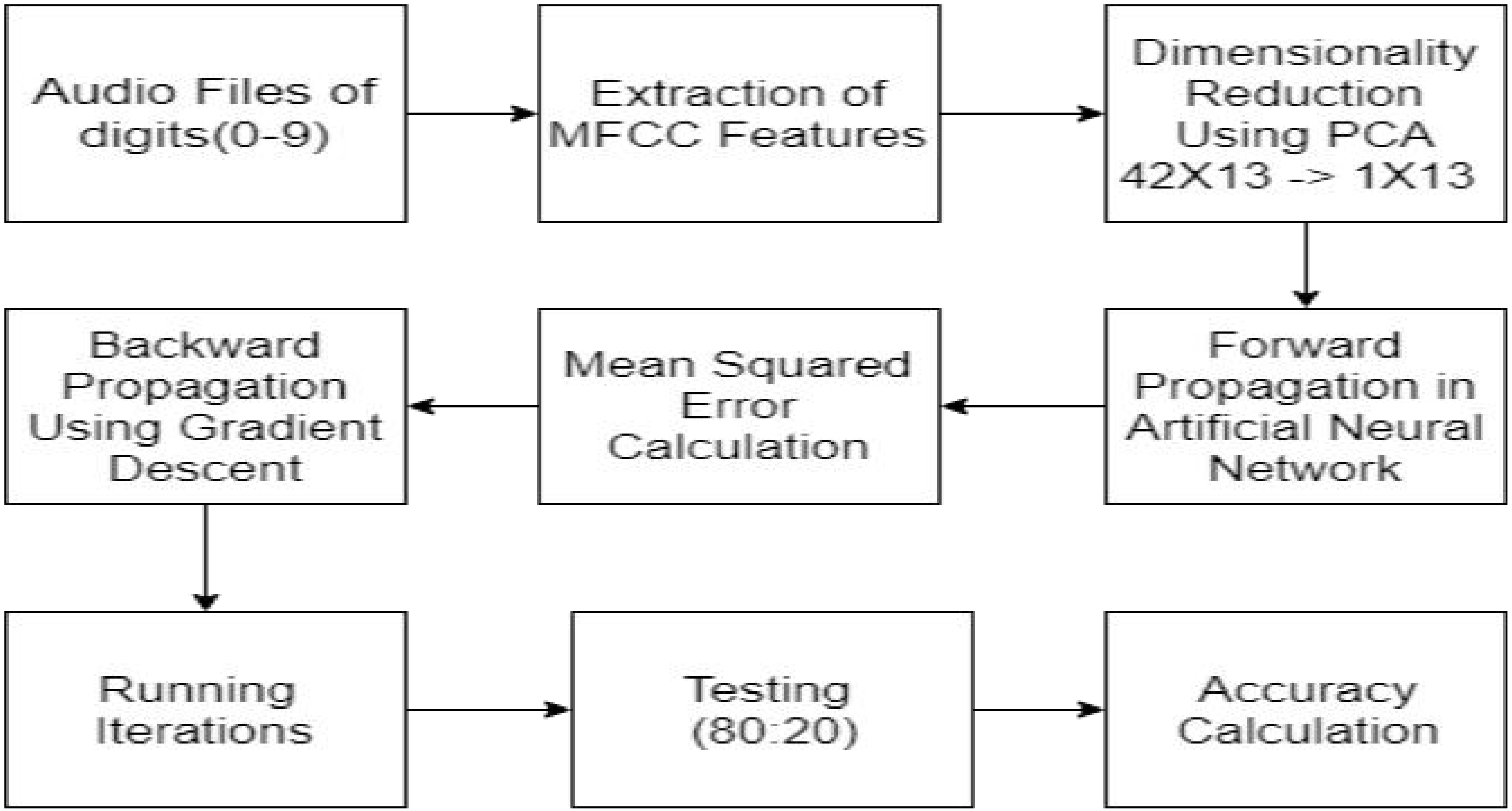
# FLOW

1. Block Diagram of Model- (ANN)
2. Dataset
3. MFCC Features Extraction
4. PCA
5. Forward Propagation
6. Backward Propogation using Gradient Descent
7. Results
8. Challenges
9. Timeline (workplan) of work for stage-3

