A Hybrid Unsupervised/Supervised Model for Group Activity Recognition

Tomoya Hirano
hirano.tomoya@ise.eng.osaka-u.ac.jp
School of Engineering
Osaka University
2-1 Yamadaoka, Suita, Osaka, 565-0871, JAPAN

Takuya Maekawa
maekawa@ist.osaka-u.ac.jp
Graduate School of Information Science and Technology
Osaka University
2-1 Yamadaoka, Suita, Osaka, 565-0871, JAPAN

ABSTRACT

The new method proposed here recognizes activities performed by a group of users (e.g., attending a meeting, playing sports, and participating in a party) by using sensor data obtained from the users. Note that such group activities (GAs) have characteristics that differ from those of single user activities. For example, the number of users who participate in a GA is different for each activity. The number of meeting participants, for instance, may sometimes be different for each meeting. Also, a user may play different roles (e.g., 'moderator' and 'presenter' roles) in meetings on different days. We introduce the notion of role into our GA recognition model and try to capture the intrinsic characteristics of GAs with a hybrid unsupervised/supervised approach.

Categories and Subject Descriptors: H.3.4 Information storage and retrieval: Systems and software.

Keywords: Activity recognition; group activity; pattern classification.

INTRODUCTION

Many existing activity recognition studies have focused on single user activities. However, because people live in social groups, they usually perform activities in a group (e.g., attending a meeting, playing sports, and participating in a party). Henceforth, we refer to such activities as 'group activities (GAs).' The existing single user activity frameworks may not be able to deal well with GAs. Assume that a user is attending a meeting and is taking notes. When we perform single user activity recognition focusing only on the user, we may be able to determine that she is taking notes. However, we cannot know that she is attending a meeting solely from her sensor data. Therefore, recently, group activity recognition (GAR), which recognizes GAs by using sensor data obtained from a group of users, has been attracting attention. Here, we define GAR as an approach that identifies a class of group activity that a group of users is performing.

GAs have the following three characteristics, and they make GAR difficult.
(1) The number of users who participate in a GA is different for each activity. For example, the number of meeting participants may differ for each meeting. That is, the number of dimensions of a feature vector constructed concatenating their sensor data features changes depending on the number of meeting participants. So, we should design a group activity recognition model (GAR model) that is capable of dealing with sensor data from any number of users.
(2) In a GA, each user plays different roles. In a meeting on a certain day, assume that user A *moderates* the meeting and user B makes a *presentation*. In the existing (single user and group user activity recognition) frameworks, a feature vector is constructed concatenating features extracted from both user A's sensor data and user B's sensor data in a time window. In a meeting on the next day, assume that user A makes a *presentation* and user B *moderates* the meeting. (Their roles are swapped.) A feature vector constructed on the next day is very different from that of the first day. (For example, a certain element value reflected by user A's sensor data is different for each day because she plays different roles on each day.) Therefore, when we use feature vectors obtained on the first day to train a discriminative classifier for GAR, the trained classifier cannot successfully recognize the second day feature vectors.
(3) Several existing group activity recognition methods require labels (teaching signals) that show which role each user is playing in a GA (e.g., 'moderator' and 'presenter' in 'meeting'). It is very costly to prepare such finely labeled training data.

When we employ the existing single user activity and group activity recognition frameworks to recognize GAs where there are changes in the numbers of participants and roles, we should prepare vast quantities of training data that were collected in various situations (e.g., a meeting with three participants, a meeting with four participants, a meeting where user A moderates, and a meeting where user B moderates). In this paper, we propose a new GAR model that can cope with the above problems. That is, the proposed model is robust against changes in the numbers of participants and roles. Also, the model does not require finely labeled training data. In this paper, we introduce the notion of role into our GAR model. A GA has several necessary roles. For example, a 'football' activity must have a 'goalkeeper' role and a 'meeting' activity must have a 'presenter' role. Also, in a group activity class, the ratio of the number of users who play a certain role to the total number of users remains fairly constant. For example, when 10 users attend a meeting, the ratio of the presenter user to the 10 users is $1/10$. When 15 users attend

ISWC'13, September 9–12, 2013, Zurich, Switzerland.

