Human Daily Activity Recognition By Guassian Mixture Based HMM

Yongwei Wang, He Zhang, He Huang
Department of Electrical and Computer Engineering
Project report of EECE 562

Abstract

Human activity recognition plays a significant role in artificial intelligence. However, automatic detection of human activities could be challenging since difference may exists among people even when they are doing the same activity, which is regarded as the individual nature of activities. In order to tackle this problem, in this project, we study a human activity detection model that users 3-D skeleton features generated from an RGB-D sensor. In order to capture the multimodel nature of 3-D positions of each skeleton joint, a Guassian Mixture Model (GMM) based Hidden Markov Model is implemented. Further, we apply the dynamic Bayessian network (DBN) to characterize the relationship between the activities and multiple joint positions, so that a more accurate human detection can be achieved. By exploiting the K-mean algorithm, and EM algorithm, we train the GMM based HMM for each activity by using the Cornell Activity Dataset. Once a sequence of skeleton features have been captured , these previously trained models produce likeligood estimation, from which the maximum is selected. By this scheme, a human daily activity recognition is achieved.

Most of the work in this project is based on the paper: L. Piyathilaka and S. Kodagoda, ”Gaussian mixture based HMM for human daily activity recognition using 3D skeleton features,” *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, Melbourne, VIC, 2013, pp. 567-572. ***Teamwork distribution***: The model study and design is discussed by Yongwei Wang, He Zhang, He Huang; The algorithm is designed by Yongwei Wang, He Zhang, He Huang; The matlab code is written by Yongwei Wang; The matlab code debugging is discussed by Yongwei Wang and He Zhang; The experiment analysis is written by Yongwei Wang; The project report is written by He Zhang.

I. Introduction

Automatic detection of human activities is now playing a significant role in Human Robot Interaction (HRI), which is a sub field in robotics. In HRI, the interaction among human and robots is studied. It combines knowledge of many interdisciplinary fields such as robotics, Artificial Intelligence, natural language processing and social sciences. The ability of robots to recognize the human activities and response correspondingly is the key for better interaction between human and robots. For example, by detecting the activities of an elderly person, a health care robot can remind them of their medications in appropriate times, or the robot can detect abnormal conditions of patients and notify the appropriate personnel.

The invention of low cost RGB-D cameras like Microsoft Kinect can provide wealth of information like depth. In addition, RGBD data from the Kinect sensor can be used to generate a Skeleton model of humans with semantic matching of 15 body parts. The information of how different body parts move across each time period can well be used to characterize human activities. However, one critical challenge of automatic detection of human activities is the individual nature of the activities. That is, often two individuals might perform the same activity in two slightly different ways. These variations make it difficult to generalize a machine learning technique that can train on one person and test on another.

In order to tackle the challenge of individual nature and well utilize the 3-D skeleton position information provided by a RGB-D camera, we utilize Gaussian Mixture Model (GMM) based HMM for human activity detection. Guassian mixtures are capable of clustering data into different groups as a collection of multinomial Guassian distributions. By applying GMM based HMM for each skeleton joint, the impact of individual nature can be eliminated and more accurate detection can be achieved. Further, human actions are a collection of how different human body poses sequentially transfer at different time. So different skeleton joints should be jointly estimated to detect the human action. In this sense, we introduce the dynamic bayesian network (DBN) so that multiple observations of skeleton joints can be characterized to infer the hidden states of individual pose. Therefore, each body pose can be devised as a collection of multinomial distribution and HMM can model the intra slice dependencies between each time period. We implement the proposed GMM based HMM using the Byasian Network TooBox in Matlab and use Cornel activity Detection Dataset to train and test the accuracy of our model.

II. Review of DBN and GMM based HMM

We first make a review of DBN and GMM based HMM, so that we can have a better understanding of the human daily activity recognition problem.

A. Dynamic Bayesian Networks

In an HMM as we learnt in the course, the hidden state is represented in terms of a single discrete random variable, which can take on M possible values, $Q_t \in \{1, \dots, M\}$. However, in a dynamic bayesian network, the hidden state is represented in terms of a set of N_h random variables, $Q_t^{(i)}$, $i \in \{1, \dots, N_h\}$, each of which can be discrete or continuous. Similarly, the observation can be represented in terms of N_o random variables, $Y_t^{(i)}$, each of which can be discrete or continuous.

