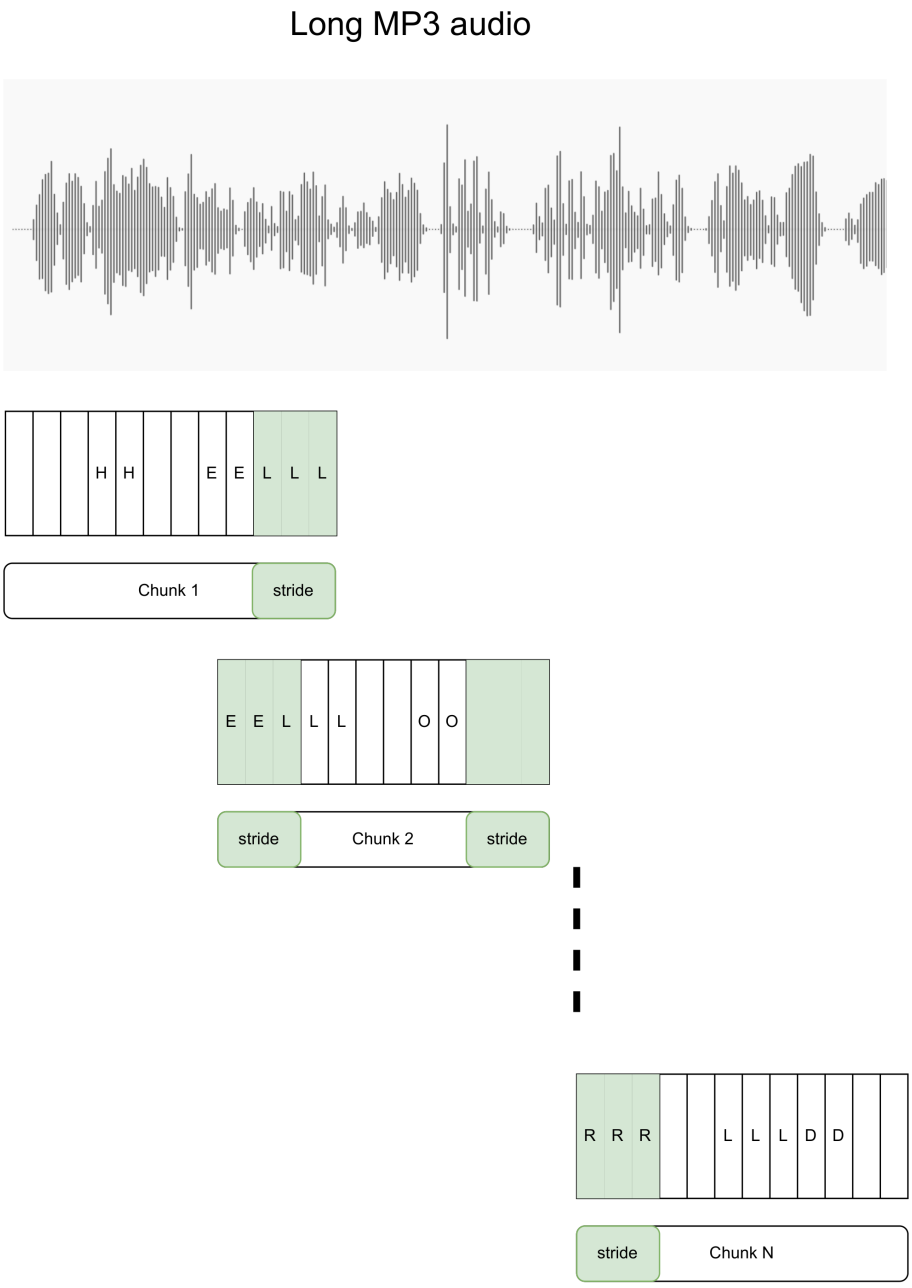

Figure 4.5: Block diagram of Chunking with stride with example of "Hello World"

Example for ASR

An example to understand this is shown below in figure 4.6. The input to this pipeline is self-recorded voice "Hello i am a boy". The sampling rate for

the voice is 48KHz so to process this with wav2vec2 model, the input voice needs to be converted into 16KHz. The conversion is done with the help of the librosa library which produces an array of samples 41463. Here 41463 number of samples is because the length of the input voice is approximately 2.5 seconds, Thus 16000 multiplied by 2.5 is 41463.

Further, this is chunked to be processed parallelly by the model. Then the input sample is passed through the feature extractor where for each sample the noise is reduced and the signal is amplified. The amplified signal is then fed into the model, which uses its machine learning algorithm to detect phonemes. There are 32 classes representing the sound of a different word. The model predicts phoneme probability for each class, this is also classed logits. Interestingly, for every logit, there is a non-zero logit score occurs. It is for the first value which represents silence. It means either the phoneme represents a word or otherwise silence.