ACM 978-1-4503-2127-3/13/09...$15.00.
http://dx.doi.org/10.1145/2493988.2494348

a meeting, the ratio of the presenter user to the 15 users is $1/15$. As mentioned above, these ratios are not very different. We model GAs by employing such intrinsic characteristics of GAs with supervised machine learning approaches. For example, a 'meeting' activity is modeled so that the activity includes 'moderator' and 'presenter' roles and the ratio of the number of presenter users to the total number of users is about $x\%$. Here we recognize which role each user is performing in an unsupervised manner from her sensor data. This permits us to achieve GAR without detailed labels showing each user's role.

RELATED WORK

Many researchers in the field of wearable computers have tried to recognize single user activities by using body-worn sensors and sensors on mobile phones [5]. As described in the introduction section, recently, GAR studies have been conducted that employ small sensors and/or cameras [8, 4, 7, 3]. Many camera-based GAR studies detect users and then recognize the single user activity of each user (e.g., moderating and making presentation in a meeting). After that, the studies construct a feature vector, which is used to train GA models and to recognize GAs, concatenating the recognition results. Many mobile phone-based and body-worn sensor-based studies also recognize the single user activity of each user, and then construct a feature vector concatenating the recognition results. Or, the studies simply construct a feature vector concatenating features extracted from sensor data obtained from the users. Therefore, the existing GAR frameworks are vulnerable to changes in roles and the number of users. Also, several studies require labels for each user's role. Preparing such finely labeled training data is costly. This study assumes that labels are given that simply indicate the activities of a group of users, and we attempt to estimate which group activity the group of users is performing.

PROPOSED METHOD

We propose a new GAR framework that is robust against changes in the numbers of users and roles. Fig. 1 shows an overview of our architecture. The key feature of our approach is that our architecture includes a role classification model, which recognizes the role that each user in a GA is playing in an unsupervised manner. With this information, we construct a feature vector, which is an input of the activity models. Here we should design the feature vectors to be robust against changes in the numbers of users and roles. Also, we assume that we know who are the members of a given group activity. In general, the group members are defined as being nearby users (detected by Bluetooth signals from other members' phones) and they are included in each other's buddy lists.

Feature extraction

Our framework has two types of feature vectors: those used as inputs of the role classification model and those used as inputs of the activity models. Here we explain the first. At time t, a feature vector f_t^i is constructed concatenating features extracted from the ith user's sensor data. (The numbers of dimensions of f_t^i and f_t^j, for example, are identical.) We explain the features extracted in our experimental study later.

Role classification model

With the role classification model, we attempt to mitigate the changes in the numbers of participants and roles. To achieve this, the role classification model finds roles included in each group activity class in an unsupervised manner by clustering training data (feature vectors) for the activity class in advance. Feature vectors constructed from sensor data obtained when a user plays the same role may be similar. So, we cluster feature vectors (data points) corresponding to the group activity class of interest, and we assume that each output cluster corresponds to a role included in the activity class of interest. We employ a mixture of Gaussians to model the roles included in each group activity class. To construct the model, we extract feature vectors for each user's training sensor data corresponding to the activity class, and cluster the vectors (estimate the parameters of the Gaussians) by employing expectation maximization (EM) [2]. By doing so, we can obtain the roles included in a group activity in an unsupervised manner.

