Voice Recognition through Machine Learing

Raktim Gautam Goswami[1], Abhishek Bairagi[2] & G V V Sharma[3]

Contents

***Abstract*—This manual shows how to develop a voice recognition algorithm and use it to control a toycar.**

1 Dataset

1.1 Draw the block diagram of the AI-ML system for the toycar.
Solution: See Fig. 1.1

1.2 Record 'forward' 80 times using you phone and save as 'forwardi.wav' for $i = 1, \ldots, 80$. The recording duration should be between 1-3 seconds.

1.3 Repeat by recording 'left', 'right', 'back' and 'stop'. Make sure that the audio files for each command are in separate directories. Download the following directory for reference

```
svn checkout https://github.com/gadepall/
    EE1390/trunk/AI−ML/audio_dataset
```

1.4 Use the following script to generate a dataset for 'back' command. Explain through a block diagram.

```
https://raw.githubusercontent.com/gadepall/
    EE1390/master/AI−ML/codes/250files.py
```

Solution: The datasets are generated through zero padding. The diagram in Fig. 1.4 explains how this is done for the back command.

1.5 Suitably modify the above script to generate similar datasets for 'left', 'right', 'stop' and 'forward'.

1.6 Summarize the datasets generated through a table.
Solution: See Table 1.6

2 Linear Regression: Least Squares

2.1 Draw the block diagram for the ML algorithm
Solution: See Fig. 2.1

2.2 List the reference vectors for all the voice commands.
Solution: See Table 2.2.

2.3 The sigmoid function is defined as

$$s(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

Sketch $s(x)$.
Solution: The following code plots $s(x)$ in Fig. 2.3.

2.4 Show that $0 < s(x) < 1$.
Solution: $s(x)$ is useful for transforming large values to a value between 0 and 1.

2.5 Formulate a regression model for the voice recognition system.
Solution: Let $\mathbf{x}$ be the voice command. The model used is

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b} \tag{2.2}$$

where $\mathbf{W}$ is the weight matrix and $\mathbf{b}$ is the bias vector. Ideally, the output $\mathbf{y}$ should be one of the reference vectors in Table 2.2.

2.6 Frame an optimization problem for estimating $\mathbf{W}$ and $\mathbf{b}$.
Solution: This is done by considering the cost

The authors are with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India . e-mail: 1. ee17btech11004@iith.ac.in, 2. ee17btech11051@iith.ac.in, 3. gadepall@iith.ac.in

Commands	Input	Output / Input file	Conditioned	Training	Testing
Back	80	250	20000	16000	4000
Forward	80	250	20000	16000	4000
Left	80	250	20000	16000	4000
Right	80	250	20000	16000	4000
Stop	80	250	20000	16000	4000
	Total		**100000**		

TABLE 1.6: File calculus

Command	Reference vector
Forward	$\begin{pmatrix}1 & 0 & 0 & 0 & 0\end{pmatrix}^T$
Back	$\begin{pmatrix}0 & 1 & 0 & 0 & 0\end{pmatrix}^T$
Left	$\begin{pmatrix}0 & 0 & 1 & 0 & 0\end{pmatrix}^T$
Right	$\begin{pmatrix}0 & 0 & 0 & 1 & 0\end{pmatrix}^T$
Stop	$\begin{pmatrix}0 & 0 & 0 & 0 & 1\end{pmatrix}^T$

TABLE 2.2: Reference vectors

function

$$\min_{\mathbf{W},\mathbf{b}} J(\mathbf{W},\mathbf{b}) = \frac{1}{2}\|\mathbf{y}-\hat{\mathbf{y}}\|^2 \tag{2.3}$$

3 Gradient Descent

3.1 Let $\mathbf{W}$ be 2×2 and $\mathbf{x},\mathbf{b}$ be 2×1. Show that

$$\frac{\partial\|\mathbf{W}\mathbf{x}+\mathbf{b}\|^2}{\partial\mathbf{W}} = 2\left(\mathbf{W}\mathbf{x}+\mathbf{b}\right)\mathbf{x}^T \tag{3.1}$$

$$\frac{\partial\|\mathbf{W}\mathbf{x}+\mathbf{x}\|^2}{\partial\mathbf{W}} = 2\left(\mathbf{W}\mathbf{x}+\mathbf{b}\right)\mathbf{x}^T \tag{3.2}$$