In an HMM, we have to define the transition model, $P(Q_t|Q_{t-1})$, the observation model, $P(Y_t|Q_t)$, and the initial state distribution, $P(Q_1)$. In a DBN, Q_t, Y_t represent sets of variables, so we define the corresponding conditional distributions using a two-slice temporal Bayes net (2TBN), which we shall denote by $B_\rightarrow$. Therefore, the transition and observation models are then defined as a product of the CPDs in the 2TBN:

$$P(Z_t|Z_{t-1}) = \prod_{i=1}^{N} P(Z_t^{(i)}|\text{PA}(Z_t^{(i)}))$$

where $Z_t^{(i)}$ is the i'th node in slice t (which may be hidden or observed, then we have $N = N_h + N_o$), and $\text{Pa}(Z_t^{(i)})$ are the parents of $Z_t^{(i)}$, which may be in the same or previous time-slice. We then represent the unconditional initial state distribution , $P(Z_1^{(1:N)})$, using a standard (one-slice) Bayes net, which we shall denote by B_1. Together, B_1 and $B_\rightarrow$ define the DBN. The joint distribution for a sequence of length T can be can then be obtained as

$$P(Z_{1:T}^{1:N}) = \prod_{i=1}^{N} P_{B_1}(Z_1^{(i)}|\text{Pa}(Z_t^{(i)})) \times \prod_{t=2}^{T}\prod_{i=1}^{N} P_{B_\rightarrow}(Z_t^{(i)}|\text{Pa}(Z_t^{(i)}))$$

For a special case of HMM that we have learned in the course, it can then be denoted as

$$P(Q_{1:T}, Y_{1:T}) = P(Q_1)P(Y_1|Q_1) \times \prod_{t=2}^{T} P(Q_t|Q_{t-1})P(Y_t|Q_t)$$

B. HMMs with mixture-of-Gaussians output

In many applications, it is common to represent $P(Y_t|Q_t = i)$ using a mixture of Guassian for each state i, we can explicitly model the mixture variable as shown in Figure 1. (M_t and Y_t are

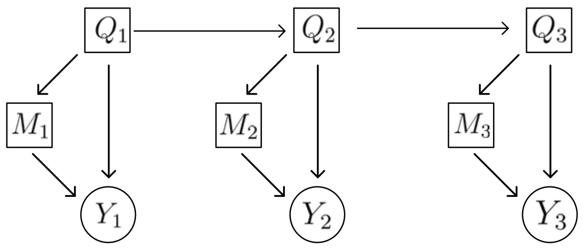

Figure 1: An HMM with mixture of Gaussian output

examples of transient nodes, since they do not have any children in the next time slice; by contrast, Q_t is a persistent node.) The CPDs for the Y_t and M_t nodes are as follows:

$$\begin{aligned} P(Y_t = y_t|Q_t = i, M_t = m) &= \mathcal{N}(y_t; \mu_{i,m}, \Sigma_{i,m}) \\ P(M_t = m|Q_t = i) &= C(i, m) \end{aligned}$$

The i'th row of C encodes the mixture weights for state i.

Assuming there is a single global pool of Gaussians, and each state corresponds to a different mixture over this pool. This is called a semi-continuous or tied-mixture HMM. In this case, there is no arc from Q_t to Y_t, so the CPD of Y_t becomes

$$P(Y_t = y_t|M_t = m) = \mathcal{N}(y_t; \mu_m, \Sigma_m)$$

Then the effective observation model becomes

$$\begin{aligned} P(Y_t|Q_t = i) &= \frac{\sum_m P(Y_t, M_t = m, Q_t = i)}{P(Q_t = i)} \\ &= \frac{\sum_m P(Q_t = i)P(M_t = m|Q_t = i)P(Y_t|M_t = m)}{P(Q_t = i)} \\ &= \sum_m P(M_t = m|Q_t = i)\mathcal{N}(y_t; \mu_m, \Sigma_m) \end{aligned}$$

III. Human Activity Recognition Model

A. Overall Process

The overall process of the human activity recognition can be explained as follows: By applying DBN and GMM based HMM, we train separate HMM for each activity in the data-set. Once a sequence of skeleton features have been captured, these previously trained models produce likelihood estimation, from which the maximum is selected. For the HMM training, we apply the data set from

