Introduction to AI-ML
$\Rightarrow$ *VoiceControlledToycar*

Who? Sahukari Chaitanya Varun
EE19BTECH11040

When? January 21, 2020

The places where math is involved are

1. Padding of files
2. MFCC
3. The RNN Structure
4. The Sigmoid Function
5. Gradient Descent Method
6. LSTM

Zero Padding of Sound files

The necessity of padding is to generate more number of samples from given limited number of samples and each of uniform size.

$$(80 \text{ files})^{25 files.py} \rightarrow 2000 files. \qquad 20$$

This is done by adding a combination of zeroes before and after array of the sample.Then the overall length becomes 250 and returns back audio files to the harddisk.

Mel Frequency Cepstral Coefficient

This place the audio files are processed.The steps involved here are
1.Frame the signal into short frames.
2.For each frame calculate the periodogram estimate of the power spectrum.
3.Apply the mel filterbank to the power spectra, sum the energy in each filter.
4.Take the logarithm of all filterbank energies.
5.Take the DCT of the log filterbank energies.
6.Keep DCT coefficients 2-13, discard the rest.

Recurrent Neural Networks

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

$$\mathbf{y} = \mathbf{x}^T \mathbf{W} + \mathbf{b} \tag{4.1}$$

Squashing

In the above case ,due to addition of matrices, the elements can exceed 1 and for case of working , we limit the set to vary in [0,1]. So squashing of y is done with the help of sigmoid function.

$$\hat{\mathbf{y}} = 1 \div (1 + exp(-\mathbf{y})) \tag{5.1}$$

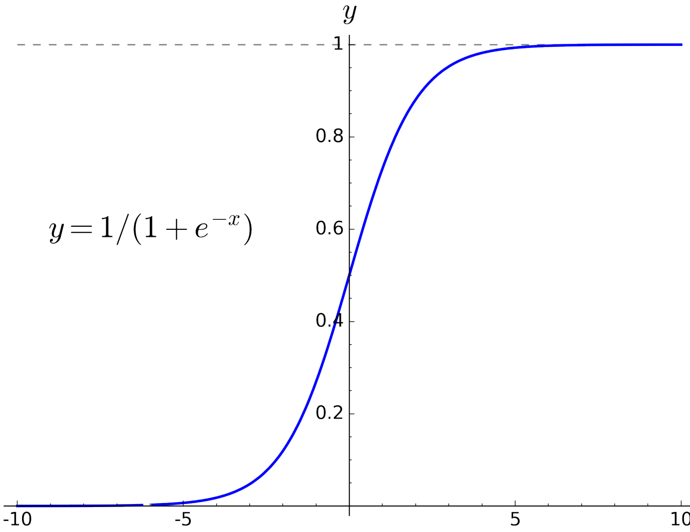

Figure: Sigmoid/Logistic curve

Error Function - Optimisation

Now the problem is to estimate **W** and **b** . This is done using the error or cost function defined as

$$L(\mathbf{W}, \mathbf{b}) = \frac{1}{2}\|\mathbf{y} - \hat{\mathbf{y}}\|^2 \tag{6.1}$$

We need to find the minimum of the error function for optimised results and is done using Gradient Descent Method.

Gradient Descent Method

$$\mathbf{W}(n+1) = \mathbf{W}(n) - \frac{\alpha}{2} \frac{\partial L\left(\mathbf{W}, \mathbf{b}\right)}{\partial \mathbf{W}} \mathbf{W}(n) \tag{6.2}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) - \frac{\alpha}{2} \frac{\partial L\left(\mathbf{W}, \mathbf{b}\right)}{\partial \mathbf{b}} \tag{6.3}$$

And this comes to be

$$\begin{aligned} \mathbf{W}(n+1) = \mathbf{W}(n) - \alpha \Big[& \mathbf{x}^T(n)\mathbf{x}(n)\mathbf{W}(n) \\ & + \mathbf{x}^T(n)\mathbf{b}(n) - \mathbf{x}^T(n)\mathbf{y}(n) \Big] \end{aligned} \tag{6.4}$$

$$\mathbf{b}(n+1) = \mathbf{b}(n) - \alpha \left[\mathbf{x}\mathbf{W} - \mathbf{b} - \mathbf{y} \right] \tag{6.5}$$

Long short-term memory

The traditional RNN cannot handle long term data much efficiently and long-range dependencies will worsen learning.LSTM is special RNN which has a special gate known as the forget gate, which makes the output from non related nodes nonsignificant. Thus the long range memories existwith out the gradients vanishing.

In a traditional LSTM with forget gates :
The initial values are $c_0 = 0$ and $h_0 = 0$,the operator $\circ$ denotes the Hadamard product (element-wise product) and subscript 't' indexes the time step.The functions are:

$$\mathbf{f}_t = \sigma_g(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f)$$

$$\mathbf{i}_t = \sigma_g(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i)$$

$$\mathbf{o}_t = \sigma_g(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o)$$

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \sigma_c(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c)$$

$$\mathbf{h}_t = \mathbf{o}_t \circ \sigma_h(\mathbf{c}_t)$$

Where the variables are :

$\mathbf{x}_t$: input vector

$\mathbf{h}_t$: output vector

$\mathbf{c}_t$: cell state vector

$\mathbf{W}, \mathbf{U}$ and $\mathbf{b}$: parameter matrices and vectors

$\mathbf{f}_t$: Forget gate vector(Weight of remembering old information)

$\mathbf{i}_t$: Input gate vector(Weight of aquiring new information)

$\mathbf{o}_t$: Output gate vector(Output candidate)

Activating functions

σ_g : Sigmoid Function

σ_c : Hyperbolic tangent Function

σ_h : x

Loss Function

The loss function in the LSTM is calculated as the categorical class entropy defined as follows

$$E = -\sum_{i}^{C} y_i log(\hat{y}_i)$$

where C is the total number of classes and $\hat{y}_i$ is score of each sample of y_i in the softmax function.

Softmax function: $f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}}$ 20