The maximum probability for each logit is then passed into decode. This step is important as only the maximum probability can define a word and all the rest can be ignored. The decoder now decodes the logits and produces text, the start offset and end offset is also produced by the decoder this can be further used to calculate time by using equations 4.3 and 4.4.

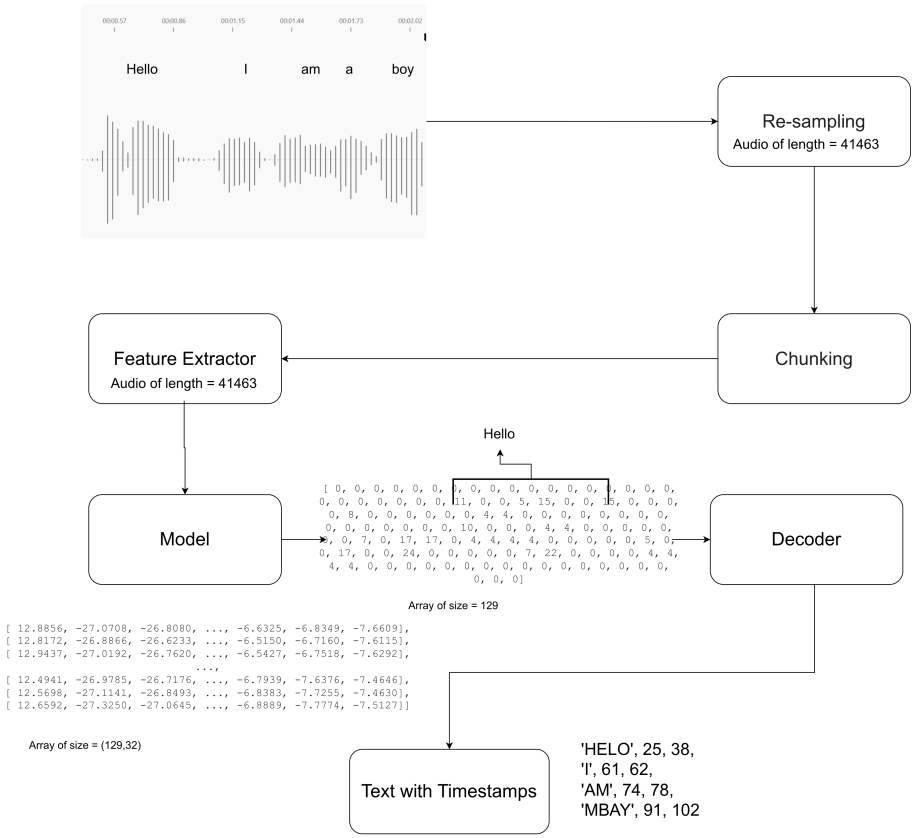

Figure 4.6: ASR pipeline with example

4.2.2 Speaker Diarization

Speaker diarization methods are discussed in the theory section of this paper. Hugging Face also provides the implementation of the speaker diarization process. It provides an open-source pre-trained model named "pyannote/speaker-diarization" for the implementation of speaker diarization. The pre-trained model takes any audio sample as input and the output is speaker identity and timestamps. The audio sample should be in waveform format to process the file. This method comes under the clustering based model where the model groups together similar segments of audio based on their acoustic characteristics such as pitch or signal energy.

The high-level working principal of a speaker diarization process is discussed below in the form of a block diagram 4.7.

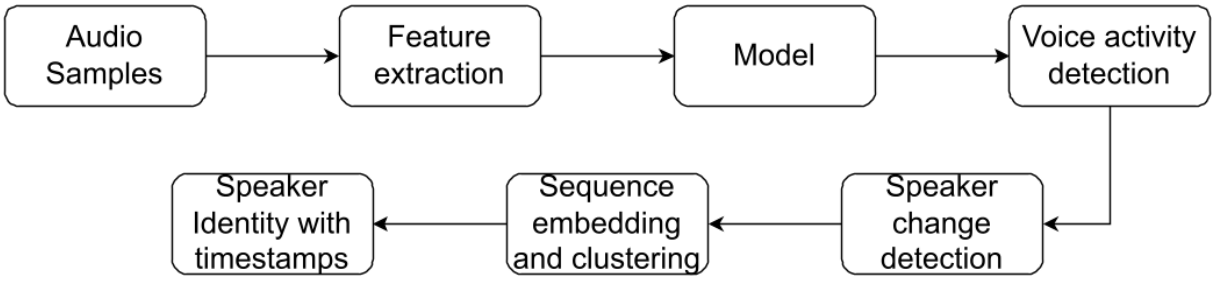

Figure 4.7: Speaker Diarization block diagram

This process is made up of following blocks [37]:

1. Feature extraction: Features are extracted from audio signals based on acoustic features such as pitch or signal energy. There are various feature extraction representation technique but one of the widely used is MFCC [46]. Given a audio signal the feature extractor turns in to sequence of feature vectors X = x1, x2, ... xt. This feature vector will be used for further calculations.

