BORDEAUX INP Enseirb-Matmeca

An isolated word speech to British Sign Language translator

Project report

Authors :
Sallmone Armela
Mony Alexandra
Kegl-Tourneix Noémie

Professor :
Lorblanchès Thomas

April 2023

Contents

1 Introduction (Armela)

Hard-of-hearing individuals may struggle to communicate with those who do not understand British Sign Language (BSL). One solution is to video call someone who can read BSL. This person will translate the BSL to speech in one direction and speech to BSL in the other direction. Our project focuses on developing a system that detects spoken words and translates them into BSL pictures. This system works like a supervised classifier trained using examples of spoken words. Then, it is used on new speakers by comparing sound features to recognize the spoken word. When a spoken word is recognized, a corresponding sign language picture or video is displayed for the hard-of-hearing person to understand. By analyzing four signals: 'one1', 'one2', 'two1', and 'two2', we will first develop general tools for time and frequency signal analysis, and then use these tools to extract features for the classifier to distinguish between words. In conclusion, we will train and test the classifier.

2 Speech signal analysis (Noémie)

In this section, the techniques used to help us understand what happens when a word is pronounced are studied.

The first aspect is to understand the difference between a voiced and an unvoiced word. For example, a vowel is voiced because the shape of the signal is periodic. It is due to the vocal cords that are used to produce this sound. An unvoiced signal is noisy, the shape doesn't look periodic. An example of an unvoiced signal is "ch", there is no sound produced by the vocal cord to have this sound.

The second aspect is the study of the Short Term Fourier Transform and Spectrogram. The Fourier Transform shows how the sound evolved with the frequency and the Spectrogram adds a dimension of time. This will help us to differenciate the different words.

2.1 Time analysis

In this section, the voiced and unvoiced parameters of the sentence "Hello word!" are studied.

To do so, the sound of someone saying the sentence is loaded and the signal in each voiced and unvoiced parts is restricted.

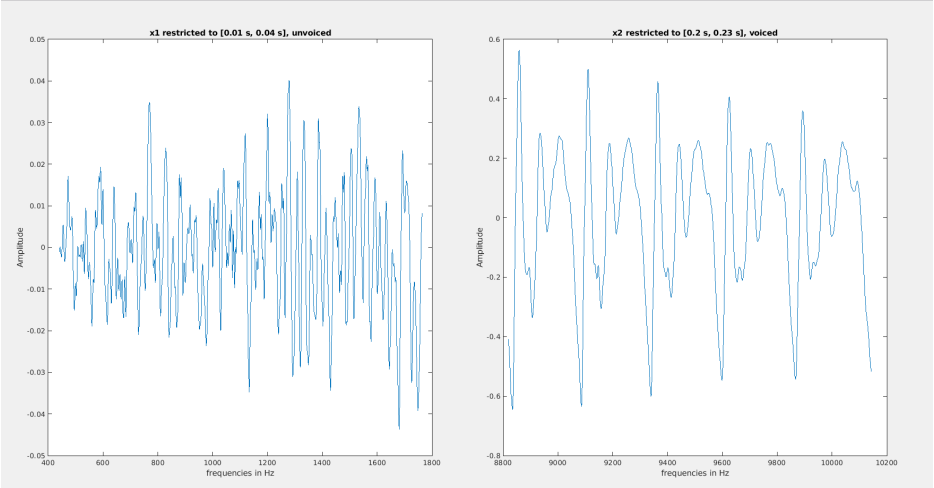


Figure 1: Hello restricted to [0.01s, 0.04s] and [0.2s, 0.23s]

Let us consider the signals on Figure 1. The signal on the left represents the x1 signal, an unvoiced signal. The signal is not periodic. It represents the phoneme "HH" at the beginning of "Hello". The sound is made with the air of the speaking person, the vocal cords are not used. On the contrary, the signal on the right represents the x2 signal, a voiced signal. The sound is made by using the vocal cords, so it is periodic on the figure.

Now, the entire word "Hello" is studied.