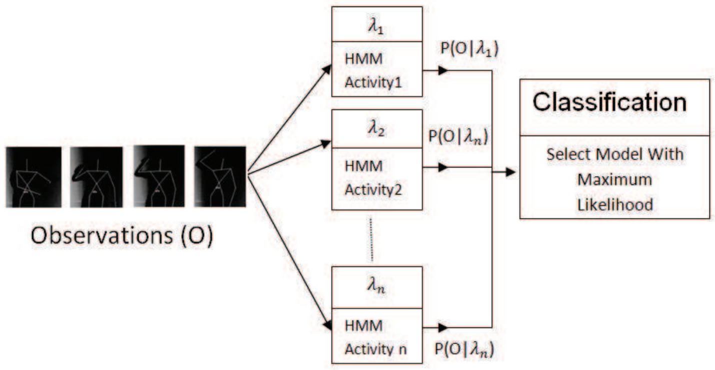

Figure 2: Human activity recognition process

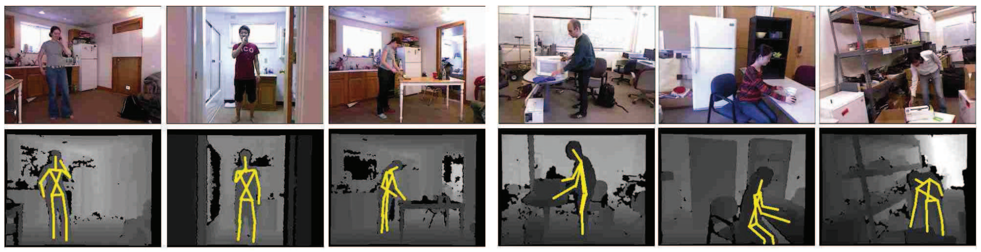

Figure 3: Skeleton visualization

Microsoft Kinect RGB-D camera, which is capable of simultaneously track 15 body positions and their rotational matrices. These positions of body parts can then be used to construct the skeleton structure of a human body, which is shown in Figure 3. We assume that each human activity is a collection of different poses that evolves over time. So the human activities are a mixture of how different body parts are sequentially interacted, position information of body parts from the kinect's human skeleton can be used to construct the HMM for human activity detection.

B. HMM With Guassian Mixture Output

We illustrate HMM with Guassian Mixture output as a Dynamic Baysian network, shown in Figure 4. In this dynamic baysian network, S_t represents the individual pose states at time t, which is the discrete hidden nodes. Since a certain human action is a collection of different poses that evolves over time, the discrete hidden nodes of individual pose form a first-order markov process. That is, the pose state at time t will only be affected by the pose state at time $t-1$. Further, the nodes J_i^t $(1 \leq i \leq 14, 1 \leq t \leq T)$ represents the i'th skeleton joint position at time t. We also note that the joint positions at time t are only directly determined by the pose state at time t, but not the the joint

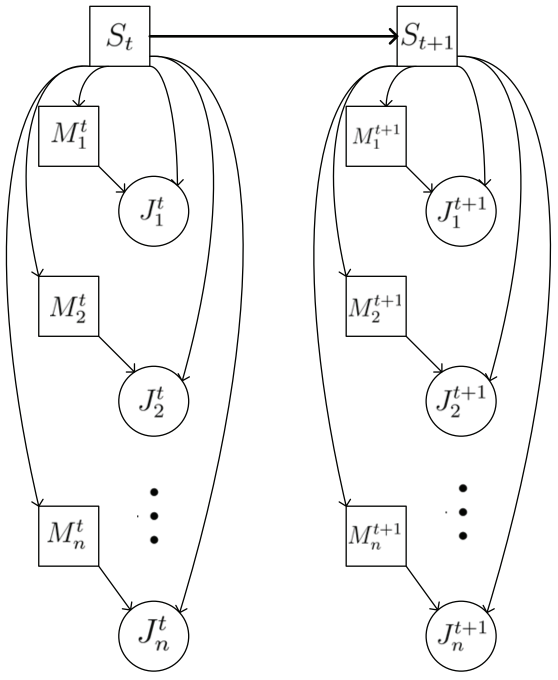

Figure 4: 2-TBN representation of the GMM based HMM

positions at time $t-1$. For each joint J_i^t, we introduce M_i^t to represent mixture weight components for the mixture gaussian output of each joint.