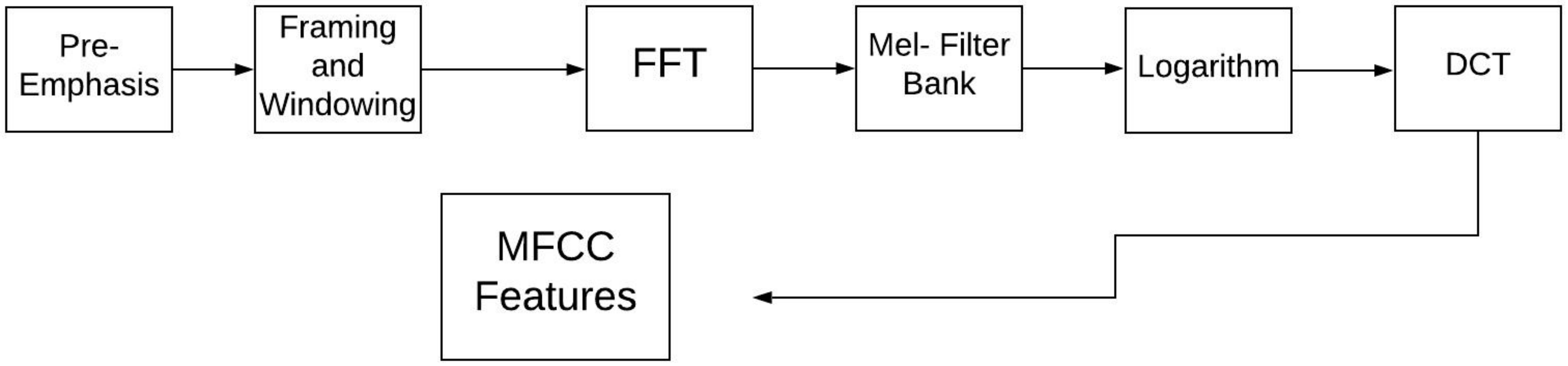
# Block Diagram of Model (ANN)



# Dataset

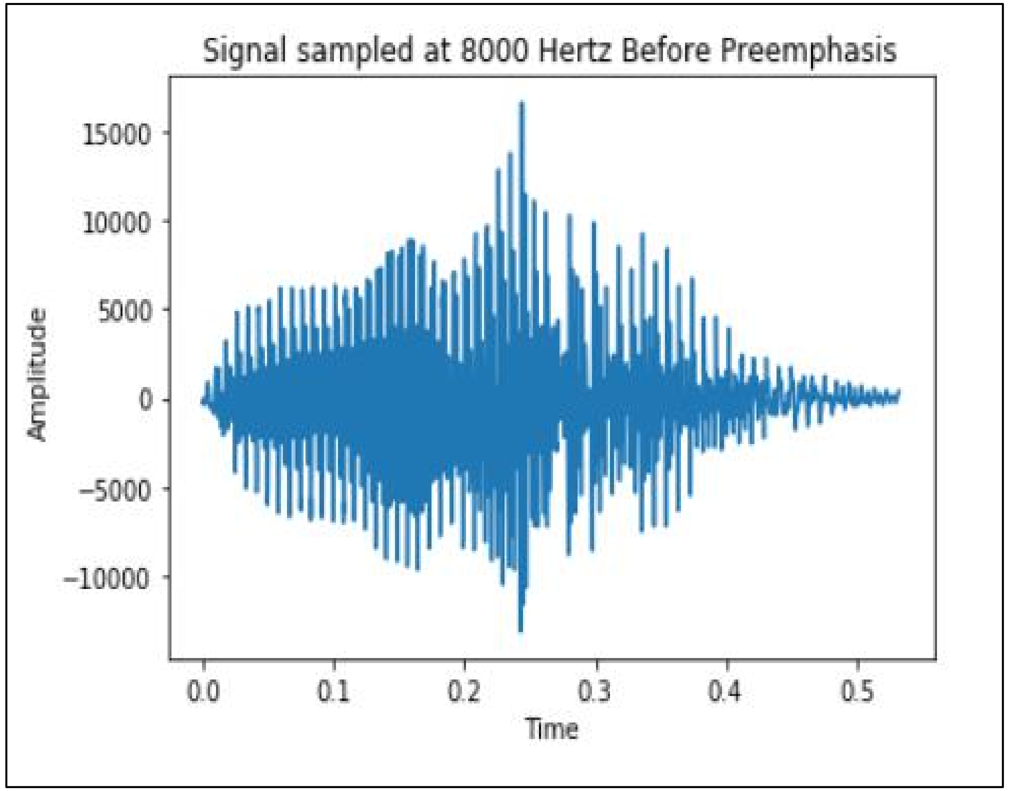
1. A simple audio/speech dataset consisting of recordings of spoken digits in wav files at 8kHz. The recordings are trimmed so that they have near minimal silence at the beginnings and ends.[1]
2. 4 speakers - ['jackson', 'theo', 'nicolas', 'yweweler']
3. 2,000 recordings (50 of each digit per speaker)
4. English pronunciations

