CHAPTER VI

RECURRENT NEURAL NETWORK BASED PHONEME RECOGNIZER

6.1 INTRODUCTION

One of the basic assumptions about speech signal is that speech signal composed of many locally stationary but temporally related segments. These quasi-stationary speech segments or a concatenation of them corresponds to the basic phonetic units such as vowels and consonants. Therefore, modelling of a speech signal essentially deals with the concurrent aspects – static features of individual phonetic or sub-phonetic segments and temporal relations of these units. The incapability of artificial neural network to deal with speech dynamic hinders its application in speech recognition. Most of the artificial neural network models are good static classifier, but do not accept sequential or time varying input. An intuitive way to deal with speech dynamic is recurrent neural network.

The recurrent neural network accepts sequentially arranged input vector $U(1), U(2), U(3), \ldots, U(t), \ldots$ one at a time. In speech applications, these vectors typically contain the frame based spectral features. The internal state of RNN at time $t$ is described by the state vector $X(t) = [x_1(t) \ x_2(t) \ \ldots\ x_N(t)]^T$, where $x_n(t)$ is the activation level of the $n^{th}$ neuron and $N$ is the total number of neurons. At each time instant, the present network state $x(t)$ is a non-linear function of current input vector $U(t)$ and previous state $X(t-1)$, i.e.,

$$X(t) = \Gamma(U(t), X(t-1)) \quad \ldots \quad (6.1)$$
where $F(\tau)$ depends on the network architectures of individual neurons. Usually a subset of the $N$ neurons are used as the output neuron whose activation level forms the output vector $Y(t)$. As a result, a sequence of output vectors $\{Y(t)\}$ is produced. Unlike static neural network classifier whose decision is given solely by instantaneous output vector, the RNN utilizes the entire output sequence to discriminate the input sequence from other.

### 6.2 ARCHITECTURE OF THE RECURRENT NEURAL NETWORK FOR SPEECH MODELLING

A fully connected recurrent neural network is used to construct the speech model. This network architecture was described by Williams and Zipser [129] and also known as Williams and Zipser's model. Let the network has $N$ neurons and out of them $k$ are used as output neurons. The output neurons are labelled from $I$ to $k$ and the hidden neurons are labelled from $k+1$ to $N$. Let $P_{mn}$ be the feed-forward connection weight from $m^{th}$ input component to the $n^{th}$ neuron and $w_{ni}$ be the recurrent connection weight from the $i^{th}$ neuron to the $n^{th}$ neuron. At time $t$, when an $M$-dimensional feature vector $U(t)$ is presented to the network, the total input to the $n^{th}$ neuron is given by

$$Z_n(t) = \sum_{i=1}^{N} w_{ni} x_i(t-1) + \sum_{m=1}^{M} P_{mn} U_m(t) \quad \cdots \quad (6.2)$$

where $x_i(t-1)$ is the activation level of the $i^{th}$ neuron at time $t-1$ and $U_m(t)$ is the $m^{th}$ component of $U(t)$. The resultant activation level $X_n(t)$ is calculated as

$$X_n(t) = f_n(Z_n(t)) = \frac{1}{1 + e^{-Z_n(t)}}, 1 \leq n \leq N \quad \cdots \quad (6.3)$$
To describe the entire network response at time \( t \), the output vector \( Y(t) \) is formed by the activation level of all output neuron, i.e.

\[
Y(t) = [x_1(t)x_2(t)\ldots x_k(t)]^T
\]

Following the conventional winner-take-all representation, one and only one neuron is allowed to be activated each time. Thus, \( k \) discrete output states are formed. In
state $k$, the $k^{th}$ output neuron is most activated over the other. Let $s(t)$ denote the output state at time $t$, which can be derived from $Y(t)$ as

$$s(t) = \arg \max_{j=1}^{k} \{ y_j(t) \}$$

... (6.5)

The RNN has been described so far only for a single time-step. When a sequence of input vector \{U(t)\} is presented to the network, the output sequence \{Y(t)\} is generated by eq (6.2) - (6.4). By eq. (6.5), \{Y\{t\}\} can be further converted into an output scalar sequence \{s\{t\}\}, and both of them have the same length as \{U\{t\}\}. \{s\{t\}\} is a scalar sequence with integer value between 1 to n. It can be regarded as a quantized temporal representation of the RNN output.

The fully connected RNN described above performs time aligned mapping from a given input sequence to an output state sequence of the RNN. Each element in the state sequence is determined not only by the current input vector but also by the previous state of the RNN. Such state dependency is very important if the sequential order of input vector is considered as an indispensable feature in the sequence mapping.

In the present study, the recurrent neural network has been used to construct two speech models. The first speech model is named as “All in one speech model” and the second model is named as “Class-based model”. Both the models has been tested and their performance has been evaluated.

