UNO Morse Conversie

***Abstract*—Morse Code is a secret code which is used by various agents like RAW, Intelligence for transmitting confidential info to their respective offices for safety of country. We can convert their manual process of sending signals through normal machine, manual decoding to a pen type single button(UNO) transmitter, automated decoding conversion(Conversie) . Decoding the received signal into audio is our project.**

I. Introduction

Trying to solve the problem:

• Covert operations use Morse code for transmitting secret info. But at destination, it takes time to decode it. Why not develop a program which can automate the conversion? Why not make it easy for these agents to communicate easily with the departments?

• Handicapped can't communicate freely with the rest. Why can't we stand as their voice?

These are the main practical challenges which made us think of this project of converting Morse signal into audio signal using Machine Learning concepts.

It is important and challenging problem because:

• Existing solution for transmission of Morse by secret agents is by using a machine connected to electricity which is immobile and should be done too carefully. It requires antennas for transmission which can't be hidden.

• Wasting time in converting the morse code might result in dangers when it comes to secret operations. Transmission and Analysis of info must be as soon as possible. Delay in this would cost lives of secret agents and would result in disasters.

• Communication, exchange of knowledge shouldn't be limited to some people, it must not be a challenge for some people to communicate or express their feelings. Everyone must be able to communicate and exchange the knowledge. Normal people are living a luxurious life by automating everything but handicapped can't even communicate easily.

Existing Solutions are:

• Manual conversion of received morse from agents by specialised recruited people which is time consuming and delays in this can result in disasters. Moreover, these recruited people must be always waiting for the Morse code to come. They might not know when the code might come.

• Braille script, which can be used for blind to understand the books but they can't communicate like us in case they are dumb.

• Sign language is used currently for communication in case of handicapped people but morse code would be good because they can be converted into audio

Core idea of our project is:

• To basically automate the process of converting the received morse from agents into audio within seconds. No people are required to wait for the morse, the system takes care of everything, they can get notification when they receive it. Our project is basically conversion.

• To convert the morse (the language with which handicapped can communicate) into audio (the language which the humans can understand easily).

Aspects which our project can solve:

For demonstration, we have taken the case of a spy sending the morse signal to departments like RAW.

• Agent or spy can use pen button to send the morse signal so easily and conversion into audio is also made automated by our work. That button is represented as "m" in our demonstration for time being.

• So, basically, we are trying to automate this using Machine learning concepts i.e., Kmeans Clustering. This will basically ease the work of secret agents of our country and will help in rescue operations or any related stuffs.

• We would be requiring just one button (eg: button on a pen) for generating a morse code, the main problem is conversion, which this project can solve. Basically, it can solve both the challenges described above within seconds but limitation is:

• Input morse should have at least 2 words.

Key Contributions of the Work:

So, the entire project has been converted into 2 sections:

A. Taking Morse input, converting it into required format (csv):

We have taken input from one button using keyboard. Then we have converted into csv which has 2 columns X and Y. ith row of X column – stores time gaps between inputs i and i-1. ith row of Y column – stores duration of input. So, these two columns are enough for decoding.

B. Decoding the csv into actual text and then into audio:

Is converted into understandable text first by applying Unsupervised learning concept i.e., Kmeans clustering on csv i.e., on X Y. Applying Kmeans clustering on Y part of csv will help us understand what each input corresponds to (Dot or Dash). Applying Kmeans clustering on X part of csv will help us understand when does the letter end, word end.
This is just Key Idea, Detailed description has been provided in Methodology section.

II. Literature Survey

References we used for our project:
Idea of how to use Kmeans algorithm in Morse recognition has been taken from a journal paper done by students of Naval University of Engineering, Wuhan, China. This is existing paper. But it has drawbacks that it didnt explain how exactly categorisation must happen when we take inputs.
So, Project's approach with existing resources is self-made.
In our methodology, clustering algorithm has been modified to our pen type input. And basic method has limited scope of just numbers, we extended it to all symbols, alphabets, special codes.etc.
For Input Mechanism:
Click here for Ref1
Click here for Ref2
Click here for Ref3
For Kmeans:
Click here for Ref1
Click here for Ref2

III. Problem Statement

" Conversion of Morse signal into Audiosignal using Machine Learning concept i.e., Kmeans clustering "

A. Objectives:

• To transmit Morse code using a single button.
• To decode the incoming Morse code into text.
• To convert the decoded text into Audio.

B. Solved using:

• By using time, keyboard libraries input mechanism is built up
• Kmeans clustering of Unsupervised learning which is a part of Machine Learning for conversion of incoming Morse code into text.
• gTTs to convert decoded text into Audio.