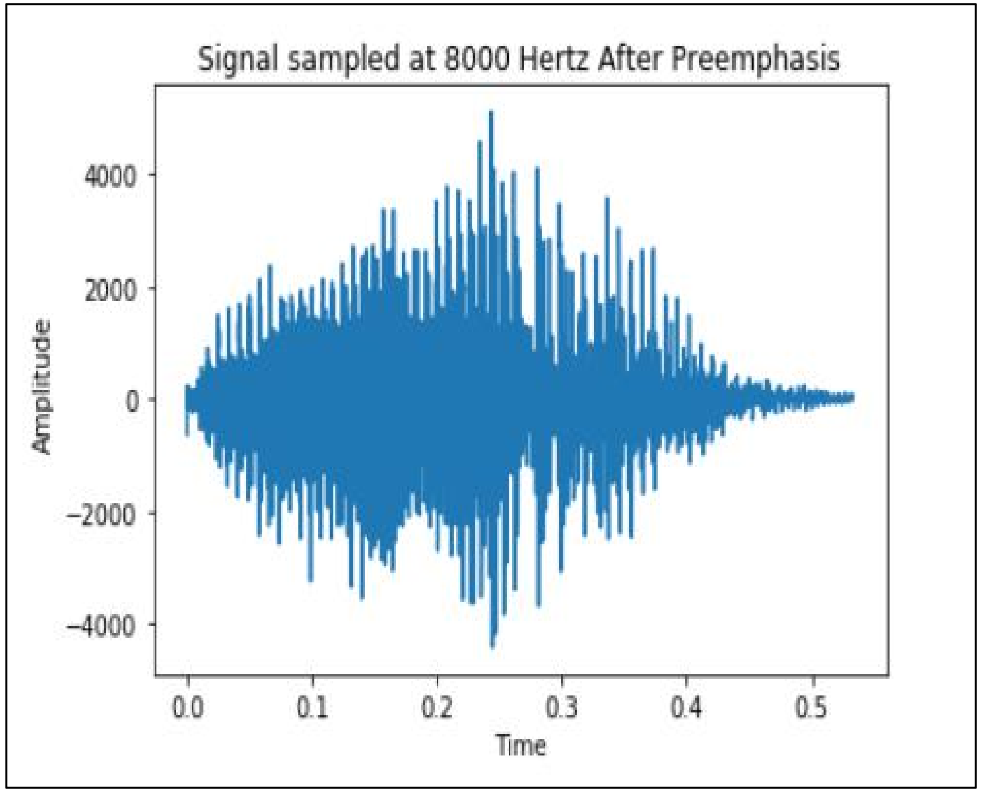
# MFCC Feature Extraction



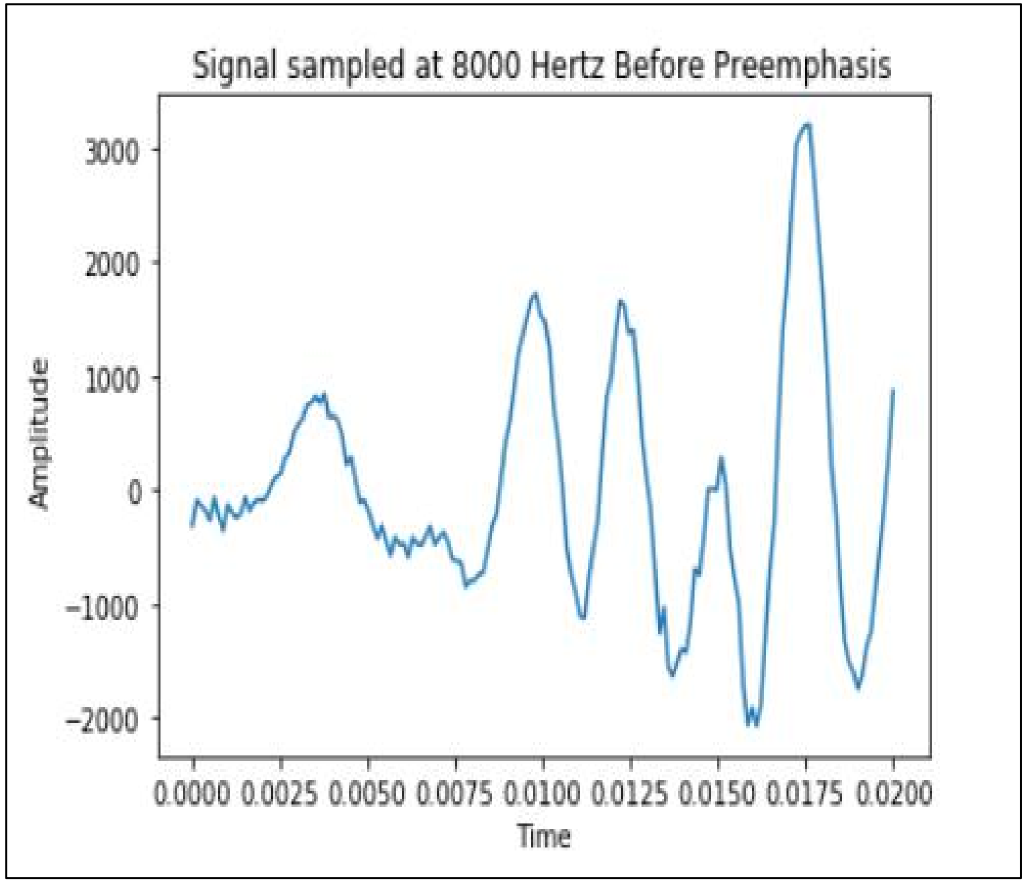