2. Model: This step can be understood as sequence labeling task. The feature vectors from the previous step is fed into model. X is input to the model. The model transform data from X to Y. This step is labeling the data into a sequence of where X is labelled into Y vectors. The expected output after all processing is a sequence of labels y = y1, y2,. . . ,yt.

3. Voice activity detection: This process detects the presence of voice in the audio samples. A tunable threshold is fixed and any value greater than the threshold is marked as speech. The speech at any time step is denoted as yt = 1 and no speech is denoted as yt = 0.

4. Speaker change detection: This task detects the speaker change points. In this step at test timestamps if the prediction scores of local maxima is greater than threshold then a speaker change is detected. The label sequence is created with yt = 0 for no change and yt = 1 for a change in speaker.

5. Sequence embedding and clustering: This is one of the most important part of any speaker diarization pipeline. In this step clustering is performed. Clustering in simple terms is grouping. Grouping of speech segments as per identity of the speaker is performed. It is performed with the help of similarity matrix.

4.2.3 Change text to statements with speakers identifications

Previously, the speech was converted to text and the speaker's identity was determined along with their corresponding timestamps. In order to calculate different speech metrics, it is necessary to transform the text into a sequence of text that includes the speakers identity and timestamps. This can be achieved by using python functions. Below figure 4.8 represents the implementation of text sequence to statements.

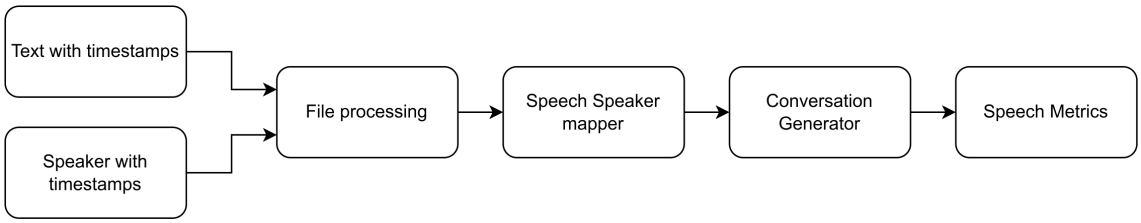

Figure 4.8: Block diagram of speech to text to calculate speech metrics

1. File Processing: Automatic speech recognition and speaker diarization process in turn generates separate comma-separated values (csv) file with the file extension ".csv". To carry out csv file read operation a python package is used called "csv". It sequentially reads the data and saves the result into an array. Finally this array is returned for further processing.

   Here csv file reader is used twice, one time for the automatic speech recognition time and another time for speaker diarizatoin.

2. Speech Speaker Mapper: This is a user defined python function which maps speech to speaker.

   The array for speech recognition comprises a list of text along with their corresponding starting and ending times. On the other hand, the array for speaker diarization contains the starting and ending times for a specific speaker. Now the speech text is iterated for each text and corresponding start and end time is matched with the speaker identity.

   If the duration of text falls within the time frame of a speaker, that speaker is recognized and the text assigned to be the respective speaker.

   For Example: Lets suppose if speaker A says "Hello I am speaker A" for 1 to 5 seconds. The ASR file generated will look like below:

1, 2, Hello
2, 3, I
3, 4, am
4, 5, speaker
5, 5, A

and speaker diarizatoin file will look like :

1, 5, speakerA

When ASR array is iterated and for each value speaker diarizaiton values are checked, if the text duration falls between speaker diariation the speaker is mapped. A new array is returned with text start time, text end time, sequence of text, and speaker.

3. Conversation Generator: After the previous step, where the speech and speaker were mapped to create a sequence along with the respective speaker's identity, the objective of this current step is to create statements by amalgamating the text from a speaker until there is a change in the speaker.

   The speech is processed repeatedly and the text obtained is stored temporarily. During each iteration, the speaker's identity is compared to that of the previous iteration. If the current speaker is the same as the previous one, the iteration continues. However, if a change in speaker is detected, the text saved in the temporary storage is attributed to the current speaker along with a sequence number. Finally an array is returned consisting of statement, sequence number, speaker, start time of statement, and end time of statement.

4. Speech Metrics: The final step is to calculate speech metrics.

   The audio samples are chosen based on the assumption that one of the person is interviewer and another person is respondent. In couple of the text audios respondent is person with autism. The various speech matrices such as average length of a sentence, average time by a speaker, average time to speak a word and average time for thinking can be calculated using the below expressions 4.5, 4.6, 4.7, and 4.8 respectively.

$$AverageSentenceLength = \frac{TotalWordCount}{TotalNumberOfStatements} \quad (4.5)$$

The output will be number of words per statement.

$$AverageSpeechTime = \frac{StatementTime}{TotalNumberofStatements} \tag{4.6}$$

Output of above equation 4.6 will be in seconds

$$AverageWordTime = \frac{StatementTime}{TotalWordCount} \tag{4.7}$$

This equation 4.7 output will in in seconds

$$AverageThinkingTime = \frac{TotalThinkingTime}{TotalNumberOfStatements} \tag{4.8}$$

Output of above equation 4.8 will be in seconds.