Specifically, in Figure 5 which shows the position information of the right hand when drinking water activity is performed by three people, few distinguishable clusters can be observed. However, despite of these clusters, few sub clusters can also be observed, which is due to the subject related variations in performing even the same activity. Therefore, this individual nature problem poses critical challenge for modeling such activities. In order to tackle this challenge, instead of applying unimodel Gaussians based HMM, we implement HMM based on Gaussian mixture model. A Gaussian mixture model is a weighted sum of M component Gaussian densities as given by the equation:

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i)$$

where x is a D-dimensional continuous-valued data vector. In our problem, since we use the 3D skeleton joint position data set, x is a 3-dimensional continuous-valued data vector. $w_i, i = 1, \ldots, M$, are the mixture weights, and we also have

$$p(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}\Sigma_i^{1/2}} \exp\{-\frac{1}{2}(x-\mu_i)'\Sigma_i^{-1}(x-\mu_i)\}$$

For now, in our approach for human activity detection, we modeled a GMM based HMM as a

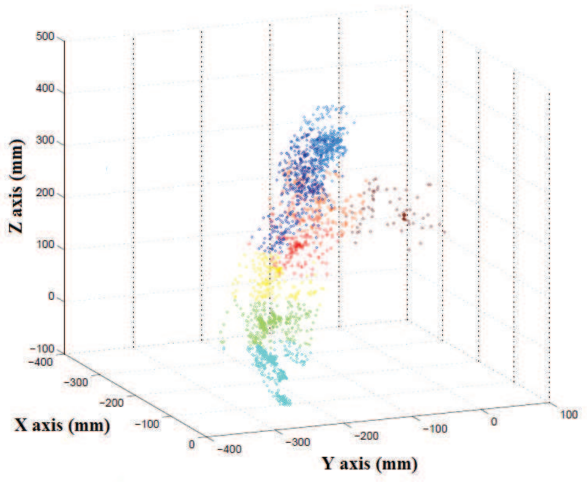

Figure 5: (x, y, z) positions of the right hand when the action "drinking water" is performed

Dynamic Baysian Network. In this model, the joint positions $\{J_i^t | t = 1, \ldots, T; i = 1, \ldots, 14\}$ are the observations. The pose class $\{S_i | i = 1, \ldots, N\}$ and the mixture weight components are hidden states.

For an HMM, there are generally three parameters that define the model, A, B and π. A represent the transition matrix, B represents the observation matrix, and π represent the initial probability distribution of the pose state. We further define s_t as the individual pose state at time t, which is drawn from the pose state class $\{S_1, S_2, \ldots, S_N\}$, in which N is the number of states. We define $a_{i,j}$ as the pose state transition probability from state i to state j, and $b_t(i)$ as the probability of the observation O_t given the i_{th} state of the pose nodes. The initial state distribution $\pi = \{\pi_i\}$ can be defined as

$$\pi_i = P(s_1 = S_i), \quad i = 1, 2, \ldots, N$$

Then, the observation probability distribution is

$$b_t(i) = P(O_t | s_t = S_i), \quad i = 1, \ldots, N, \ \ t = 1, \ldots$$

O_t is the joint observation at time t.

Further, for the mixture of Guassian approach the observation probability can be modeled as

$$b_t(i) = \prod_{n=1}^{J} \left[\sum_{m=1}^{M_i^n} w_{i,m}^n N(O_t^n, \mu_{i,m}^n, \Sigma_{i,m}^n) \right]$$

where J represents the total number of joints, O_t^n the observation vector of the n^{th} node at time t, M_i^n is the number of mixture components in the joint n and state i, and $\mu_{i,m}^n, \Sigma_{i,m}^n, w_{i,m}^n$ are the mean, covariance matrix, and mixture weight for the n^{th} joint, i^{th} state, and m^{th} Gaussian mixture component, respectively.

The state transition probability distribution can be denoted as

$$a_{i,j} = P(s_{t+1} = S_j | s_t = S_i) \quad i, j = 1, \ldots, N$$

Specifically, in this project we consider three individual pose states. Further, we note that human activities are done in specific order, so the additional constrained are placed on the state transition coefficient to make sure that large changes in the state indices do not occur. Due to the repetitive nature of the human activities we only allow swquential transition from state $1 \to 2 \to 3 \to 1$. Then the form of the state transition matrix is given as

$$\begin{bmatrix} a_{11} & a_{12} & 0 \\ 0 & a_{22} & a_{23} \\ a_{31} & 0 & a_{33} \end{bmatrix}$$

IV. Implementation

A. Toolbox and Dataset

We use the Bayes Net Toolbox for Matlab to implement the GMM based HMM. Bayes Net Toolbox can be used to create and manipulate Bayesian networks, both static and dynamic Bayesian networks. In our implementation, we focus on the 2-TBN represented in Figure 2, so that the number of parameters that have to be defined can be reduced. For our model, we define three hidden states for pose node and three components for mixture node.