**Block Diagram of MFCC**

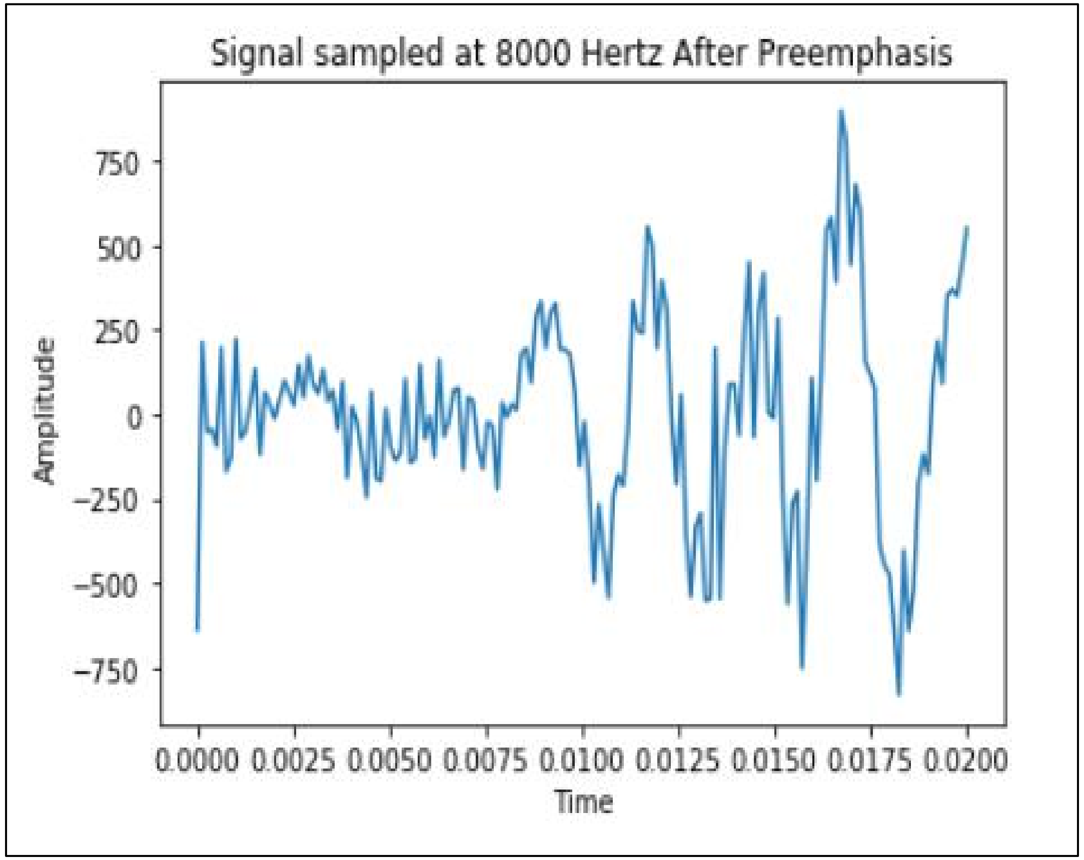
# Before preemphasis vs After (x(t)-0.97*x(t-1))