It is feasible to compute these metrics for speakers belonging to various groups, including normal individuals various age categories and autistic persons of different age ranges and a comparison can be carried out to know if there is any such trend exists.

4.2.4 Identify question and predict type of question

Questions in a speech can easily be predicted by looking for interrogative words like what, when, where, why, do, does, and many more. The next part is to predict the type of question. Questions can broadly be classified into two main types first one is a "WH" question which belongs to questions like what, where, when, etc and another one is a "YES/NO" question meaning the answer to the question can simply be given in the form of "YES or NO". Naive Bayes Classifier is used to predict the type of question. This is already discussed in the earlier section 2.5.

The nltk library [27] provides the implementation of Naive Bayes theorem. It separates labels and features from the training data and trains the classifier. The classifier is trained on Naive Bayes theorem where the probability of each label and the probability of each feature given label is calculated. Once the training is done the classifier can be tested on testing data to know the accuracy. The aim should be to better accuracy to predict question type accurately. This can be achieved by training model with accurate labels and large data sets. After the classifier is set the statement by a speaker can be fed into the classifier and labels can be determined.

4.3 Machine learning approach

Another approach to identifying autism disorder is to use a machine learning approach. Eye gaze is one of the useful biomarkers used in the detection of autism in children. Children with autism show reduced eye gaze [47]. Researchers are now working on algorithms to find patterns in eye gaze movement. This can be an effective method in detecting autism spectrum disorder.

Chapter 5

Results

As explained in earlier sections, this report covers the remedies to obtain speech-to-text transcriptions alongside speaker identification and timestamps. Additionally, it highlights multiple speech metrics for various groups. The subsequent sections will provide a detailed discussion of these outcomes.

Data used in this project is obtained from open source recordings, and some personal recordings. This also poses a challenge to effectively measure the metrics and obtain results. But this report is effective in terms of technology to speech to text translation and speech metrics.

5.1 ASR and Speaker Diarization

As the primary objective of this project is to aid in the identification of autistic symptoms in individuals with ASD, this requires to analyze high volume of data in the form of large video sizes. The video recordings can be more that 1 hour long. This section discusses the time taken by the model to process videos of various sizes.

The below table 5.1 shows the time taken by ASR and SD process to compute audio of various length. All the time is in seconds.

Audio Time (in sec.)	ASR (in sec.)	Speaker Diarization (in sec.)
190.8	4.2	20
303	6	24
606	12.6	44
747	22	53
925.8	26	60
3600	232	420

Table 5.1: Table to show time required by ASR and SD to process audio of different length.

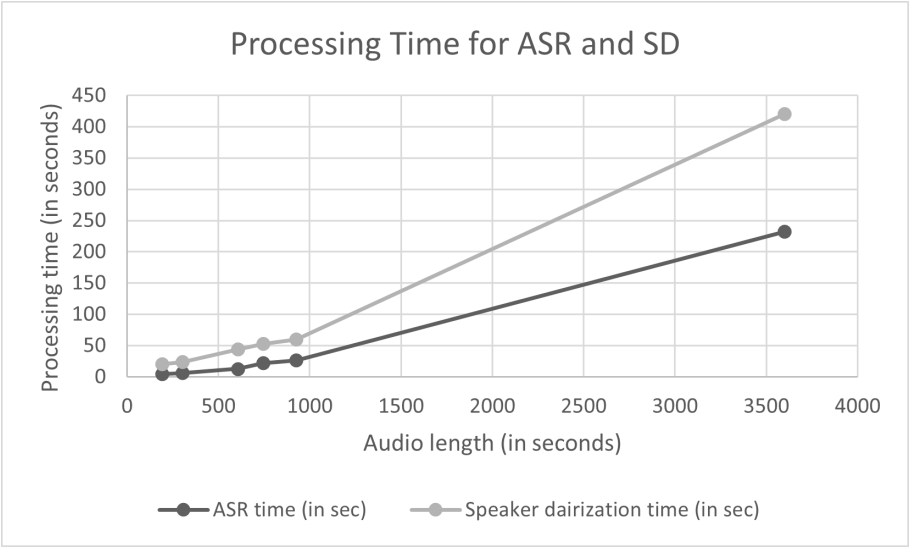

Figure 5.1: Graphical representation of above table 5.1

The above figure 5.1 plots the time taken by different processors such as automatic speech recognition and speaker diarization vs audio length. Here all the time is in seconds. This experiment is conducted for short to long videos, that is from 3-minute videos to 1-hour-long videos. It can be interpreted from the data that the time taken by speaker diarization process is higher than that of automatic speech processing. It is also important to note here that ASR uses chunking, which significantly improves performance and speaker diarization is without chunking. For small audios, the ASR takes one-fourth time of speaker diarization but for a longer video, this time difference reduces to half of the time. Both process separately shows a linear increase in time consumption as the length of audio increases.