Further, we employ the Cornell Activity Dataset 60 (CAD 60) to test our model. This dataset use the Microsoft Kinect RGBD sensor to record both depth and skeleton data of human daily activities. It is consisted of 20 unique activities done in five different environments: office, kitchen, bedroom, bathroom, and living room. Data is collected with four different people: two males and two females, recorded for about 45 seconds with each person. The dataset also consists of a random activity of each individual, which is not similar to any other activity done before.

B. Training GMM based HMM

When a DBN contains any hidden nodes, expectation maximization (EM) algorithm can be used to train parameters. In this project, we need to train the following parameters: the weight $w_{i,m}^n$, the mean $\mu_{i,m}^n$ and the covariance $\Sigma_{i,m}^n$ of the Gaussian mixture model for each joint and each pose state, the transition probability a_{ij} for $i, j = 1, 2, 3$. The training process is done for each activity separatively.

When applying EM algorithm, it is well known that EM algorithm only converges to a local optimum due to the none convex nature of the optimization equation. Therefore, initial parameters need to be properly initialized to minimize the effect of local optimum issue. The initialization process is explained as follows: For each activity, we consider the GMM for each skeleton joint separatively. We first use the function $mk\_stochastic()$ to randomly initialize and normalize the transition matrix A and the observation matrix B. The function $mk\_stochastic()$ is a matlab function that allows to learn HMM and infere with possibly noisy and uncertainty knowledge on hidden states. Then, we collect skeleton observation data of persons that are used to train the GMM based HMM model, and apply K-mean algorithm to cluster the 3D skeleton position information into different groups and derive the parameters of the Gaussian mixture model. After the initialization procedure, we run the expectation maximization (EM) algorithm to further estimate the model parameters to optimize the likelihood of the training set.

C. Activity Recognition

Once a HMM is trained for each action class, then the idea is to select the most likely activity given the observation sequence, i.e., choose among all the activity models which best matches the observations. Therefore, it can be formulated as follows. Given the observation sequence of skeleton joints $O = O_1, O_2, \ldots, O_t, \ldots$ and the set of models for each activity $\{\lambda_a = (A_a, B_a, \pi_a) | a = 1, 2, \ldots, 14\}$, in which a represent the a_{th} activity, how do we efficiently compute likelihood $P(O|\lambda_a)$, the probability of the sequence once the model is given. Intuitively, given λ_a, the likelihood can be calculated by the following equation:

$$\begin{aligned} P(O|\lambda_a) &= \sum_{\text{all } S} P(O|S, \lambda_a) P(S|\lambda_a) \\ &= \sum_{s_1, s_2, \ldots, s_T} \pi_{s_1} b_{s_1}(O_1) a_{s_1 s_2} b_{s_2}(O_2) \cdots a_{s_{T-1} s_T} b_{s_T}(O_T) \end{aligned}$$

Then the matching activity is chosen as

$$a^* = \underset{a}{\text{argmax}} P(O|\lambda_a)$$

However, the computation of the likelihood by this way is on the order of $V \times T \times N^T$, in which V is the number of activities. This calculation is too large to be solved. In order to tackle this problem, we introduce the forward algorithm to efficiently calculate the likelihood.

Specifically, consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, s_t = S_i|\lambda)$$

i.e., the probability of the partial observation sequence, $O_1, O_2, \ldots, O_t$ (until time t) and state S_i at time t, given the model λ. We can solve for $\alpha_t(i)$ inductively, as follows:

1)Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

2)Induction

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{j=1}^{N} \alpha_t(j) a_{ij} \quad \begin{aligned} 1 \ &\leq \ t \leq T-1 \\ 1 \ &\leq \ j \leq N \end{aligned}$$

3)Termination

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

By the forward algorithm, the likelihood can be derived more efficiently. For dataset with $V = 14$ activities, $N = 3$ states and $T = 1000$ (average) observation sequence a total of $V \times N^2 \times T = 126000$ computation is required for activity recognition. Clearly, this amount of computation is modest as compared to the capabilities of most modern computers.

V. Experiment

A. *Experiment Scenarios*

We have used two scenarios for examining the accuracy of the algorithm.

1)Unseen person: i.e. Leave one out cross validation is performed, that is, model is trained on three of the four subjects and test the model on the data from the fourth person.

2)Previously seen person: One half of the dataset is used for training while other half is used for testing. The training data and testing data are not overlapped.