IV. Methodology

For simulation and demonstration purposes, exporting and importing csv has been done. But practically, just the button on pen can send radio signal to stations which can be converted into csv (as described below) and can be decoded into audio (as described below).
As discussed earlier in Key Contributions, the project is basically split into 2 sections:

A. Taking input from a single button and convert it into Morse code:

This is basically done by Shashi Prakash and Naveen.

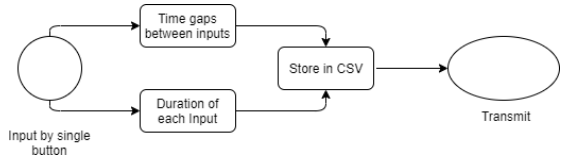

Fig. 1. Architectural Diagram for Input Mechanism

Above architectural diagram describes how this objective is achieved. So, Input is categorised into 2 columns: X (which stores info of time gaps between inputs), Y (which stores duration of input).
We thought of this idea because:
• We will require time gaps for finalising where each discrete input ends, where the letter ends, where the word ends.
• We will require duration of each input(time for which button is pressed at a time) for getting to know whether it is a dot or a dash (elementary particles of a Morse code).
For achieving this objective, we used the following libraries:
• Time: To find time duration time gap between the inputs.
• Keyboard: To recognise when was the key pressed and when it's not.
• Pandas and csv: To organise the input into a csv in a systematic way.
Suppose, we the input is "NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA". CSV is like:

X	Y
0.439437	0.364921
1.214664	0.349808
0.160359	0.341395
0.333286	0.396302
0.930155	0.45638
0.116044	0.106879
0.154803	0.426901
1.308877	0.153776
0.100278	0.069934
0.107978	0.078022
1.12941	0.089816
2.598788	0.060222
0.254822	0.381522
0.177349	0.085594
0.225862	0.402255
1.333611	0.086671
0.229004	0.323673
0.284354	0.418517
1.328338	0.521349

0.280457	0.439203
1.605461	0.122652
0.077557	0.099557
0.243219	0.369079
1.266167	0.086739
1.775321	0.094356
0.278513	0.371013
0.132288	0.119728
0.231019	0.400273
1.14711	0.139784
0.945635	0.336089
0.065607	0.107623
1.484257	0.141302
0.265472	0.417716
0.244237	0.444395
1.356375	0.427457
0.30393	0.452026
0.09292	0.09292

Fig. 2. CSV file of NITK

B. Decoding the csv into actual text and to audio:

This is basically done by Rishik and Praneeth.

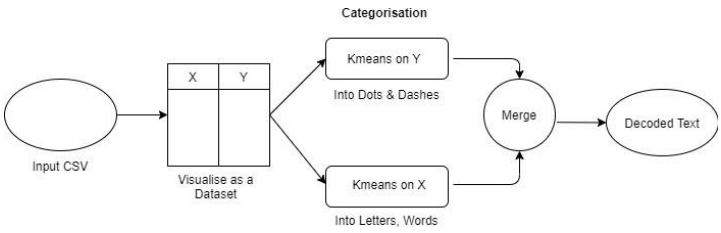

Fig. 3. Architectural Diagram for conversion into text

Above architectural diagram describes how this objective is achieved. Now, the csv must be converted into words and sentences. For this we need classification of each column like:

• X must be decoded to understand when each input ends, when letter ends, when word ends. So, we need to categorise this into 3 clusters (Red, Green, Blue as shown in code pdf). This was done using Machine Learning concept "Kmeans clustering".

• Y must be decoded to understand what each input is: whether it is a dot or a dash. So, we need to categorise this into 2 clusters (Red, Blue as shown in code pdf). This was also done using Kmeans clustering.

Reason for using Kmeans:

We basically need to segregate inputs into groups. But we never know how many groups (since it depends on input) so we used above procedure which can decode any huge input into constant no. of clusters. In, Morse, each category corresponds to one letter so basically categorisation is necessary because each group of inputs must map to unique combination of dots dashes in Morse.

Dots in input are decoded to 0s and Dashes in input are decoded into 1s (say Ylabels). Time gaps have been decoded into 0s, 1s ,2s (say Xlabels).

So lets take "NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA".

	X_labels	Y_labels
0	2	0
1	0	1
2	2	1
3	0	0
4	0	0
..	...	...
85	2	0
86	2	1
87	0	0
88	2	1
89	1	0

[90 rows x 2 columns]

Fig. 4. Data after processing

So, this is how we have decoded the csv by using Kmeans clustering. It has 90 rows. So, agent has pressed the button 90 times to transmit this. Even in real scenarios this happens. But we simplified and automated this by using ML.

By above method, X and Y are classified into dots dashes, letters, words. After clustering, we need to decode into text:

• By iterating through Xlabels and Ylabels, we can understand where letter ends so we can compare the decoded string of 0s and 1s with morsedict (dictionary which we defined that stores combination of 0s and 1s with each letter to which it corresponds to). We can understand where the word ends, thereby appending a space to the sentence. So, by iterating through this, we basically decoded the entire input.