5.2 Speech Metrics

One of the feature of this project is to calculate the various metrics in conversation. Metrics can form a quantitative insight in recognizing autism spectrum disorder, some of the symptoms in autism include are slow response in social interaction can be detected by using matrices. The below tables 5.2, 5.3 and 5.4 summarizes average sentence length, average speech time and average word time for different audio length using equations 4.5, 4.6 and 4.7 respectively.

Audio Time	Av. sentence length interviewer	Av. sentence length respondent
190.8*	35.57	14.66
303	96	37.2
606	7	66.41
747*	52.38	4.42
3600	34.77	26.96

Table 5.2: Average sentence length for interviewer and respondent for different audio length (* autistic interviews).

Audio Time	Av. speech time interviewer	Av. speech time respondent
190.8*	20.47	6.71
303	28.94	35
606	2.71	43.83
747*	30.57	2.43
3600	11.19	7.84

Table 5.3: Average speech time for interviewer and respondent for different audio length (* autistic interviews).

Its can be noticed from tables 5.2 and 5.3 that if the sentence length is more the average speech time is also more, meaning if a person is speaking for more time then average speech time will be more and also number of words in a single speech streak will also be more.

This result only forms a pattern to calculate speech matrices, for the detection of autism spectrum disorder a large data sheet is recurred based on various groups such as age and region. However, during the experiment phase, most of the videos used are from open source tools, which mostly has edited and this poses a challenge in metrics authenticity.

Audio Time	Av. word time interviewer	Av. word time respondent
190.8*	0.57	0.45
303	0.3	0.94
606	0.38	0.66
747*	0.58	0.55
3600	0.32	0.32

Table 5.4: Average word time for interviewer and respondent for different audio length (* autistic interviews).

The table below 5.5 shows that the thinking time is case of autistic respondents are much higher compared to normal person. Another finding is in case of autistic interviews (* marked) the average metrics for autistic person is found to be lesser than interviewer meaning the autistic person do not participate in conversation. This certainly need to be examined for large data sets to have a better evidence.

Audio Time	Av. thinking time of respondent
190.8*	0.26
303	0.06
606	0.108
747*	0.47
3600	0.006

Table 5.5: Average thinking time of respondent for different audio length (* autistic interviews).

Chapter 6

Conclusion

Recent innovation in natural language processing field can help in detection of autism spectrum disorder symptoms. This section will summarize the work achieved during the course and future scope of this project.

6.1 Summary

There are evidences that language skills play a significant role in detection of autism spectrum disorder. The person with autism sometimes confuses between "WH" questions for example, if What question is asked they may understand this as why question and answer significantly.

Automatic speech recognition is one of the important technology in the process of speech analysis. Recently there are many models in ASR but during this project, Facebook developed xls-r based wav2vec2 model is discussed in depth. The model is helpful in recognizing text from different languages. This model is chosen over other models as this project is based in Germany and most of the children are native German speakers, but sometimes there a need may also arise to detect another language such as English, French, or Spanish.

Automatic speech recognition is made up of several blocks such as a feature extractor, model, and decoder. The feature extractor extracts the features and reduces noise to improve the signal-to-noise ratio, the model labels the features into different classes, and the decoder output the text with offsets. This offset can later be used to calculate the time of the word. It is also supported by a language model which helps it to make meaningful sentences. Due to physical factors of sound such as noise, overlapping speech, and change in the pronunciation of certain words this model sometimes produces error word output but this can be fine-tuned with more data and better noise reduction techniques.

Another technology that is involved in speech processing is the speaker diarization process which maps the audio time with speaker identity. This is a clustering based model where the speech segments are clustered with similar speech segments based on physical properties of sound like pitch and tone.

Text from ASR is combined with speaker identity to form sentences with speaker identity. To calculate various speech metrics such as average sentence length, average speech time is calculated by knowing the time taken for each sentence and the number of words spoken during the duration.

Lastly, the statements from the speech are used to identify if a particular statement is a question or not. This is achieved by using Naive Bayes classifier. This classifier uses a statistical approach to predict if a sentence contains some words then it is an interrogative sentence or not. This classifier also predicts the type of question such as "WH" question or "YES/NO" question.

The time to process ASR and speaker diarization is linear, meaning as the input audio length increases processing time also increases. During the experiments up to one hour of videos are processed. The main hurdle in ASR comes during the decoding phase as all the logits are needed to be evaluated at once and this completely occupies the memory space which throws out of memory exceptions.