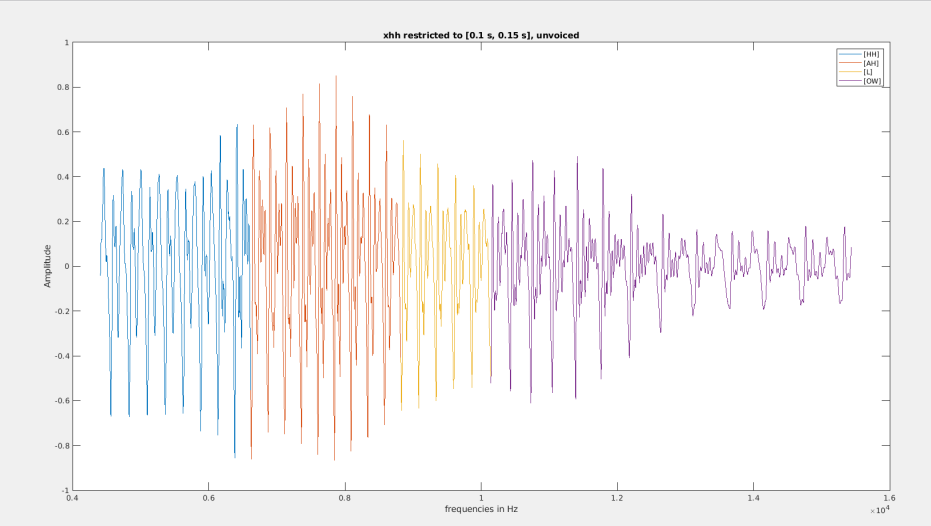
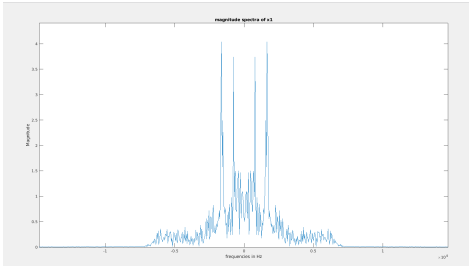


Figure 2: "Hello" restricted to [0.01s, 0.04s] and [0.2s, 0.23s]

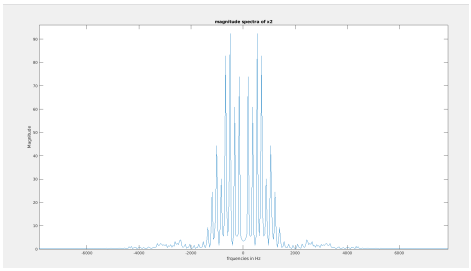
On Figure 2, the different phonemes are separated.
- The first one (blue), represents the "HH" phoneme, it is an unvoiced word: unperiodic signal.
- The second one (red), represents the "AH" phoneme, it is a voiced word: periodic.
- The third one (yellow), represents the "L" phoneme, it is a unvoiced word: unperiodic.
- The fourth one (violet), represents the "OW" phoneme, it is a voiced word: periodic.

2.2 Frequency analysis

Now, the magnitude spectra of the signals is studied, using the fft function of matlab.



(a) magnitude spectra of x1



(b) magnitude spectra of x2

Figure 3: Magnitude spectra of the signals x1 and x2

Figure 3a represents the magnitude spectra of a an unvoiced signal. Due to its unvoiced characteristic, the signal does not have a particular frequency component (it is not periodic). This signal is close to a white noise. The signal is supposed to have a uniform appearance on the spectrum. The peaks represented on the figure are potentially made by residue of the voice that is not perfectly like a white noise.

Figure 3b represents the magnitude spectrum of a voiced signal. The voiced characteristic is due to the vibration of the vocal cords. This vibration has fundamental and harmonics frequencies that create the sound. Those frequencies are the one represented on the spectrum.

2.3 Short Term Fourier Transform and Spectrogram

Now the Short Term Fourier Transform and Spectrogram is studied.
The signal has periodic variations. A window is used to work with those variations. Because of the periodic aspects of those variations, the window can be multiplied equal parts of the signal.
Then, a spectrogram is plot, that represents the modulus squared of the STFT.