Activity	Test on person 1	Test on person 2	Test on person 3	Test on person 4	Test on randomly
Still	100	100	100	100	100
Talking on the phone	100	100	100	100	78
Writing on white board	62	100	100	100	100
Drinking water	100	100	100	100	100
Rinsing mouth with water	100	100	100	100	100
Brushing teeth	38.5	100	100	53	58
Wearing contact lense	100	36.5	100	13.5	100
Talking on the couch	100	35.5	92.5	100	81.5
Relaxing on the couch	100	23.5	100	0	66.5
Cooking (chopping)	65.5	100	100	100	100
Cooking (stirring)	100	100	96.5	100	100
Opening pill container	100	100	35.5	100	100
Working on computer	100	100	100	100	100
Overall Average	89.7	84.27	94.2	82.4	91.1

Table I: Recognition accuracy with respect to individuals (1st joint data)

B. Experiment Results and Analysis

In the experiment, we first trained our GMM-HMM model using first half of the dataset ($T/2 = 100$) of each persons activity recordings. After training each activity separately, we obtained 13 probability transition matrices A, and observation transition probability matrices B. Then, we randomly selected data from second half of the dataset to test our recognition accuracy. Specifically, to choose samples randomly, we first randomly selected a person from the four people. Next, we chose the starting point of the sample using a random number, and length of testing sequence is a quarter of dataset ($T/4 = 50$), so that the selection operation is equivalent to adopt a sliding window with 50% overlapping. The number of random samples for training is 200. Finally, for each testing sample, we respectively calculated likelihood of this observation with estimated parameters from each activity. The likelihood was obtained using forward algorithm. The activity class with the largest likelihood is supposed to be the true activity.

Table I shows the results of the recognition accuracy just using features from the first skeleton. The number of pose state we used was 3, and joint state number is 5. The average overall accuracy is 91% if all testing samples are randomly selected from four persons. To evaluate individual recognition performance, we further conducted experiments where samples were just from one single person. We find testing samples from person 3 achieves the highest accuracy (94.2%) while test on person 4 gives the worst performance (82.4%). The result that recognition accuracy varies amongst individuals indicates a possible relationship between different people and recognition performance. If we could include more people in our training step, probably we could get a better result since more individuals diversities are considered.

This recognition result is better than the one given in our referenced paper. We think the main reason is that we changed joint state number from 3 in the paper to be 5. The reason that we made the change was that we could not explain that three states should give the optimum performance, while intuitively some other states could also yield a same, if not better performance. The following figure shows an approximate relationship between recognition accuracy and several different numbers of joint states or pose states. The joint number chosen from 3 to 7, and pose number selected as 2,3,or 4, is more likely to yield a higher accuracy.

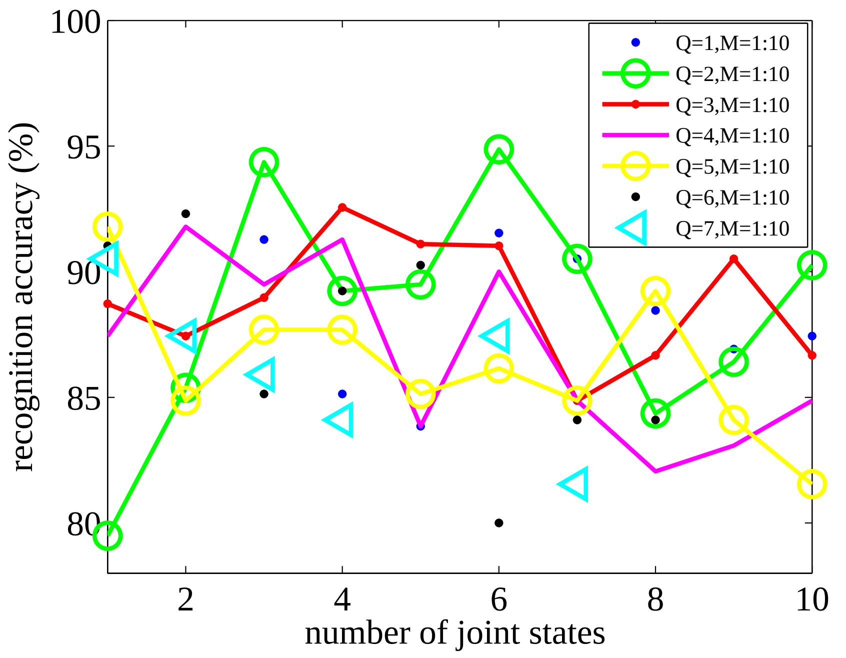

Figure 6: Simulation results of recognition accuracy with the number of joint state