With the role classification model, which consists of a Gaussian distribution for each role, we estimate which role the ith user is playing at time t. Here, assume that a feature vector f_t^i is constructed from the ith user's sensor data at time t. We compute the probability with which the ith user plays the mth role of the nth activity class (R_m^n) as follows.

$$p(R_m^n|f_t^i) = \frac{p(f_t^i|R_m^n)p(R_m^n)}{p(f_t^i)} = \frac{p(f_t^i|R_m^n)p(R_m^n)}{\sum_{x,y} p(f_t^i|R_y^x)p(R_y^x)}$$
$$= \frac{p(f_t^i|R_m^n)\frac{1}{NM}}{\sum_{x,y} p(f_t^i|R_y^x)\frac{1}{NM}} = \frac{p(f_t^i|R_m^n)}{\sum_{x,y} p(f_t^i|R_y^x)},$$

where N and M show the numbers of activity classes and roles in each activity, respectively, and $p(f_t^i|R_m^n)$ is the likelihood of the nth activity's mth role (cluster) for f_t^i as follows.

$$p(f_t^i|R_m^n) = \pi_m^n \mathcal{N}(f_t^i, \mu_m^n, \Sigma_m^n),$$

where π_m^n is the mixture weight of R_m^n in the nth activity class, and μ_m^n and Σ_m^n show the mean vector and covariance matrix of R_m^n's cluster (multivariate Gaussian distribution), respectively. Also, we assume the prior density $p(R_m^n)$ to be $\frac{1}{NM}$. By using the equation, we can know that, for example, user A plays the 3rd role of the 'meeting' activity class 90% of the time, and plays the 1st role of the 'meeting' activity class 10% of the time.

Activity models

With the probabilities computed by the role classification model, we compute features and construct a feature vector f_t, which is an input for the activity models, by concatenating the features. To capture the intrinsic characteristics of a group activity, we compute the following features for each role R_m^n in the activity.

(1) Whether a user who plays R_m^n exists: This feature is closely related to estimating whether or not the users perform the nth activity class. This feature is computed as $\max_i(p(f_t^i|R_m^n))$. If there are one or more users who play R_m^n, this feature value may be close to 1. If not, this value may be close to 0.

(2) The ratio of the number of users who play R_m^n to total numbers of users: This feature is computed as

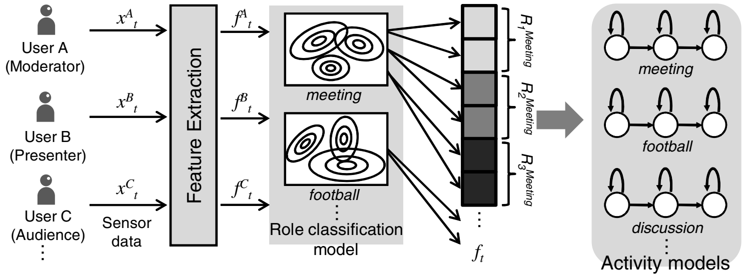

Figure 1. Proposed group activity recognition framework.

Table 1. Activities performed in our experiment.

	Activities		Activities		Activities
A	baseball	E	karaoke	I	walking
B	football	F	volleyball	J	bowling
C	meeting	G	discussion		
D	discussion (whiteboard)	H	eating lunch		

$\frac{1}{I}\sum_i p(f^i_t|R^n_m)$, where I shows the total number of group users. When the three users perform the nth activity class and only one user plays R^n_m, for example, this feature value may be close to $1/3$.

We compute the above features for each role and construct a feature vector f_t concatenating the features as shown in Fig. 1. For example, the first and second elements store feature values related to the first role included in the 'meeting activity' ($R_1^{meeting}$). The vector becomes an input of the activity models. We prepare a model for each group activity class by using a left-to-right HMM where the values of the observed variables correspond to the computed feature vectors (f_t), and we represent its output distributions by using Gaussian mixture densities. We employ the Baum-Welch algorithm [6] to estimate the HMM parameters. When we recognize test data (feature vector sequence) using the learned HMMs, we use the Viterbi algorithm to find the most probable state sequence in/across the HMMs [6]. From the state sequence, we can know into which HMM (activity class) a feature vector at time t is classified.

