Pattern Recognition - Report for Assignment 4

Navneet Agrawal, navneet@kth.se
Lars Kuger, kuger@kth.se

I. Introduction

In this assignment, we will rectify the errors in code *forward_fixme.m* given by the course assistant and describe the speech samples recorded for the training of our speech recognizer. In following sections we will discuss the method used to find the errors, which errors are found and how they were resolved, and finally the verification method. We will also discuss the consequences of each error if its not been fixed. After that, a short description of the recorded vocabulary and a simple application for it are proposed.

II. List of corrections

Line	Error / Description	Error Type & Rectification
41	`T = size(pX,1);` `pX` matrix dimensions are correctly represented by [number of states (N), Sequence length (T)]	**Logical flaw** `[numberOfStates, T] = size(pX);`

51	`if(rows  = columns)`
	`q = [q;0];`
	`tempz = log(tempx); end`
	No need to have initial probability for exit state.
		Unnecessary step
		Removed

58	`initAlfaTemp(j) = q(j)*B(i,1);` Incorrect variable `i`	**Syntax error** `initAlfaTemp(j) = q(j)*B(j,1);`
65	`for t=2:2` The forward algorithm should iterate from sequence 2 to T	**Syntax error** `for t=2:T`
68	`alfaTemp(j) = B(j,t)*(sum(alfaHat(:,t-1)'*A(:,j)));` `sum` is not required. Output of (Row * Column) matrix multiplication will give sum of product of corresponding terms.	**Unnecessary step** `alfaTemp(j) = B(j,t)*(alfaHat(:,t-1)'*A(:,j));`
69	`if(tempx>tempx-1) break; end;` Always break the loop at first value	**Runtime error** *Removed*
74	`for j=1:numberOfStates/2` `alfatemp` should be updated for all states.	**Syntax error** `for j=1:numberOfStates`
82	`c = [c 0.0581];` scaling factor for final state cannot be a constant	**Logical flaw** `c = [c sum(alfaHat(:,end)'*A(:,end))];`

TABLE I: List of errors found in the code *forward_fixme.m*

III. Identification and Correction of errors

Errors in *forward_fixme.m* are identified by following the code systematically and comparing it with forward algorithm. Forward algorithm has three parts - *Initialization*, *Iteration* and *Termination*. We analyzed the code to identify these parts, and then checked these parts for valid implementation of the algorithm. We found some critical errors (error at line number 41, 58, 65, 69, 74 & 82 in table I) and some unnecessary steps. Critical errors must be rectified in order to get the code working correctly. Other errors such as unnecessary steps or memory allocation issues makes the code slower. Rectification of these errors will make the code run faster.

IV. Verification

The corrected code *forward_fixme.m* is verified by generating *alfahat* and *scaling factor (c_t)* values using the function *forward*. The script used for verification *verify.m* can be found in the zip archive submitted with the assignment. The code *verify.m* uses various classes (*@GaussD*, *@DiscreteD*, *@Markov* and *@HMM*) to create objects required to model the random parameters. These classes are provided as part of the project.

A. Finite HMM

For verifying *finite HMM*, we used model $\lambda = \{q, A, B\}$ given by equation 1.

$$q = [1; 0],\ A = \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0 & 0.9 & 0.1 \end{bmatrix},\ B = \begin{bmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(3,1) \end{bmatrix} \tag{1}$$

The values of *alfahat* and *scaling factor (c_t)* generated by the corrected *forward_fixme.m* (equation 4) are correct and can be confirmed from [1] page 238.

$$\hat{\alpha} = \begin{bmatrix} 1 & 0.3847 & 0.4189 \\ 0 & 0.6153 & 0.5811 \end{bmatrix},\ \mathbf{c} = [1, 0.1625, 0.8266, 0.0581] \tag{2}$$

B. Infinite HMM

Infinite HMM can be verified by using the model $\lambda = \{q, A, B\}$ given by equation 3. The values are taken from Execise 5.1(b) of [1].

$$q = [1; 0; 0],\ A = \begin{bmatrix} 0.3 & 0.7 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{bmatrix},\ B = \begin{bmatrix} DiscreteD(\,[1,0,0,0]\,) \\ DiscreteD(\,[0,0.5,0.4,0.1]\,) \\ DiscreteD(\,[0.1,0.1,0.2,0.6]\,) \end{bmatrix} \tag{3}$$

The values of *alfahat* and *scaling factor (c_t)* generated by the corrected *forward_fixme.m* (equation 4) are correct and can be confirmed from [2] page 26.

$$\hat{\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0.1429 & 0.0127 & 0 \\ 0 & 0 & 0.8571 & 0.9873 & 1 \end{bmatrix},\ \mathbf{c} = [1, 0.3500, 0.3500, 0.5643, 0.0994] \tag{4}$$

V. Recorded Speech Samples

A. Recording Procedure

The recorded speech samples can be found in the folder *audiofiles* and are grouped in folders in a hierarchical order depending on the word and the speaker. The recording process was done using the MATLAB scripts *recordWord.m* and *recordWord_navneet.m* which allow the user to enter the word to be recorded, her name, and the number of repititions. The scripts differ only slightly in terms of how they access the audio recording device on different computers.
In our particular case, we recorded 11 words with 15 repititions by each speaker as suggested in [1]. The words were recorded by two male and two female speakers so that we will be able to train the speech recognizer independently of the pitch of the voice. Note that the accents of the speakers differ noticeably. A good example for this is the word *Wednesday* which is pronounced differently by the speaker Lars as compared to the speaker Navneet. Also, it is nocticeable that one speaker can pronounce the same word in various manners, i.e. the word Saturday by the speaker Navneet.
The technical specifications of the audiofiles follow the proposed ones in [1], e.g. the recordings were done at a sampling frequency $f_s = 22050$ Hz with a resolution of $N = 16$ bits/sample and with one channel only.

B. Application

One application that could employ this speech recognition is a calendar app. A simple speech user interface could be as follows.

Speaker: Hello.
App: Welcome to the speech recognition of this calendar app. Do you want me to view one day of the upcoming week?
Speaker: Yes.
App: Which day would you like to view?
Speaker: Wednesday.
App shows appointments for Wednesday.
Speaker: Bye.
App is closed.

In order to implement such an application we recorded the following 11 words.

- Hello
- Bye
- Yes

- No
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday
- Sunday

These words were recorded as described in the previous section. The complete zip archive can be found in https://drive.google.com/file/d/0B80K4ePe7CHAY0VhdHd4allaOEU/view.

REFERENCES

[1] Arne Leijon and Gustav Eje Henter, *Pattern Recognition - Fundamental Theory and Exercise Problems*, KTH School of Electrical Engineering, 2015

[2] Arne Leijon, Et Al., *Solutions Manual : Pattern Recognition*, KTH School of Electrical Engineering, 2015, *https://www.kth.se/social/files/5612757af2765428622c5b4a/Ch.5.pdf*