6.2 Future Scope

This project laid a foundation to process natural language in various languages to translate into text with speaker embedding. While recognizing speech there were some errors that can be made better by using different decoders with better language models.

Speaker diarization also sometimes produces an error in detecting overlapped speech from various speakers especially if a speaker is talking in a low pitch. This can be further improved by using techniques such as reducing signal-to-noise ratio.

Also, the current model to detect interrogative sentences works only for the English language further this classifier can be trained to detect interrogative sentences in German language and for other regional languages as well.

Appendix A

Naive Bayes classifier

Naive Bayes is widely used algorithm for classification problems. This can be understood by using an example. Let us assume a football game, the aim of the game is to score a goal. Goal can be scored or not depends upon the strike player abilities and defending player abilities. Lets assume weather conditions can also affect the chances of scoring a goal. So in total there are three features weather, striker and defence and the outcome of this will be goal.

Below table summarises the various possibilities and past results.

Weather	Striker	Defence	Goal
Sunny	Good	Bad	Yes
Hot	Average	Bad	Yes
Rain	Bad	Average	No
Hot	Good	Good	No
Rain	Average	Average	No
Sunny	Bad	Good	No
Rain	Good	Bad	Yes
Hot	Bad	Bad	Yes
Sunny	Good	Good	Yes
Rain	Good	Average	Yes
Hot	Bad	Average	No

Table A.1: Naive Bayes classification example

The above table A.1 represents various states of weather, Striker and Defence. Now the conditional probabilities can be calculated by counting the

number of labels. Below tables A.2, B.1, A.4 and A.5 represents the conditional probabilities of label given features like weather, Striker and defence respectively.

P(Weather)	Number of Yes	Number of No	P(Y|Weather)	P(N|Weather)
Sunny	2	1	2/6	1/5
Hot	2	2	2/6	2/5
Rain	2	2	2/6	2/5
Total	6	5		

Table A.2: Probability of label|Weather

P(Striker)	Number of Yes	Number of No	P(Y|Striker)	P(N|Striker)
Good	4	1	4/6	1/5
Average	1	2	1/6	2/5
Bad	1	2	1/6	2/6
Total	6	5		

Table A.3: Probability of label|Striker

P(Defence)	Number of Yes	Number of No	P(Y|Defence)	P(N|Defence)
Good	1	2	1/6	2/5
Average	1	2	1/6	2/5
Bad	4	1	4/6	1/5
Total	6	5		

Table A.4: Probability of label|Defence

P(Goal)	Events	P(label)
Yes	6	6/11
No	5	5/11
Total	11	

Table A.5: Probability of Goal

Now the next task is to calculate the probability of test condition. So lets assume test condition to be probability of scoring a goal when weather is sunny, strike player is average and defence is bad. This can be equated in equation 2.2 as below A.3 and A.4 assuming the denominator to be constant as of now. At the later stage this denominator will be cancelled out in calculation so its not required.

$$P(Yes|S,A,B) = P(Yes)*P(Sunny|Yes)*P(Average|Yes)*P(Bad|Yes) \tag{A.1}$$

$$P(No|S,A,B) = P(No)*P(Sunny|No)*P(Average|No)*P(Bad|No) \tag{A.2}$$

After equating values in above equation from above tables these equation becomes:

$$P(Yes|S,A,B) = \frac{6}{11}*\frac{2}{6}*\frac{1}{6}*\frac{4}{6} = \frac{2}{99} \tag{A.3}$$

$$P(No|S,A,B) = \frac{5}{11}*\frac{1}{5}*\frac{2}{5}*\frac{1}{5} = \frac{2}{275} \tag{A.4}$$

To calculate probability of scoring a goal will be

$$P(Goal) = \frac{P(Yes|S,A,B)}{P(Yes|S,A,B)+P(No|S,A,B)} = \frac{\frac{2}{99}}{\frac{2}{99}+\frac{2}{275}} = 0.7353 \tag{A.5}$$

And probability of not scoring a goal will be equation below

$$P(Goal) = \frac{P(Yes|S,A,B)}{P(Yes|S,A,B)+P(No|S,A,B)} = \frac{\frac{2}{275}}{\frac{2}{99}+\frac{2}{275}} = 0.2647 \tag{A.6}$$

It can be seen from the above equation that the probability of scoing a goal is higher compared to not scoring a goal.

This classifier can similarly be used to determine the type of question depending upon the words used in a sentence the features and labels can be calculated and then for a test condition the various probabilities can be used to calculate the probability of label.