EVALUATION

Data set

A group of four experimental participants carried Google Nexus One in their pants pockets and took part in a session that included the sequence of GAs listed in Table 1. The participants took part in ten sessions in total. In the GAs shown as A to E in Table 1, their roles are clearly divided. For example, the 'football' activity has 'goalkeeper,' 'defender,' etc. We assigned a randomly determined role to a participant for each session. Throughout the session, the participant played the assigned role. On the other hand, in GAs F to J, no participant played a fixed role throughout a session. In the 'discussion' activity, for example, a participant usually switched her role from 'teller' to 'listener.' Also, to investigate the effects of the number of participants, groups of three participants and five participants took part in the sessions.

Here we describe how the activities were performed in the experiment. In activities A and B, the participants were assigned player positions (e.g., 'pitcher,' 'catcher,' 'strikers,' and 'defenders'). In activity C, the participants were assigned 'presenter,' 'moderator,' and 'audience' roles. In activity D, one participant wrote something on a whiteboard and discussed it with the other participants. In activity E, the participants were assigned 'singer' and 'audience' roles. In activity H, the participants had lunch together. In activity I, the participants walked around together. In activity J, the participants threw in turns.

In this study, our implemented application on a mobile phone collected the three kinds of sensor data described below. We also introduce features that constitute a feature vector f^i_t. We compute a feature vector for a two-second sliding time window with a 50% overlap.

- Three-axis acceleration data obtained from an accelerometer on the phone at about 16 Hz: Human motion is closely reflected in the acceleration data. We simply employ mean and variance values for each axis in a time window [5].
- Sound recorded by a microphone on the phone at 8 kHz: We extract sound pressure level and MFCC components from the sound data. In several activities, the sound pressure levels obtained from microphones may differ for each role. For example, in the 'meeting' activity, the sound pressure levels of a 'presenter' user may be higher than those of an 'audience' user. We compute the average sound pressure level in a sliding window and use it as a feature. Also, environmental sounds such as 'wind noise,' 'noise of a crowd,' and 'human voice' relate closely to human activities. In [1], the Mel-Frequency Cepstral Coefficient (MFCC) is reported to be the best transformation scheme for environmental sound recognition.
- Bluetooth scan data (received signal strengths and BSSID of Bluetooth modules on other mobile phones) obtained from a Bluetooth module on the phone: With the signal strength data, we can know how close two mobile phones (participants) are to each other. These data are useful for understanding GAs. For example, the distances between any two participants in the 'football' activity and 'volleyball' activity may be different. We obtain signal strengths from other participants and employ the average strength as a feature.

Evaluation methodology

To investigate the effectiveness of our approach, we also test the naive method that does not include the role classification model. That is, the naive method simply constructs a feature vector by concatenating the features from the users in a predefined order, and recognizes the vectors with the set of HMMs. As mentioned above, our method can model GAs with smaller quantities of training data than the naive method can. In other words, the naive method requires training data that are collected in various situations. So, we employ 'leave-n-session-out' cross validation, and change n to evaluate the methods. We regard n-sessions' sensor data as test data and the remaining sessions' data as training data, and we compute the classification accuracy of the test data. We iterate the procedure so that each combination of sessions is used as test data once.

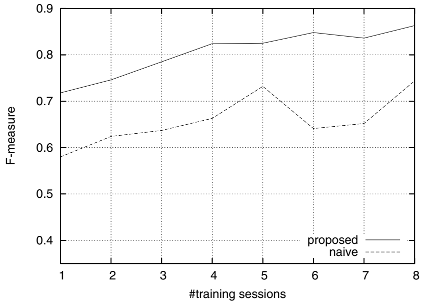

Figure 2. Transitions of accuracies when we change # of training sessions.