Another factor we noticed that might affect the recognition accuracy is the selection of skeleton

Activity	Test on person 1	Test on person 2	Test on person 3	Test on person 4	Test on randomly
Still	100	100	100	100	100
Talking on the phone	100	100	100	100	100
Writing on white board	100	100	100	100	100
Drinking water	100	100	100	100	100
Rinsing mouth with water	100	100	100	100	85.5
Brushing teeth	83	100	100	100	80.5
Wearing contact lense	100	100	100	100	100
Talking on the couch	100	100	100	100	100
Relaxing on the couch	100	100	100	100	75.5
Cooking (chopping)	100	100	100	100	100
Cooking (stirring)	100	100	100	100	100
Opening pill container	100	100	100	100	100
Working on computer	100	100	100	100	100
Overall Average	98.7	100	100	100	95.6

Table II: Recognition accuracy with respect to individuals (2nd joint data)

joints. Table II gives the recognition accuracy of individuals using measurement from the second skeleton joints. The experiment was conducted with the same parameters as used in Table I. By inspection Table I and Table II, we find the result is surprisingly better than the first one, where we have an average recognition accuracy of 95.6%. One possible explanation is this skeleton probably conveys more information of activity features so that features themselves could have better distinctions.

However, in the unseen case as demonstrated in the referenced paper, our averaged recognition result (ranging from around 20% to 30%) is inferior to their results (78%). Although we had a low averaged recognition accuracy, the recognition accuracy could achieve 100% for some activities, such as Rinsing mouth with water and Cooking (stirring). Future work could turn to analyze the relationship of the same activity amongst different individuals so that we could achieve higher classification accuracies.

VI. Conclusion

In this report, we studied the GMM-HMM model for human activity recognition. Instead of using features from all skeleton joints mentioned in the referenced paper, we could achieve even higher activity recognition accuracy with a single skeleton node with a different joint state number. Then we conducted experiments to further investigate the relationship between recognition accuracy and different choices of state numbers. In our case, the joint number from 3 to 7, and pose number such as 2,3,or 4,tends to give a higher performance. Finally, we explored recognition performance with respect to different choices of skeleton jointssome skeleton joint simply shows an obviously superior performance (skeleton 2 in our case) than the other joints. This could provide insight into selecting more suitable joints for recognition purposes.

References

[1] L. Piyathilaka and S. Kodagoda, ”Gaussian mixture based HMM for human daily activity recognition using 3D skeleton features,” *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, Melbourne, VIC, 2013, pp. 567-572.

[2] Ghahramani, Z. 1998 Learning dynamic Bayesian networks. In *Adaptive Processing of Sequences and Data Structures*. Lecture Notes in Artifical Intelligence (ed. M. Gori and C. L. Giles), pp. 168197. Springer.

[3] Z. Ghahramani, ”An introduction to hidden Markov models and Bayesian networks,” *Int. J. Pattern Recognition and Artificial Intelligience*, vol. 15, no. 1, pp. 942, 2001.

[4] OpenNI organization, ”OpenNI User Guide”, November, 2010, Last viewed 19-01-2011 11:32, http://www.openni.org/documentation

[5] PrimeSense Inc., Prime Sensor NITE 1.3 Algorithms notes, 2010, Last viewed 19-01-2011 15:34, http://www.primesense.com

[6] Cornell Robot Learning Lab, http://pr.cs.cornell.edu/humanactivities/data.php

[7] Dynamic Baysian Network Toolbox, https://www.cs.ubc.ca/ murphyk/Software/

Notes

This model is an extension of HMM learnt in EECE 562 course. In addition to HMM and GMM models, in this project we were also involved with topics such as EM for parameter estimation, K-means clustering and forward algorithms to compute likelihoods in Markov chains. Although we met plenty of difficulties in this project, such as how to build models, extract features, initialize parameters in faced with this brand-new area, we could eventually find a way to the answer with background knowledge and group efforts.