Appendix B

Automatic Speech Recognition Classes

In the earlier section it is noticed that the model output provides logits in 32 classes these 32 classes represents some words and they are as follows:

Class	Value	Class	Value	Class	Value
0	Silence	11	f	22	q
1	$< /s>$	12	g	23	r
2	?	13	h	24	s
3	Repetition	14	i	25	t
4	′	15	j	26	u
5	–	16	k	27	v
6	a	17	l	28	w
7	b	18	m	29	x
8	c	19	n	30	y
9	d	20	o	31	z
10	e	21	p		

Table B.1: Classes to word representation mapping in automatic speech recognition decoder

Bibliography

[1] Signs and symptoms of autism spectrum disorder. *Centers for Disease Control and Prevention*. URL: https://www.cdc.gov/ncbddd/autism/signs.html [cited 28.03.2022].

[2] Autism spectrum disorder. *Mayo Clinic*. URL: https://www.mayoclinic.org/diseases-conditions/autism-spectrum-disorder/symptoms-causes/syc-20352928.

[3] Tim Dutton. The beneficial ai movement. *Medium*. URL: https://medium.com/politics-ai/the-global-politics-of-ai-1-the-beneficial-ai-movement-b17ac411c45b.

[4] Engineering with the brain. *Neuralink*. URL: https://neuralink.com/applications.

[5] What is alexa? *Amazon*. URL: https://developer.amazon.com/en-US/alexa.

[6] Siri. *Apple*. URL: https://www.apple.com/siri/.

[7] Nicholas C Jacobson Cyrus Chang Eduardo L Bunge Gilly Dosovitsky, Blanca S Pineda. Artificial intelligence chatbot for depression: Descriptive study of usage. *Journal of Medical Internet Research*. URL: https://formative.jmir.org/2020/11/e17065.

[8] Chatgpt. https://chat.openai.com/chat. Accessed: 2022-01-12.

[9] Jennifer "Jay" Palumbo. A new study shows the importance of early diagnosis of autism spectrum disorder. *forbes*. URL: https://www.forbes.com/sites/jenniferpalumbo/2021/12/31/how-a-new-study-shows-the-importance-of-early-diagnosis-of-autistic-infants/?sh=417afd8634b4.

[10] Taha H. Rassem Mohammed Hamzeh Salameh Ahmad Shatnawi Salwa Mutahar Alwazer Ibrahim Abdulrab Ahmed, Ebrahim Mohammed Senan and Mohammed Alshahrani. Eye tracking-based diagnosis and early detection of autism spectrum disorder using machine learning and deep learning techniques. *MDPI*. URL: https://doi.org/10.3390/electronics11040530.

[11] Mahmoud Elbattah; Jean-Luc Guérin; Romuald Carette; Federica Cilia; Gilles Dequen. Nlp-based approach to detect autism spectrum disorder in saccadic eye movement. *IEEE*. URL: https://doi.org/10.1109/SSCI47803.2020.9308238.

[12] Soorena Salari Mariam Zomorodi-Moghadam Moloud Abdar U Rajendra Acharya Reza Khosrowabadi Vahid Salari Zeinab Sherkatghanad, Mohammadsadegh Akhondzadeh. Automated detection of autism spectrum disorder using a convolutional neural network. *frontiersin*. URL: https://doi.org/10.3389/fnins.2019.01325.

[13] Ibrahima Faye-Ahmed Faeq Hussein Mohammed Isam Al-Hiyali, Norashikin Yahya. Identification of autism subtypes based on wavelet coherence of bold fmri signals using convolutional neural network. *MDPI*. URL: https://doi.org/10.3390/s21165256.

[14] Cragun Brian J. Eichenlaub Anthony W. Petri John E. Unterholzner John C. Clark, Adam T. Diagnosing autism spectrum disorder using natural language processing. *United States Patent Application 20160180038*. URL: https://www.freepatentsonline.com/y2016/0180038.html.

[15] Dsm-5. *Autism Society*. URL: https://www.autism-society.org/what-is/diagnosis/diagnostic-classifications.

[16] Screening diagnosis identifying autism. *Autism Society*. URL: https://autismsociety.org/screening-diagnosis.

[17] Alan Schnee. What goes into teaching children to answer wh questions? *Association for Science in Autism Treatment (ASAT)*. URL: https://asatonline.org/research-treatment/clinical-corner/teaching-wh-questions/.

[18] Signs of autism in children. *National Health Service*. URL: https://asatonline.org/research-treatment/clinical-corner/teaching-wh-questions/.

[19] NIDCD Information Clearinghouse. Autism spectrum disorder: Communication problems in children. *National Institute on Deafness and Other Communication*. URL: https://www.nidcd.nih.gov/health/autism-spectrum-disorder-communication-problems-children.

[20] Nltk corpora. *NLTK Org*. URL: https://www.nltk.org/nltk_data/.

[21] Eliza. *Wikipedia*. URL: https://en.wikipedia.org/wiki/ELIZA.

[22] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge University Press, 2008.

[23] Tokenization. *Count Words Free*. URL: https://countwordsfree.com/stopwords.