3.2 $\mathbf{W}$ and $\mathbf{b}$ can be estimated from (2.3) using

$$\mathbf{W}(n+1) = \mathbf{W}(n) - \frac{\alpha}{2}\frac{\partial J(\mathbf{W},\mathbf{b})}{\partial\mathbf{W}} \tag{3.3}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) - \frac{\alpha}{2}\frac{\partial J(\mathbf{W},\mathbf{b})}{\partial\mathbf{b}} \tag{3.4}$$

Show that (3.3) can be expressed as

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \alpha\left(\mathbf{y}-\hat{\mathbf{y}}\right)\mathbf{x}^T \tag{3.5}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) + \alpha\left(\mathbf{y}-\hat{\mathbf{y}}\right) \tag{3.6}$$

Solution: From (2.3) and (**??**),

$$J(\mathbf{W},\mathbf{b}) = \frac{1}{2}\|\mathbf{y}-\hat{\mathbf{y}}\|^2 \tag{3.7}$$

$$= \left(\mathbf{W}\mathbf{x}+\mathbf{b}-\mathbf{y}\right)^T\left(\mathbf{W}\mathbf{x}+\mathbf{b}-\mathbf{y}\right) \tag{3.8}$$

$$= \left(\mathbf{x}^T\mathbf{W}^T+\mathbf{b}^T-\mathbf{y}^T\right)\left(\mathbf{W}\mathbf{x}+\mathbf{b}-\mathbf{y}\right) \tag{3.9}$$

$$= \mathbf{W}^T\mathbf{x}^T\mathbf{x}\mathbf{W} + \mathbf{W}^T\mathbf{x}^T\mathbf{b} - \mathbf{W}^T\mathbf{x}^T\mathbf{y} \tag{3.10}$$

$$+ \mathbf{b}^T\mathbf{x}\mathbf{W} + \mathbf{b}^T\mathbf{b} - \mathbf{b}^T\mathbf{y} - \mathbf{y}^T\mathbf{x}\mathbf{W} \tag{3.11}$$

$$- \mathbf{y}^T\mathbf{b} + \mathbf{y}^T\mathbf{y} \tag{3.12}$$

Using

$$\frac{\partial}{\partial\mathbf{W}}\mathbf{W}^T\mathbf{x}^T\mathbf{x}\mathbf{W} = \tag{3.13}$$

3.3 Store the complete dataset in a directory and run **code.py** from within the directory. Note that this should be done on a powerful workstation. This will generate two files **W1.out** and **b.out**.

3.1 Python code

https://github.com/raktimgg/ML-algorithm-for-speech-recog This is the full code that is used for training. The accuracy we are getting is around 98 percent.

3.2 Dataset

We have made our own dataset by recording 25 samples of each word. Each of these samples are recreated by adding empty elements in the front and back in many different cobinations to create a dataset of 6250 samples for each word. All the audio files are imported to an array in the code and converted to mfcc format before training. For

creating training dataset we recorded 25 audio file of each of the following word -
1)Forward
2)Left
3)Right
4)Back
5)Stop
The code for generating 6250 samples for each word from 25 samples can be found in the github link attached.
https://github.com/abhishekbairagi/Making-Dataset-for-ML/t

4 Transfering the weights to Raspberry Pi (Yet to be done)

The weight(W1 and B) are saved in a file at the end of the code. These weights will be transferred to the raspberry pi and a simple program written, will record audio on the raspberry pi, do the calculations using the weights and predict the text output. This output will be sent,using bluetooth, to the toy car, which will move accordingly.