# Taking a small frame:



# PCA (Principal Component Analysis) [3]

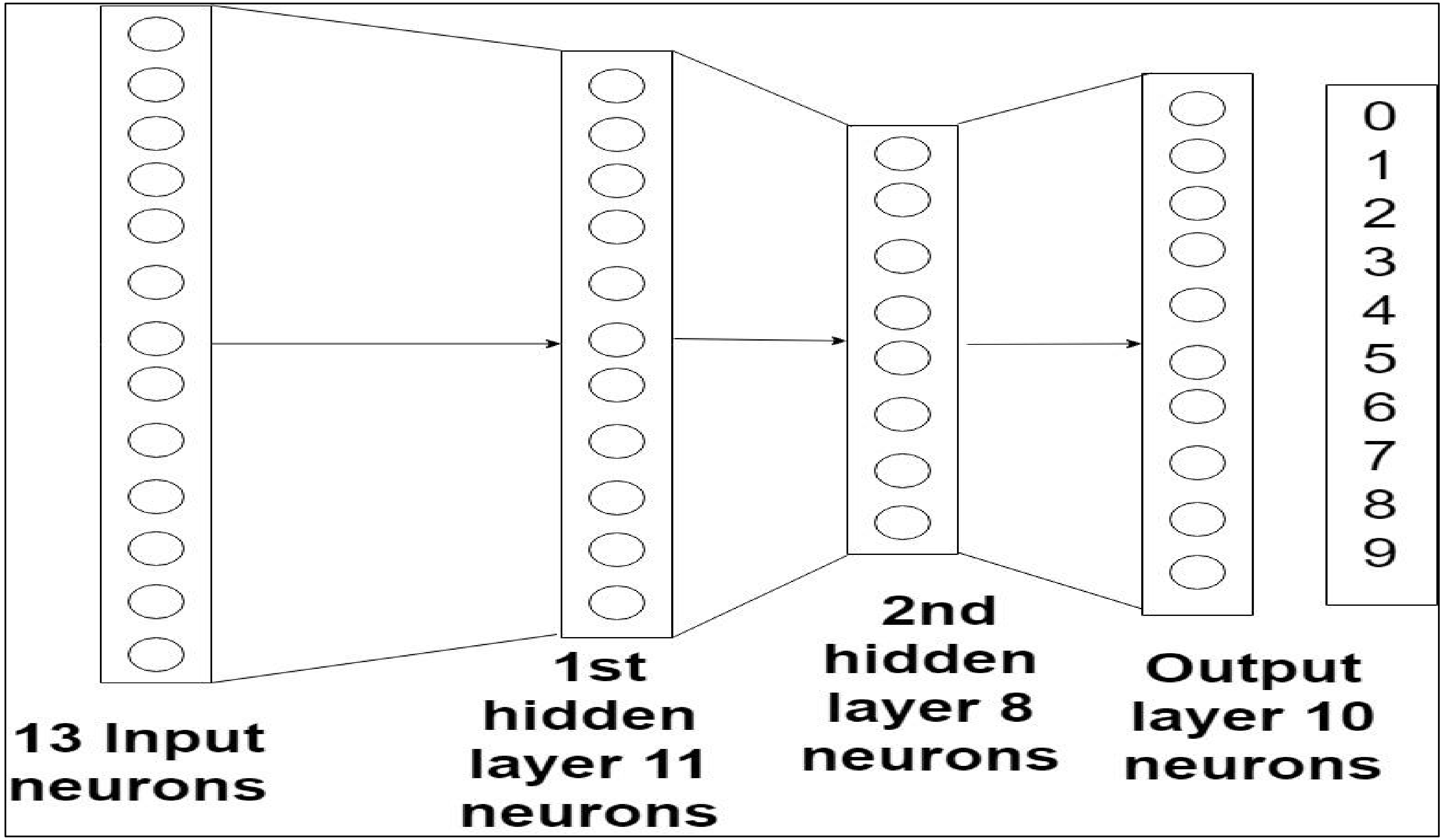
**42X13 Feature Vector to 13X1 Feature Vector**

- A=U.E.V'
- E=Variance Covariance Matrix (Diagonal Matrix)
- U,V= Orthogonal Matrix (U'.U=U.U'=I)
- E is given by (A'.A)/13