Now, we need to convert the above resulted text into an audio.

This was done by using:

• gTTs module for conversion of text to mp3.

• MP3 to Audio by using PyDub.

V. Results and Analysis

The above discussed example (NITK expansion) has been demonstrated using a video, python notebook pdf which is available at Drive Link (Click Here)

After Kmeans Clustering of X and Y columns:

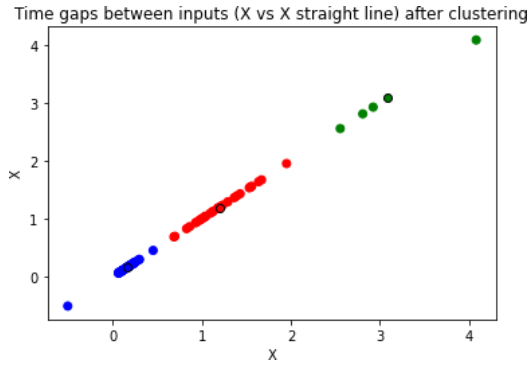

Fig. 5. X vs X graph after clustering

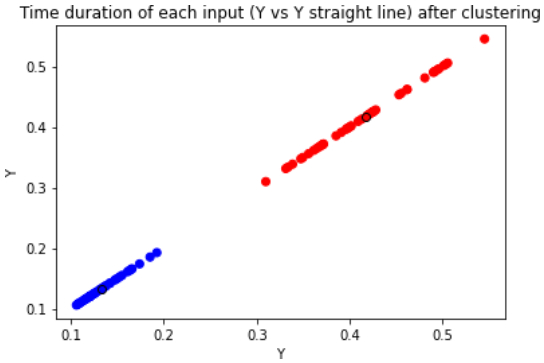

Fig. 6. Y vs Y graph after clustering

Both the axes have been same column for better visualisation of how exactly the clustering is done by Kmeans.

Final Result / Output:

```
[20] #Decoded Text
     print(word)

     NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
```

Fig. 7. Decoded into Text

This is text output where it decoded above shown csv into "National Institute of Technology Karnataka"

```
audio=Audio('output.wav')
audio
```

Fig. 8. Decoded into Audio

This is final audio which pronounces National Institute of Technology Karnataka.

Time Complexity Analysis

```
Time taken for converting Morse to Text: 0.8933380000000002 sec
Time taken for converting Text to Audio: 0.23341999999999974 sec
Total time of Execution (Morse to Audio): 1.126758 sec
```

Fig. 9. Time of Execution

This is execution time for entire conversion of Morse code (csv) into Text, Audio. It required more time for converting Morse(csv) to Text than Text to Audio. This is reverse if less words are there in the input. For conversion of text to Audio, gTTs follows a standard procedure but while converting large inputs into Text, relatively it takes more time because clustering involves many loops iterations.

For various inputs it worked well with almost full accuracy but the place where there is limitation is when input is just one word. Remedy for this is for each pen (which each agent uses) can have a default input of a word maybe of his name. So, that Kmeans will have at least 2 words. So, result would be better.

In real world, some recruited people are always waiting for input and they take time to decode it since it is manual conversion may add human errors too. This project would make the work faster, better and easier.

VI. Conclusion

The idea which we had was not implemented anywhere. So, constant discussions on how to work on made us develop an idea divide the work out of us to get the best out of us. After coming to a conclusion on how to work on, we designed the architectural diagram which came handy of what is the next goal we have.

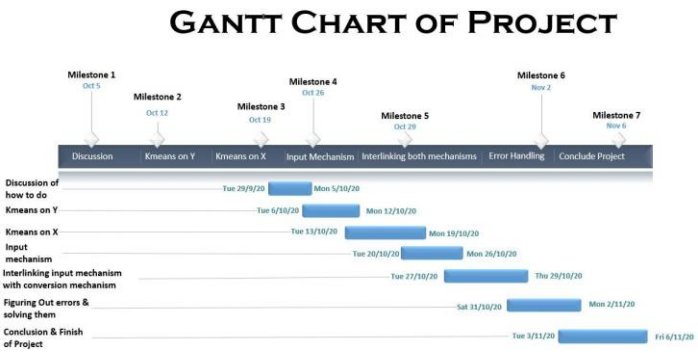

Fig. 10. Decoded into Audio

Induvidual Contribution

Milestone 1: All team members
Milestone 2: Rishik and Praneeth
Milestone 3: Rishik and Praneeth
Milestone 4: Shashi Prakash and Naveen
Milestone 5: All team members
Milestone 6: All team members
Milestone 7: All team members

VII. References

[1] Qu, Shanhu,Liu Hongbo, Zhang Xu IEEE journal on Morse Recognition Algorithm based on Kmeans. Naval University of Engineering, Wuhan, China.