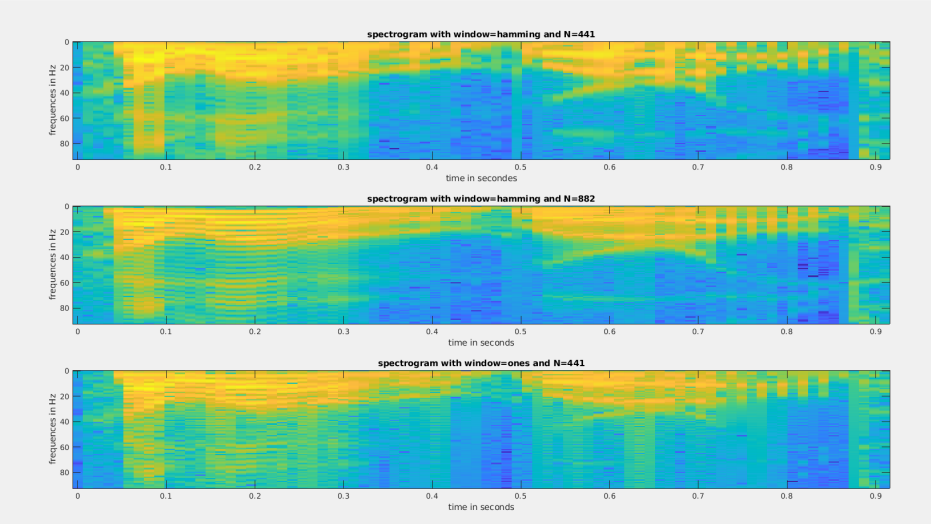


Figure 4: Spectrograms

On Figure 4, the spectrograms are represented with different values of:

- N: the length of the cut parts of the signal
- w: The window, *hamming* or *ones*

The more N is important, the more precise the graphics are. That is logical because, for example, instead of having 441 square, the double is used.

The *ones* window gives us a more precise spectrogram in a temporal way and the *hamming* window gives us a more precise spectrogram in a frequency way.
The *ones* window has a simple rectangular form so it provides precise temporal localization, the rapid changes in the signal are easily located.
On the other hand, the *hamming* window has a curve shape and it sacrifices the precision at the time level to have a better frequency precision. Thanks to this window, the close spaced frequency components are easily seen.

3 Feature extraction (Armela)

This section focuses on the extraction of features from a 30ms speech signal
$x_i = (x_i[0], x_i[1]...x_i[N-1])$. These features include the voiced/unvoiced flag, pitch, and Mel Frequency Cepstrum Coefficients (MFCC).

3.1 Voiced/Voicless flag and pitch

The voiced/unvoiced flag and pitch computation for each frame of signals one1, one2, two1, and two2 was carried out using the function called isvoiced.m. The autocorrelation of the speech signal x_i was used to determine the voiced/unvoiced status and pitch information. By applying isvoiced.m to the respective signals, the necessary features related to voicing and pitch were extracted.

Auto-correlation of $x_i[m]$ using the following unbiased estimator:

$$\gamma_u[p] = \begin{cases} \frac{1}{N-p} \sum_{k=p}^{N-1} x_i[k] x_i^[k-p] & \text{for } p = 0, 1, \ldots, N-1, \\ \gamma_u[-p] & \text{for } p = -1, \ldots, -N+1. \end{cases} \quad (1)$$

The pitch features for each frame of the four signals were computed by using the *isvoiced.m* function. The pitch features are one of the key features used by the classifier to determine which word was spoken. Therefore, the pitch features extracted from the signals one1, one2, two1, and two2 will be crucial in training and testing the classifier.