**Now Using SVD:**

- A'.A=V.E.'E.V' and A.A'=U.E.E'.U'
- Thus finding out Eigen values and eigen vectors of A'.A, we get U(42X42) matrix
- So final Feature Vector after PCA is : A.U[:,1] -> (13X42)dot(42X1) -> 13X1 vector

# Model Architecture




W1 W2 W3

# Sigmoid VS tanh [2]

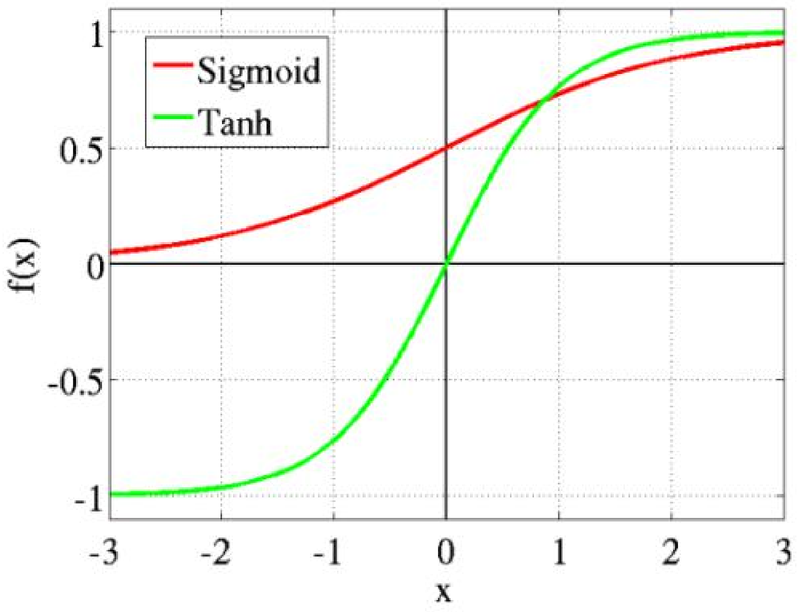



Fig: tanh v/s Logistic Sigmoid

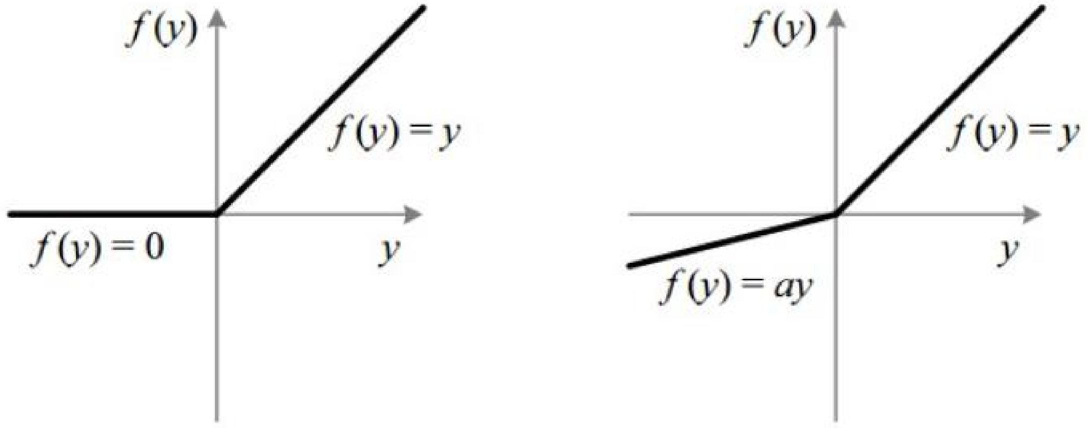



Fig : ReLU v/s Leaky ReLU

$$\cosh(x) = \frac{1}{2}(e^x + e^{-x});\ \sinh(x) = \frac{1}{2}(e^x - e^{-x});\ \tanh(x) = \frac{\sinh(x)}{\cosh(x)}$$