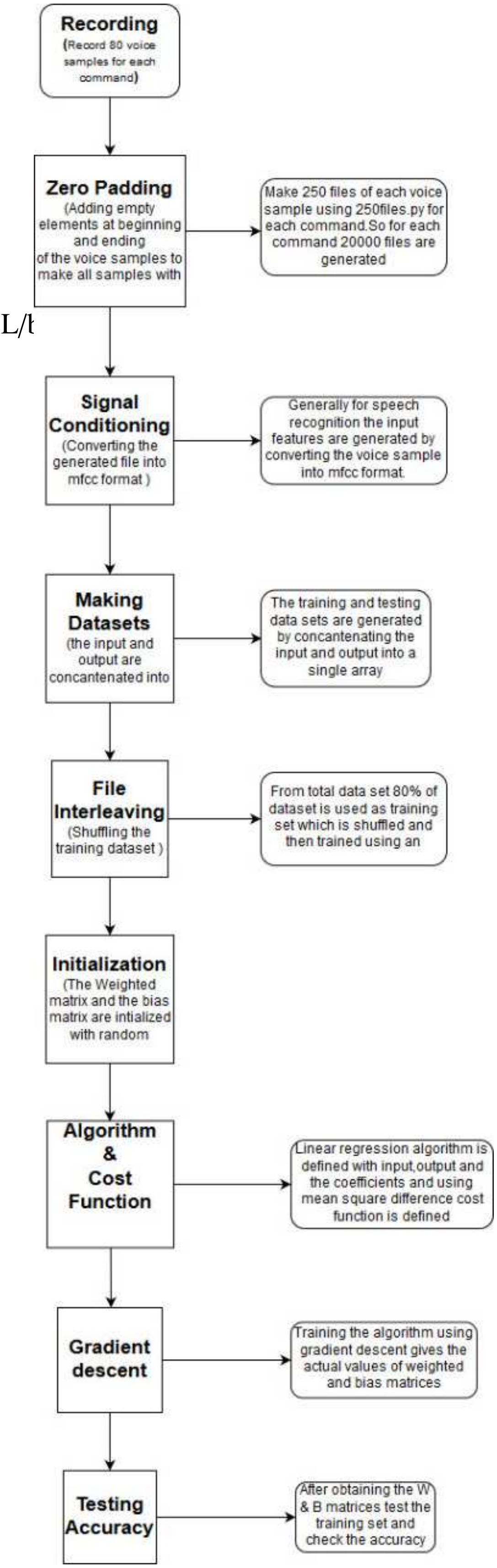
Recording
(Record 80 voice samples for each command)
Zero Padding
(Adding empty elements at beginning and ending of the voice samples to make all samples with
Make 250 files of each voice sample using 250files.py for each command.So for each command 20000 files are generated
Signal Conditioning
(Converting the generated file into mfcc format)
Generally for speech recognition the input features are generated by converting the voice sample into mfcc format.
Making Datasets
(the input and output are concantenated into
The training and testing data sets are generated by concantenating the input and output into a single array
File Interleaving
(Shuffling the training dataset)
From total data set 80% of dataset is used as training set which is shuffled and then trained using an
Initialization
(The Weighted matrix and the bias matrix are intialized with random
Algorithm & Cost Function
Linear regression algorithm is defined with input,output and the coefficients and using mean square difference cost function is defined
Gradient descent
Training the algorithm using gradient descent gives the actual values of weighted and bias matrices
Testing Accuracy
After obtaining the W & B matrices test the training set and check the accuracy