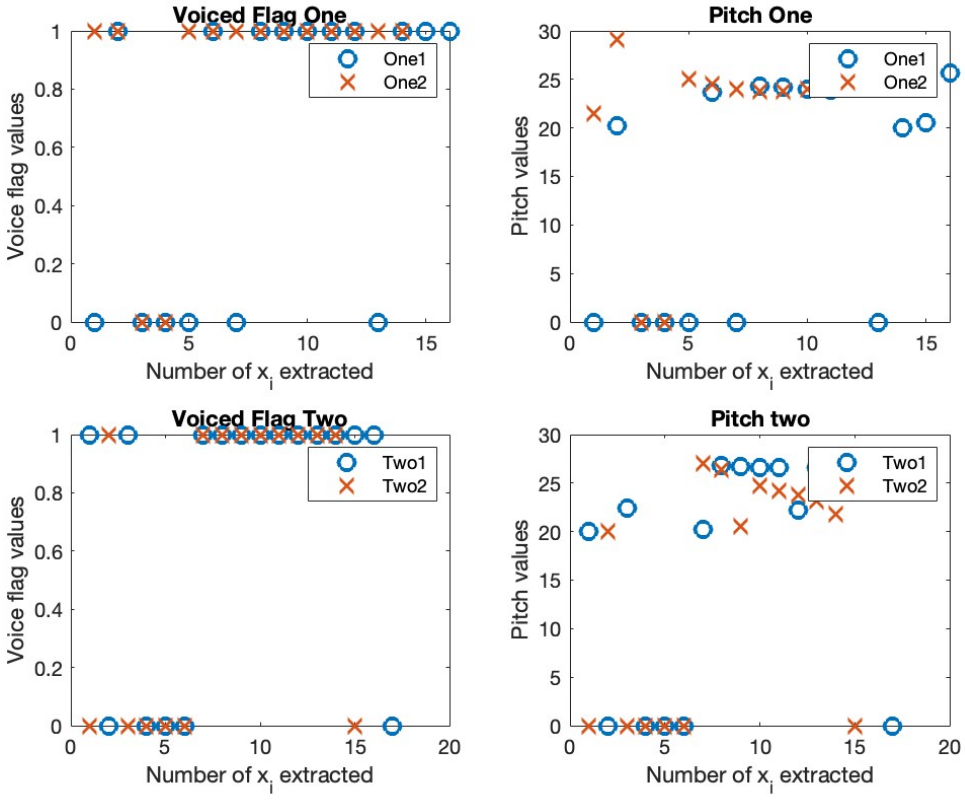


Figure 5: The pitch features for each frame of the signals one1,one2,two1,two 2

3.2 MFFC

Mel-Frequency Ceptrum Coeficients (MFCC) are the cepstral parameters mostly employed for speech classification and recognition. The main idea behin MFCC is to represent a frequency spectrum in a way that is more relevant to how humans perceive sound. To compute the MFCC features we followed these steps:

- Divided the signal into frames of length N and shift each frame by id.
- Applied the Hamming window to each frame
- Computed the magnitude of the Fourier transform for each frame.
- Applied a filterbank of P triangular filter to the magnitude spectrum.
- Computed MFCC coefficients as the following Discrete Cosine Transform (DCT):

$$mfcc_i = \sqrt{\frac{2}{P}} \sum_{j=p}^{P} \log(E_j) \cos(\frac{\pi}{P} i(j - 0.5))$$

The figure demonstrates a significant resemblance between the sounds one1 and one2, and the same principle applies to signals two1 and two2. There is a clear difference between "one" and "two" groups.