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}.$$

# Forward Propagation

W1,W2,W3 are weight matrices from Input Layer to Output Layer

They are randomly initialized

Let X=Input : 13X1 feature vector

H1= First Hidden Layer

H2= Second Hidden Layer

Out= Result at output Layer

**For ith Example:**

**H1=simgoid(X(i).T * W1+b1)**

**H2=sigmoid(H1.T*W2+b2)**

**Out=sigmoid(H2.T*W3+b3)**

# Backward Propagation

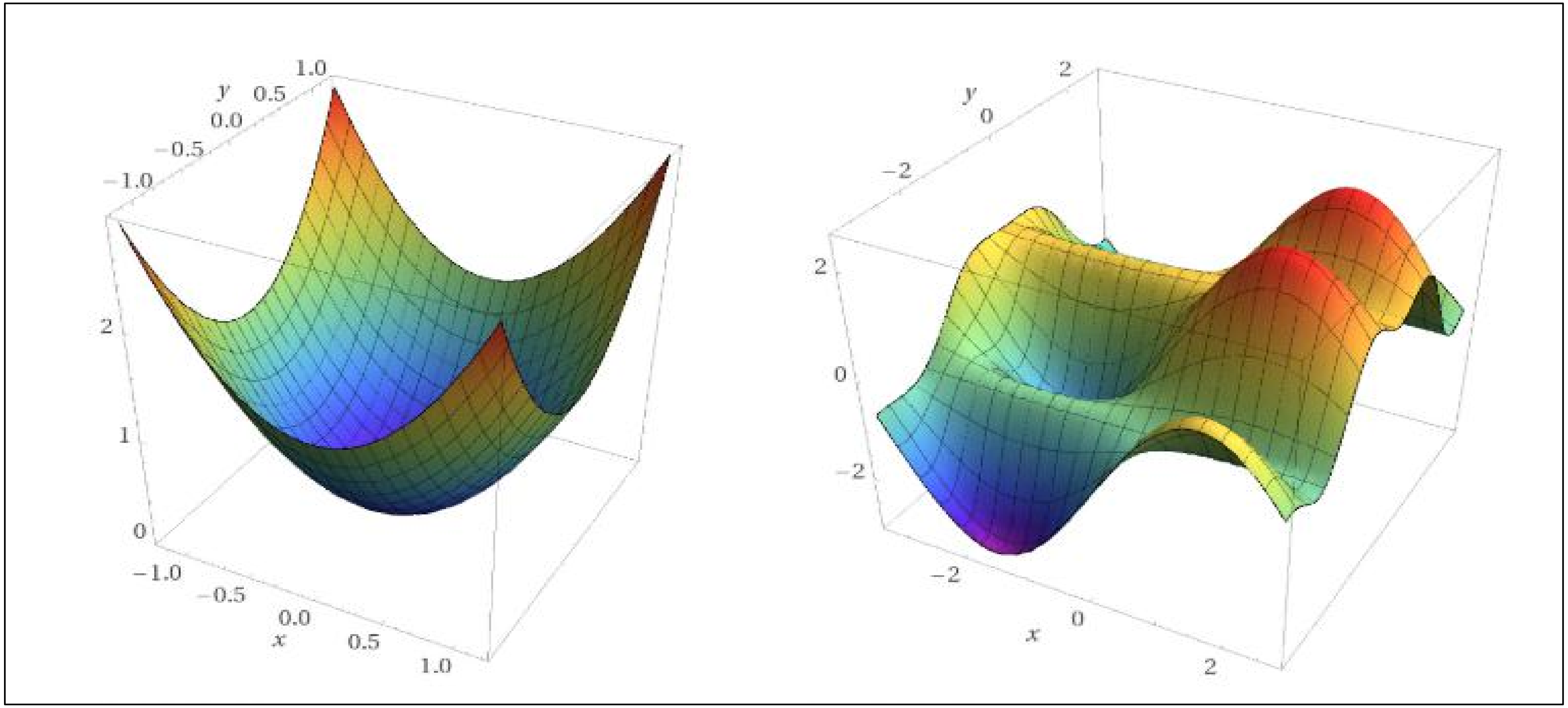
1. Error at Output Layer is Calculated as Mean Squared Error (MSE):