6.3 TRAINING THE RECURRENT NEURAL NETWORK

The Real Time Recurrent Learning (RTRL) algorithm was first proposed by Williams and Zipser [129] for temporal supervised training of fully connected recurrent neural networks, The temporal supervised training means that the output vectors $Y(t)$ are
to match specified target vectors over a certain period of time. Let

\[ Y^* (t) = [x_1^* (t), x_2^* (t), x_3^* (t), \ldots, x_k^* (t)]^T \]

be the target vectors over a certain period of time.

The instantaneous output error is given by

\[ e(t) = \frac{1}{k} \sum_{i=1}^{k} [x_i^* (t) - x_i (t)]^2 \]  

... (6.6)

suppose \( Y^* (t) \) is specified for the entire input sequence for \( t=1 \) to \( T \). The overall average output error is given by

\[ E = \frac{1}{T} \sum_{t=1}^{T} e(t) \]  

... (6.7)

\( E \) is minimized using gradient descent technique[40]. For a recurrent connection \( w_{ij} \) \( (1 \leq i, j \leq N) \) the total weight adjustment \( \Delta w_{ij} \) is given by

\[ \Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} \]  

... (6.8)

where \( \eta > 0 \) is the learning rate. Substituting (6.7) into (6.8) we get

\[ \Delta w_{ij} = \sum_{t=1}^{T} \frac{\eta}{T} \frac{\partial e(t)}{\partial w_{ij}} \]

\[ = \sum_{t=1}^{T} \Delta w_{ij} (t) \]  

... (6.9)

where \( \Delta w_{ij} (t) = \frac{\eta}{T} \frac{\partial e(t)}{\partial w_{ij}} \)

\[ = \eta \frac{2}{T} \frac{k}{k} \sum_{i=1}^{k} [x_i^* (t) - x_i (t)] \frac{\partial x_i (t)}{\partial w_{ij}} \]  

... (6.10)

By differentiating the dynamic equation (6.2) and (6.3) we get

\[ \frac{\partial x_i (t)}{\partial w_{ij}} = f'_n (z_i (t)) \cdot \frac{\partial z_i (t)}{\partial w_{ij}} \]
\[ f'_n(z_n(t)) = f_n(z_n(t)) \left[ 1 - f_n(z_n(t)) \right] \] 

... (6.12)

and

\[ \partial_m = \begin{cases} 
1 & \text{if } n = i \\
0 & \text{if } n \neq i 
\end{cases} \] 

... (6.13)

Assuming

\[ \frac{\partial x_n(t)}{\partial w} \bigg|_{t=0} = 0 \] 

... (6.14)

Similarly, for a feed-forward connection \( P_{ih} (1 \leq i \leq N, 1 \leq h \leq M) \), it can be derived that

\[ \Delta P_{ih} = \sum_{t=1}^{T} \Delta P_{ih}(t) \] 

... (6.15)

and

\[ \Delta P_{ih}(t) = \eta \frac{2}{Tk} \sum_{j=1}^{k} \left[ x_j(t) - x_j(t) \right] \frac{\partial x_j(t)}{\partial p_{ih}} \] 

... (6.16)

where

\[ \frac{\partial x_n(t)}{\partial p_{ih}} = f'_n(z_n(t)) \left[ \sum_{l=1}^{N} w_{ml} \frac{\partial x_l(t-1)}{\partial p_{ih}} + \partial_m U_n(t) \right] \] 

... (6.17)

In the practical implementation of RTRL algorithm, the connection weights are continuously increased by an amount \( \Delta w_{ij}(t) \) or \( \Delta p_{ih}(t) \) at each step. Theoretically, such a real-time system has a potential disadvantage that it may not necessarily follow the precise negative gradient of temporal error function \( E \). However, it has been found in the present study that if the learning rate is small enough, only a small discrepancy would be caused and the RTRL operate satisfactorily to minimize \( E \).
6.3 THE ALL IN ONE SPEECH MODEL

In the present approach, a single RNN has been used to model all the 53 phonemes of Assamese and Bodo languages. The network consists of 18 input units, variable number of hidden units and 53 output units. The sequentially arranged input vector \{\{U(1), U(2), U(3), \ldots\ldots\ldots, U(T)\}\}, extracted from an utterance of a phoneme has been given as input to the speech recognizer. The 53 output units correspond to 53 phonemes of Assamese and Bodo languages. The decision on the number of hidden units is a crucial one since the number of hidden units has a direct effect on recognition accuracy and convergence time. With the increasing number of hidden nodes the recognition accuracy as well as convergence time increases. The system developed for the present study used 40 hidden units, which is found to be optimal in the model used for the recognition of 53 phonemes of Assamese and Bodo languages.