Fig. 1.1: ML System

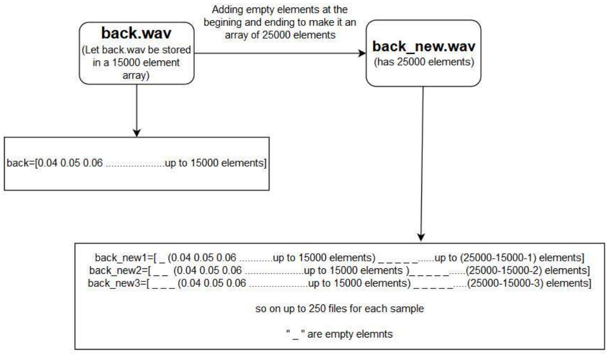
back.wav
(Let back.wav be stored in a 15000 element array)
Adding empty elements at the begining and ending to make it an array of 25000 elements
back_new.wav
(has 25000 elements)
back=[0.04 0.05 0.06 ....up to 15000 elements]
back_new1=[_ (0.04 0.05 0.06 ....up to 15000 elements) _ _ _ _ _......up to (25000-15000-1) elements]
back_new2=[_ _ (0.04 0.05 0.06 ....up to 15000 elements)_ _ _ _ _......(25000-15000-2) elements]
back_new3=[_ _ _ (0.04 0.05 0.06 ....up to 15000 elements) _ _ _ _ _.....(25000-15000-3) elements]
so on up to 250 files for each sample
" _ " are empty elemnts

Fig. 1.4: Zero padding

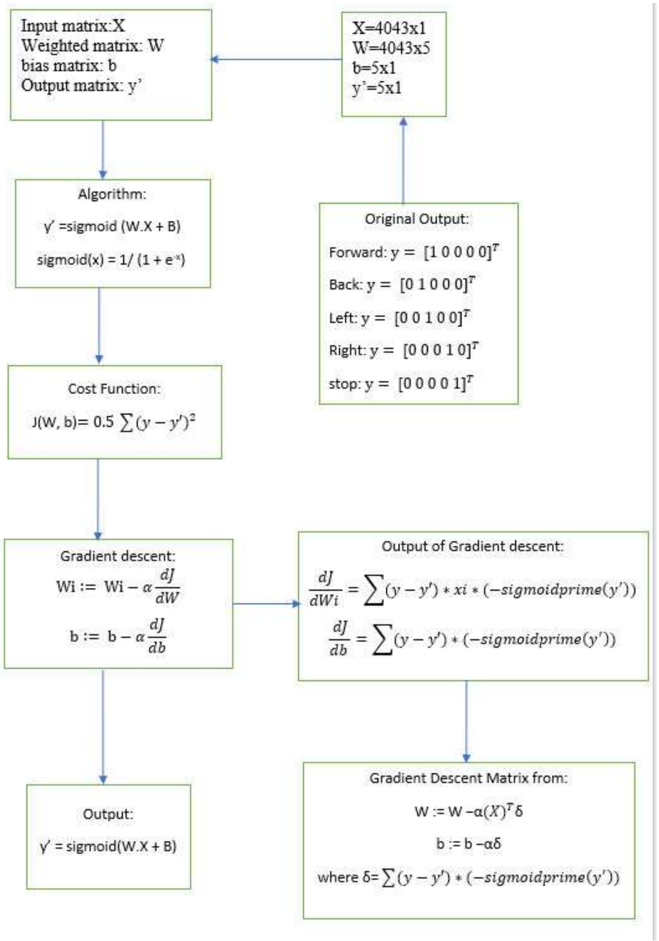

Fig. 2.1: Least squares and gradient descent

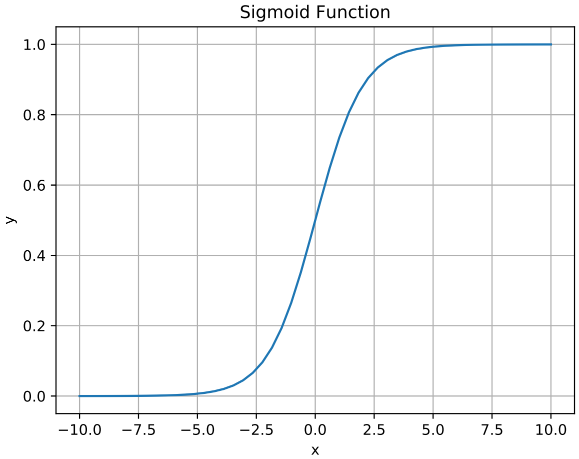

Fig. 2.3: Sigmoid function