$$J = \text{Cost Function} = (1/2)*(1/10)*(\text{Out-Expected})^2$$

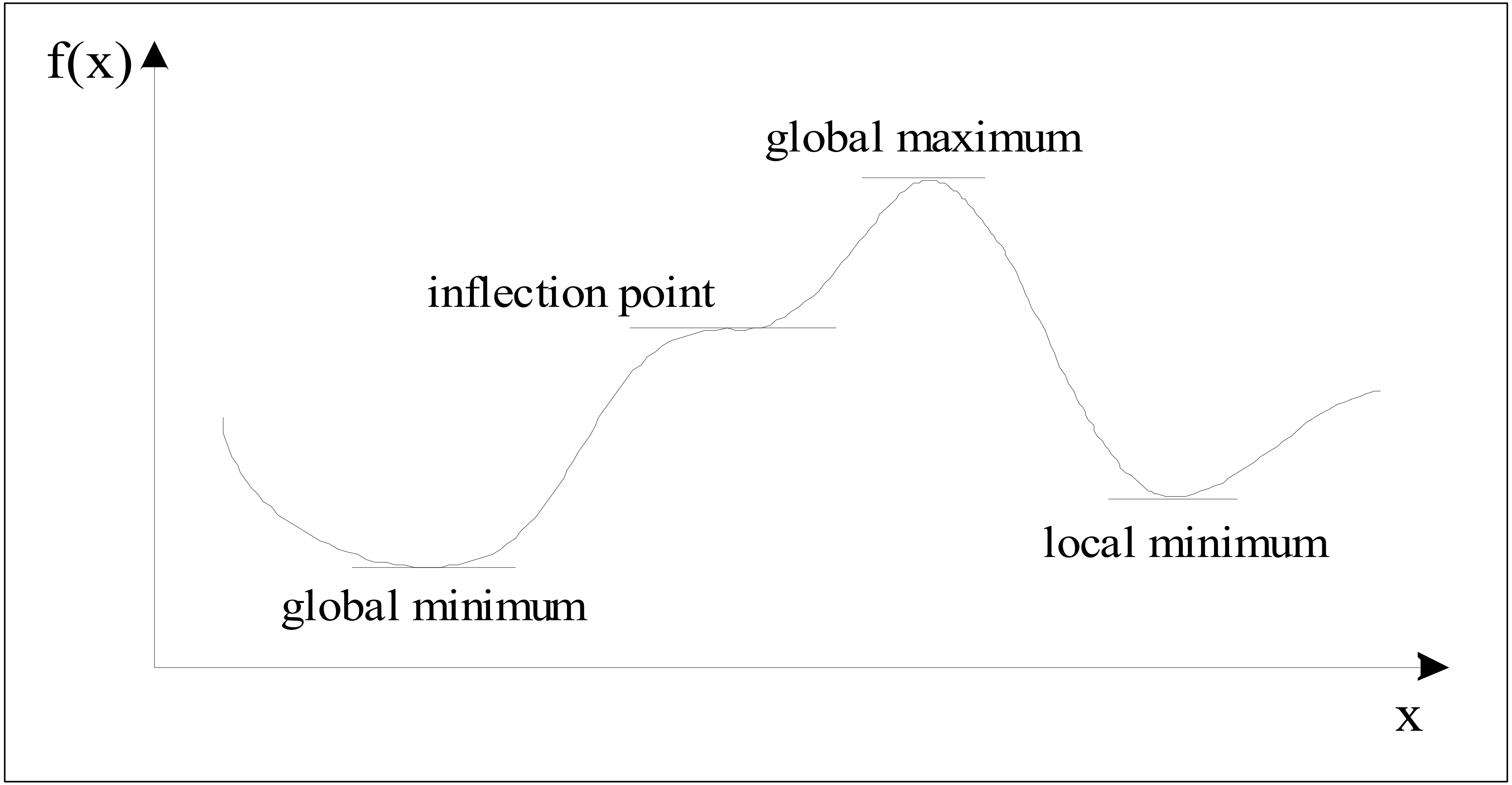
2. Gradient Descent Algorithm (Calculate gradients using Chain Rule)

$$\begin{aligned}&\text{Repeat until convergence } \{\\ &\qquad \theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)\\ &\}\end{aligned}$$

# Convex vs Non- Convex Functions



# Challenges for Gradient Descent



# Parameters Selection:

1. What should be the learning rate ?
2. How many iterations?
3. How to initialize weights? All zeroes or taking from a probability distribution.
4. Using which activation function- Sigmoid, tanh or RelU, leaky RelU etc.
5. Stochastic Gradient Descent or Mini Batch Gradient Descent?

# Results

| Iterations | Learning Rate | Accuracy |
|---|---|---|
| 1000 | 0.1 | 88.83 |
| 1000 | 0.05 | **92.33** |
| 5000 | 0.1 | 92.08 |
| 5000 | 0.05 | 88.75 |
| 1000 | 0.01 | 80.5 |

# Timeline

Implementing LSTM for speech recognition for larger dataset until next presentation

# References

1. Zohar Jackson, César Souza, Jason Flaks, Yuxin Pan, Hereman Nicolas, & Adhish Thite. (2018, August 9). Jakobovski/free-spoken-digit-dataset: v1.0.8 (Version v1.0.8). Zenodo. http://doi.org/10.5281/zenodo.1342401
2. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6
3. https://www.youtube.com/watch?v=mBcLRGuAFUk&t=612s
4. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

# Thanks