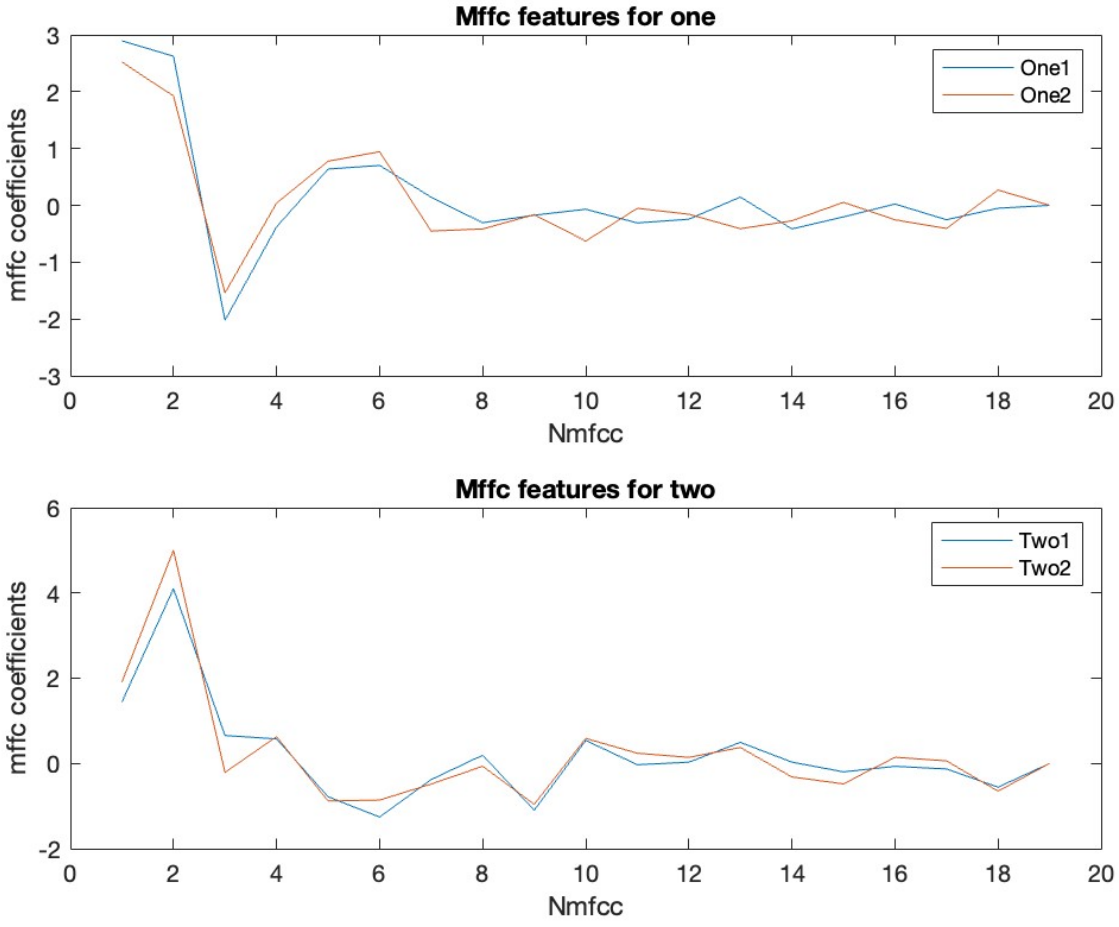


Figure 6: MFCC features for the signals one1, one2, two1, and two2

However the performance of the classifier may depend on the quality and diversity of the training data used to train it.

4 Classification (Alexandra)

This section focuses on recognizing isolated spoken words through feature extraction from the classifier. The goal is to apply the K-Nearest Neighbors (KNN) algorithm for speech signal classification using Mel-Frequency Cepstral Coefficients (MFCC) as extracted features. The classification process involves two main steps: training and testing.

4.1 Train the Classifier

During the training step, a known dataset called the training set was used to learn the model parameters. Features are extracted for each signal in the training set, specifically focusing on the last 11 MFCC coefficients out of a total of 12. These features are stored in a matrix called matF.

A function called **train_classifier** was implemented to address the task of training the classifier. This function takes two inputs: **data**, which represents the dataset, and **train**, which corresponds to the training set. The purpose of this function is to extract the necessary features from each element in the training set and store them in a matrix **matF**.

By utilizing the **train_classifier** function and passing the appropriate **data** and **train** inputs, the model corresponding to the elements in **train_1** and **train_2** can be identified. The resulting **matF** matrices provide a representation of the extracted features for each training example, which can be further utilized for classification purposes.

4.2 Test the Classifier

In the testing step, the performance of the classifier was evaluated using a separate dataset known as the test set. The test set consists of a set of signals, denoted as x, for which the corresponding classes need to be determined. The classifier accomplishes this by comparing the extracted features, represented as **f**, of each signal **x** in the test set with the features **f'** extracted from the signals **x'** in the training set.

To measure the similarity between the features f and f', the Euclidean distance metric, denoted as d(**f**, **f'**), was employed. This distance is calculated as the Euclidean norm of the difference between f and f', which is defined as $d(\mathbf{f}, \mathbf{f'}) = ||\mathbf{f} - \mathbf{f'}||_2$, where $||\mathbf{f}||_2$ represents the Euclidean norm given by $||\mathbf{f}||_2 = \sqrt{\sum_k |\mathbf{f}_k|^2}$.

4.2.1 K-Nearest Neighbors (KNN) Algorithm

The KNN algorithm was utilized to classify the signals in the test set. The following summary outlines the KNN algorithm:

a. For each signal **x'** in the training set, the distance d(**f**, **f'**) was computed to measure the distance between the test features and the training features of the training signal.

b. The K nearest neighbors of the test signal's features **f** in the training set were selected, forming the set KN(**f**) that contains the K nearest neighbors.

c. The number of elements in KN(**f**) that belong to the same class as the signal **x'** was counted.

d. The class label of the test signal **x** was determined based on the most represented class within the set KN(**f**).

4.2.2 Evaluation of the Classifier

In the evaluation phase, the classifier was tested using the provided test sets: **test_1** and **test_2**. The KNN algorithm was applied to estimate the class of each element x in the test sets, and the accuracy of the classification was measured.

