भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology
Hyderabad

Voice Recognition through Machine Learing

Teaching Learning Center
IIT Hyderabad

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
```

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

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. **??**.

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

2.5 Consider $\mathbf{x}$ be 4043×1 to be human voice issuing either 'forward', 'left', 'right', 'back' and 'stop'. Let $\mathbf{W}$ be 4043×5 and $\mathbf{b}$ be 5×1. $\mathbf{W}$ and $\mathbf{b}$ are the machine parameters. Then the machine makes a decision based on

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

2.6 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**.

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

2.7 The problem is to estimate $\mathbf{W}$ and $\mathbf{b}$. This is done by considering

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

2.1 Solution: Gradient Descent

2.1.1 $\mathbf{W}$ and $\mathbf{b}$ can be estimated from (2.2) using

$$\mathbf{W}(n+1) = \mathbf{W}(n) - \frac{\alpha}{2}\frac{\partial J(\mathbf{W},\mathbf{b})}{\partial \mathbf{W}} \tag{2.4}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) - \frac{\alpha}{2}\frac{\partial J(\mathbf{W},\mathbf{b})}{\partial \mathbf{b}} \tag{2.5}$$

Show that (2.3) can be expressed as

$$\mathbf{W}(n+1) = \mathbf{W}(n) - \alpha\left[\mathbf{x}^T(n)\mathbf{x}(n)\mathbf{W}(n) + \mathbf{x}^T(n)\mathbf{b}(n) - \mathbf{x}^T(n)\mathbf{y}(n)\right] \tag{2.6}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) - \alpha\left[\mathbf{x}\mathbf{W} - \mathbf{b} - \mathbf{y}\right] \tag{2.7}$$

Solution: From (2.2) and (2.1),

$$J(\mathbf{W},\mathbf{b}) = \frac{1}{2}\|\mathbf{y}-\hat{\mathbf{y}}\|^2 \tag{2.8}$$

$$= (\mathbf{x}\mathbf{W}+\mathbf{b}-\mathbf{y})^T(\mathbf{x}\mathbf{W}+\mathbf{b}-\mathbf{y}) \tag{2.9}$$

$$= \left(\mathbf{W}^T\mathbf{x}^T + \mathbf{b}^T - \mathbf{y}^T\right)(\mathbf{x}\mathbf{W}+\mathbf{b}-\mathbf{y}) \tag{2.10}$$

$$= \mathbf{W}^T\mathbf{x}^T\mathbf{x}\mathbf{W} + \mathbf{W}^T\mathbf{x}^T\mathbf{b} - \mathbf{W}^T\mathbf{x}^T\mathbf{y} \tag{2.11}$$

$$+ \mathbf{b}^T\mathbf{x}\mathbf{W} + \mathbf{b}^T\mathbf{b} - \mathbf{b}^T\mathbf{y} - \mathbf{y}^T\mathbf{x}\mathbf{W} \tag{2.12}$$

$$- \mathbf{y}^T\mathbf{b} + \mathbf{y}^T\mathbf{y} \tag{2.13}$$

Using

$$\frac{\partial}{\partial \mathbf{W}}\mathbf{W}^T\mathbf{x}^T\mathbf{x}\mathbf{W} = \tag{2.14}$$

2.2 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.

2.3 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

3 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.

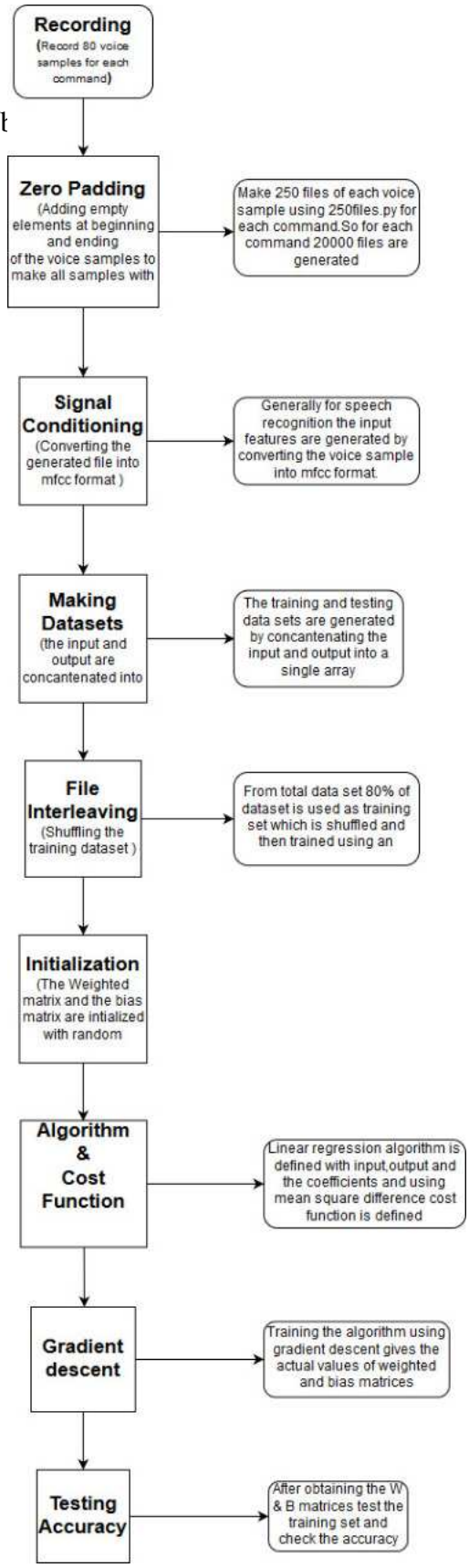

Fig. 1.1: ML System

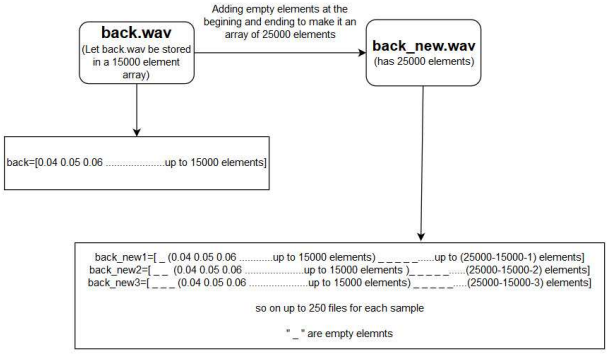

Fig. 1.4: Zero padding

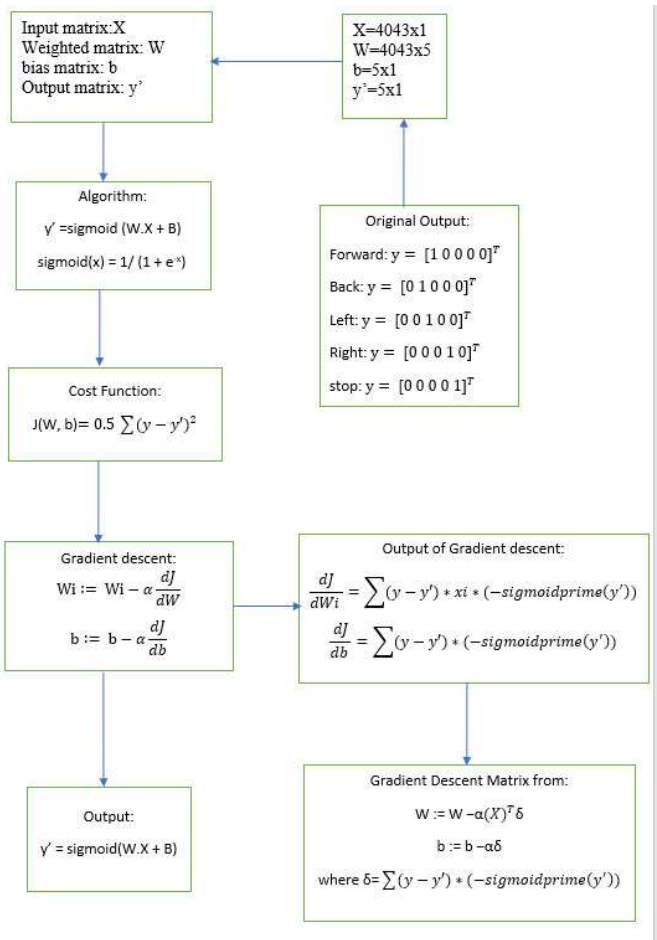

Fig. 2.1: Least squares and gradient descent

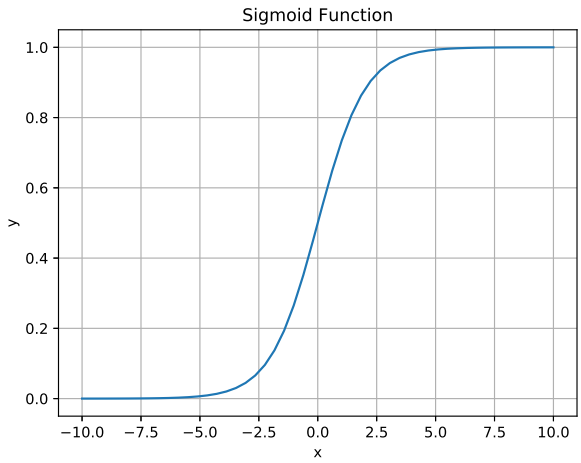

Fig. 2.3: Sigmoid function