The feature vector extracted from each frame has been taken as input to the speech recognizer. To make the comparison between the speech recognizer discussed here and discussed in the chapter-V fair, same input vector is used for the recognizer. The normalized feature vector having 18 elements for each frame has been taken as input. The database has been divided into two parts – 10 samples of each phoneme have been taken for training and the remaining 20 samples are taken for evaluating the performance of the system. Initially, the number of hidden nodes is kept at 20 and gradually it is increased to 60 in the steps of 10. The performance of the recognizer is evaluated for each phoneme and result is summarized in the Table – (6.1).
Table (6.1): Recognition accuracy of the recognizer

<table>
<thead>
<tr>
<th>Number of Hidden Nodes -&gt;</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vowel</td>
<td>57.53</td>
<td>68.34</td>
<td>79.51</td>
<td>91.72</td>
<td>92.23</td>
<td>92.16</td>
</tr>
<tr>
<td>Consonant</td>
<td>53.59</td>
<td>64.33</td>
<td>74.75</td>
<td>86.22</td>
<td>86.50</td>
<td>86.14</td>
</tr>
<tr>
<td>Average</td>
<td>55.56</td>
<td>66.34</td>
<td>77.13</td>
<td>88.97</td>
<td>88.87</td>
<td>89.15</td>
</tr>
</tbody>
</table>

6.4 THE CLASS BASED SPEECH MODEL

When only a single recurrent neural network is used for the recognition of all phonemes of Assamese and Bodo languages, it has been observed that the computational time increases and recognition accuracy decreases. One way to cope with this problem is to divide the entire phoneme recognition task into two subtasks – (a) recognition of the phoneme class and (b) recognition of the phoneme within that class. All the phonemes of Assamese and Bodo languages are divided into eight categories – vowels, nasals, voiced fricatives, unvoiced fricatives, voiced stop, unvoiced stop, rolled and glides. The list of phoneme under each category is given in the Table-(6.2).
Table (6.2): Classification of Assamese and Bodo phonemes

<table>
<thead>
<tr>
<th>Class</th>
<th>Assamese</th>
<th>Bodo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vowel</td>
<td>[ə], [a], [ɛ], [i], [ɛ], [o], [ɔ], [u]</td>
<td>[ɪ], [ɛ], [a], [u], [o], [w]</td>
</tr>
<tr>
<td>Nasal</td>
<td>[m], [n], [ŋ]</td>
<td>[m], [n], [ŋ]</td>
</tr>
<tr>
<td>Voiced Fricative</td>
<td>[z],[h],[l]</td>
<td>[z],[h],[l]</td>
</tr>
<tr>
<td>Unvoiced Fricative</td>
<td>[s],[x]</td>
<td>[s]</td>
</tr>
<tr>
<td>Voiced Stop</td>
<td>[b], [bʰ],[d], [dʰ],[g],[gʰ]</td>
<td>[b],[d],[g]</td>
</tr>
<tr>
<td>Unvoiced Stop</td>
<td>[p], [pʰ],[t],[tʰ],[k],[kʰ]</td>
<td>[pʰ],[tʰ],[kʰ]</td>
</tr>
<tr>
<td>Roll</td>
<td>[r]</td>
<td>[r]</td>
</tr>
<tr>
<td>Glide</td>
<td>[w],[j]</td>
<td>[w],[j]</td>
</tr>
</tbody>
</table>

The structure of the phoneme recognizer is shown in the Fig-(6.2). The first unit, i.e. phoneme classifier consist of 18 input units, 8 output units and 10 hidden unit, which is found to be optimal. The second part, that is the phoneme identifier, consist of 18 input units, 1~8 output units and 10 hidden units. The system is trained and tested with database used in the previous experiment. The recognizer has been trained using RTRL algorithm with 10 samples of each phoneme and 20 samples of each phoneme is used to test the performance of the recognizer. The results of the experiments are listed in the Table – (6.3).
Fig (6.2): Two-layered RNN based speech recognizer

Table (6.3): Recognition accuracy of the individual classes and their average

<table>
<thead>
<tr>
<th>Category</th>
<th>Sub Category</th>
<th>Recognition Accuracy (in %)</th>
<th>Average Recognition Accuracy (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vowel</td>
<td>Vowel</td>
<td>96.52</td>
<td>96.52</td>
</tr>
<tr>
<td>Consonant</td>
<td>Nasal</td>
<td>82.17</td>
<td>86.10</td>
</tr>
<tr>
<td></td>
<td>Voiced Fricative</td>
<td>91.45</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unvoiced Fricative</td>
<td>78.53</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voiced Stop</td>
<td>92.67</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unvoiced Stop</td>
<td>83.44</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Roll</td>
<td>81.21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Glide</td>
<td>93.23</td>
<td></td>
</tr>
</tbody>
</table>

Overall Recognition Accuracy 91.31