The probability of success for classification was computed by dividing the number of correctly classified elements in the test set by the total number of elements. The results obtained for the classification success probabilities in the test sets are promising. In **test_1**, we achieved a success probability of 0.7391, indicating that our classifier correctly classified approximately **73.91%** of the elements in this set. Similarly, in **test_2**, we obtained a success probability of 0.8101, implying that our classifier achieved accurate classification for around **81.01%** of the elements in this set.

These results highlight the effectiveness and generalization capabilities of the classification model based on the K-Nearest Neighbors (KNN) algorithm. It is important to note that the choice of parameter k played a significant role in maximizing the probability of success. Specifically, for **test_1**, k was set to 2, while for **test_2**, k was set to 3 to achieve the highest possible probability. These findings underscore the potential of the classifier to accurately predict signal classes based on their extracted features. However, it is important to consider that the performance levels may vary depending on the specific problem and dataset characteristics.

The performance of the classifier can be further analyzed by computing the confusion matrix Cm using the test sets **test_1** and **test_2**. Each element Cm(i, j) represents the probability that an element x is classified in class j given that it belongs to class i. The confusion matrix provides detailed information about the classification results for each class and allows for a comprehensive evaluation of the classifier's performance.

Upon analyzing the matrix, it can be observed that the diagonal elements have relatively high values, suggesting accurate predictions for the corresponding classes. This indicates that accurate identification of instances belonging to these classes is achieved by the classifier.

However, non-zero values off the diagonal are present, indicating misclassifications. These values represent the probabilities of instances being classified into incorrect classes. These misclassification patterns and areas of improvement can be further investigated.

Overall, the confusion matrix provides a comprehensive overview of the classifier's performance across different classes, allowing for evaluation of its strengths and weaknesses. By considering this information, valuable insights can be gained to enhance the classification model.

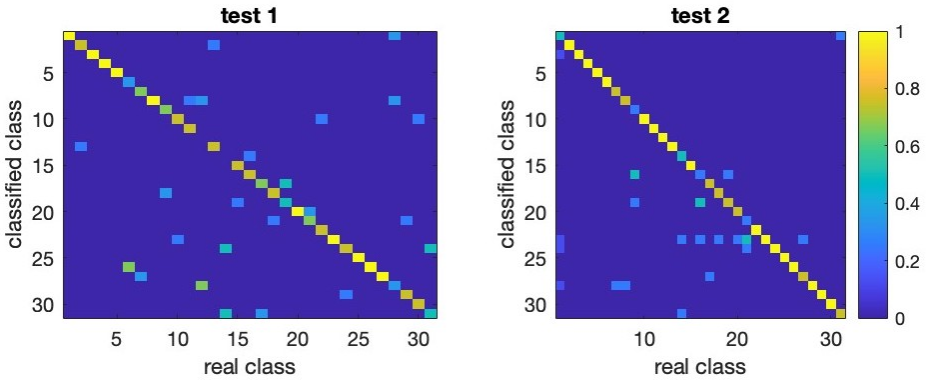


Figure 7: Confusion matrix with success probaility

To compute the probability of success (P) from the confusion matrix,

$$P(\text{success}) = \frac{\sum(\text{diagonal elements of the confusion matrix})}{\text{Total number of elements in the test set}} \tag{2}$$

By summing the diagonal elements of the confusion matrix, the total number of correctly classified elements is obtained. Dividing this sum by the total number of elements in the test set gives the probability of success, which indicates the rate of correct classification.

This measure provides valuable insights into the performance of the classifier in accurately classifying the elements, without explicitly mentioning the active agent.

In this section, the classifier was trained using the training sets, and the KNN algorithm was utilized for classifying isolated spoken words. Features were extracted using MFCC coefficients, and the performance was evaluated using the test sets. The probability of success and the confusion matrix provided insights into the accuracy of the classifier. Analysis and interpretation of the results can be conducted for a comprehensive understanding.

5 Improved versions of the classifier (Alexandra)

In order to enhance the classifier's performance, the addition of personal voice data was necessary. By incorporating the individual's voice into the training process, the classifier becomes more attuned to their specific speech characteristics, accent, and pronunciation. This allows for a more accurate recognition of spoken words by the classifier when compared to a generic model trained on diverse voices. The inclusion of personal voice data helps to personalize the classification model and improve its ability to accurately classify spoken words.