[24] M. F. Porter. An algorithm for suffix stripping. URL: https://www.cs.toronto.edu/~frank/csc2501/Readings/R2_Porter/Porter-1980.pdf.

[25] Part-of-speech tutorial. *Google Sites*. URL: https://sites.google.com/site/partofspeechhelp/home/dt_nnp.

[26] Daniel Jurafsky, James H. Martin, and Hinrich Schütze. *Speech and Language Processing*. Stanford University, 2023.

[27] Source code for nltk.classify.naivebayes. https://www.nltk.org/_modules/nltk/classify/naivebayes.html. Accessed: 2023-15-02.

[28] Speech recognition from audrey to alexa – a brief history. *dictate It*. URL: https://dictateit.com/speech-recognition-from-audrey-to-alexa-a-brief-history/.

[29] Shazeer N. Parmar N. Uszkoreit J. Jones L. Gomez A. N. Kaiser L. Polosukhin I. Vaswani, A. Attention is all you need. *ArXiv*, 2017. URL: https://arxiv.org/abs/1706.03762.

[30] K-means. *Stanford*. URL: https://nlp.stanford.edu/IR-book/html/htmledition/k-means-1.html.

[31] Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, and Oriol Vinyals. Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(2):356–370, 2012. doi:10.1109/TASL.2011.2125954.

[32] Davy M. Houacine A.. Fergani, B. Speaker diarization using one-class support vector machines. *Elsevier*, 2008. URL: https://doi.org/10.1016/j.specom.2007.11.006.

[33] Babu A. Wang C. Tjandra A. Lakhotia K. Xu Q. Goyal N. Singh K. von Platen P. Saraf Y. Pino J. Baevski A. Conneau A. Auli, M. Xls-r: Self-supervised cross-lingual speech representation learning at scale. *ArXiv*, 2021. URL: https://arxiv.org/abs/2111.09296.

[34] Hannun A. Xu Q. Cai J. Kahn J. Synnaeve G. Liptchinsky V. Collobert R. Pratap, V. Wav2letter++: The fastest open-source speech recognition system. *ArXiv*, 2018. URL: https://arxiv.org/abs/1812.07625.

[35] Baevski A. Collobert R. Auli M. Schneider, S. Wav2vec: Unsupervised pre-training for speech recognition. *ArXiv*, 2019. URL: https://arxiv.org/abs/1904.05862.

[36] Core transcription. https://www.assemblyai.com/products/core-transcription. Accessed: 2023-03-15.

[37] Bredin H. Yin R. Coria J. M. Gelly G. Korshunov P. Lavechin M. Fustes D. Titeux H. Bouaziz W. Gill, M. Pyannote.audio: Neural building blocks for speaker diarization. *ArXiv*, 2019. URL: https://doi.org/10.48550/arXiv.1911.01255.

[38] Jennifer Langston. Microsoft announces new supercomputer, lays out vision for future ai work. https://news.microsoft.com/source/features/ai/openai-azure-supercomputer/. Accessed: 2023-15-03.

[39] Google colaboratory. https://colab.research.google.com/. Accessed: 2022-11-15.

[40] Cloud gpus. https://cloud.google.com/gpu. Accessed: 2022-11-15.

[41] Fine-tuned xls-r 1b model for speech recognition in english. https://huggingface.co/jonatasgrosman/wav2vec2-xls-r-1b-english. Accessed: 2022-11-15.

[42] Fine-tuned xls-r 1b model for speech recognition in german. https://huggingface.co/jonatasgrosman/wav2vec2-xls-r-1b-german. Accessed: 2022-11-15.

[43] Transformers. https://huggingface.co/docs/transformers/index. Accessed: 2022-11-15.

[44] Vieting P. Lüscher C. Michel W. Schlüter R. Ney, H. On architectures and training for raw waveform feature extraction in asr. *ArXiv*, 2021. Human Language Technology and Pattern Recognition Group, Computer Science Department, RWTH Aachen University. URL: https://doi.org/10.48550/arXiv.2104.04298.

[45] Making automatic speech recognition work on large files with wav2vec2 in transformers. https://huggingface.co/blog/asr-chunking. Accessed: 2022-11-15.

[46] Park T. J. Kanda N. Dimitriadis D. Han K. J. Watanabe S. Narayanan, S. A review of speaker diarization: Recent advances with deep learning. *ArXiv*, 2021. URL: https://doi.org/10.48550/arXiv.2101.09624.

[47] Carrasco-Ribelles LA Marín-Morales J Minissi ME Teruel-García G Sirera M Abad L. Alcañiz M, Chicchi-Giglioli IA. Eye gaze as a biomarker in the recognition of autism spectrum disorder using virtual reality and machine learning: A proof of concept for diagnosis. *PubMed*, 2022. URL: https://pubmed.ncbi.nlm.nih.gov/34811930/.