Codes attached.

```
% Matlab code for EECE 562 project
% This code is to implement human activity recognition
%
% notes:
%       1. For table 1, use case1 and respectively change Itest from [1] to
%       [2],[3],[4], and [1 2 3 4]
%       2. Uncomment case 2 instead, and test unseen case
%       3. change node from 1 to 11 to try different skeletons

close all;
clear;clc

%% para. setting
Ifull=[1 2 3 4];    % full dataset of person 1 to 4

%## case 1 START ##
Itrain=[ 2 3 4 ];   % full dataset used for training,repace it with other set if necessary
Itest=[1];          % change this value respectively to 2,3,4
%## case 1 END ##

% %## case 2 START ##
% Itrain=[2 3 4];   % full dataset used for training,repace it with other set if necessary
% Itest=[1];
% %## case 2 END ##

nSamples=200;           % # of random samples to test

C={'still','talking_on_the_phone','writing_on_the_whiteboard','drinking_water',...

'rinsing_mouth_with_water','brushing_teeth','wearing_contact_lenses_2','talking_on_couch',...

'relaxing_on_couch','cooking_chopping','cooking_stirring','opening_pill_container','working_on_c
omputer'};
```

```
T=200;                    % length of time slices, first half for training, seond for testing
O = 3;    % 3d position
nex = length(Itrain); % eg. 3 person's data
M = 5;    % state# of a single joint
Q = 3;    % state# of activity    % change    Q to other number
node=1; % skeleton node
cov_type = 'full';
Data=zeros(3*length(C),T,length(Ifull));
A_all=zeros(Q,Q,length(C));      % prob. trans. mat. of each activity
B_all=zeros(Q,M,length(C));      % obser. prob. trans. mat.
p_all=zeros(Q,length(C));
mu_all=zeros(3,Q,M,length(C));
Sigma_all=zeros(3,3,Q,M,length(C));

loglik=zeros(1,length(C));
label_true=(1:length(C))'*ones(1,nSamples);    % true label
label_est=zeros(length(C),nSamples);      % recognitized label

%% training GMM-HMM model for each activity
% load and transform data for easier manipulation
for i=1:length(Ifull)
    load (['person_',num2str(i),'.mat']);
    for j=1:length(C)
        Dtmp=eval(cell2mat(C(j)));         % Dtmp: tempory variable, transform cell-named data
to variable
        Data((j-1)*3+1:j*3,:,i)=Dtmp(1:T,1:3)';
    end
end

% train GMM-HMM model
for j=1:length(C)
    clc;
% initial guess of parameters
    p0 = normalise(rand(Q,1));
    A = mk_stochastic(rand(Q,Q));

    data=Data((j-1)*3+1:j*3,1:floor(T/2),Itrain);
    [mu0, Sigma0] = mixgauss_init(Q*M, data, cov_type);
    mu0 = reshape(mu0, [O Q M]);
    Sigma0 = reshape(Sigma0, [O O Q M]);
    B = mk_stochastic(rand(Q,M));

  % train the model using EM
    [LL, p_est, A_est, mu_est, Sigma_est, B_est] = ...
```

```
            mhmm_em(data, p0, A, mu0, Sigma0, B, 'max_iter', 50);
  % save para.
        A_all(:,:,j)=A_est;
        B_all(:,:,j)=B_est;
        p_all(:,j)=p_est;
        mu_all(:,:,:,j)=mu_est;
        Sigma_all(:,:,:,:,j)=Sigma_est;
end

    fprintf('recognization in process, please wait: \n');
  %% testing stage -- random selected samples (unseen data)
  for j=1:length(C)
    % generate random testing subject
         subj_idx=randi(length(Itest),1,nSamples);
    % generate random starting point given subject
         start_idx=randi(floor(T/4),1,nSamples);      % for the 2nd half T/2, using slideing window
of .5 overlapping
    % generate random testing samples
         for k=1:nSamples
data=Data((j-1)*3+1:j*3,floor(T/2)+start_idx(k):floor(T/2)+start_idx(k)+floor(T/4)-1,Itest(subj_id
x(k)));
              for kk=1:length(C)                              % matching over all activities
                   A_tmp=reshape(A_all(:,:,kk),size(A_est));
                   B_tmp=reshape(B_all(:,:,kk),size(B_est));
                   p_tmp=p_all(:,kk);
                   mu_tmp=reshape(mu_all(:,:,:,kk),size(mu_est));
                   Sigma_tmp=reshape(Sigma_all(:,:,:,:,kk),size(Sigma_est));
                   loglik(kk) = mhmm_logprob(data, p_tmp, A_tmp, mu_tmp, Sigma_tmp, B_tmp);
              end

              [~,label_est(j,k)]=max(loglik);      % recognized activity

         end

  end
  % get accuracy
     accuracy=mean(label_est==label_true,2);
     clc;
     fprintf('recognition accuracy is: \n');
     disp(accuracy);
     fprintf('overall average accuracy is: \n');
     disp(mean(accuracy));

%% END of the main code
```