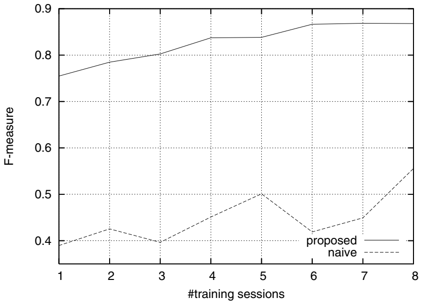

Figure 3. Transitions of accuracies related to GAs A to E shown in Table 1.

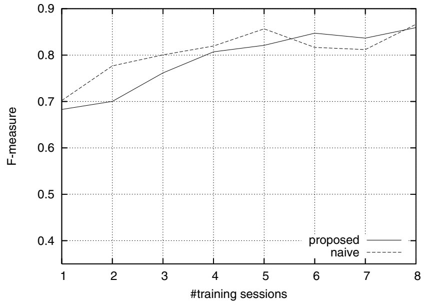

Figure 4. Transitions of accuracies related to GAs F to J shown in Table 1.

Results

Quantities of training data

To evaluate the performance of our method, we used F-measure ($\frac{2 \cdot precision \cdot recall}{precision + recall}$) calculated based on the results for the estimated class at each time slice. Fig. 2 shows transitions of the recognition accuracies (F-measure) of our method and the naive method when we increase the number of sessions used as training data. (# of participants was 4.) Our method could achieve about 80% accuracy even though we used only four-session data as training data. On the other hand, even when we used eight-session data as training data, the accuracy of the naive method was only about 75%. As above, we could confirm that our method can accurately recognize GAs with small quantities of training data.

Types of group activities

We investigated the results in detail. Fig. 3 shows transitions of the recognition accuracies related to GAs A to E shown in Table 1. In these GAs, the participants' roles were clearly divided. That is, in several cases, a role a certain participant played in the training sessions was different from that played in the test sessions. Because we design our method taking the problem into consideration, our method could greatly outperform the naive method that does not deal well with the problem. On the other hand, Fig. 4 shows the transitions of the recognition accuracies related to GAs F to J in Table 1. Both our method and the naive method achieved good accuracies related to such GAs. In these GAs, participants did not play fixed roles throughout a session. That is, because one-session data include sensor data for a participant who played various roles, these methods achieved good accuracies with only one-session training data.

Effect of number of participants

Even when the test and training data are obtained from different numbers of group users, our method can recognize the test data by using a GAR model trained with the training data. (The naive method cannot deal with such situations.) Fig. 5 shows the transitions of accuracies when we used training and test data obtained from different numbers of participants. (The x-axis indicates the number of training sessions.) For example, the '3-4' line shows the recognition accuracies when we used training data from three participants and test data from four participants. In many cases, we could achieve good accuracies (about 75%) even when test and training data were obtained from different numbers of group participants. However, when we used test data from three participants, the accuracies were somewhat poor. This may be because certain roles were not included in the test sessions related to some GAs. (Because the number of participants was small, they could not play several roles included in the training data obtained from many users.)

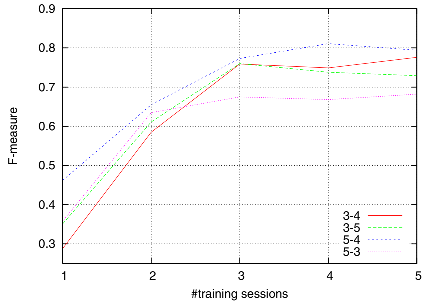

Figure 5. Transitions of accuracies when we used training and test data from different numbers of participants.

CONCLUSION

In this paper, we try to recognize group activities by employing sensor data obtained from a group of users. Group activities have several properties that make their recognition difficult. For example, the number of users who participate in a group activity differs for each activity. We designed a hybrid unsupervised/supervised model for group activity recognition that can deal well with the properties.

REFERENCES

1. Cowling, M. *Non-speech environmental sound recognition system for autonomous surveillance*. PhD thesis, Griffith University, 2004.

2. Dempster, A., Laird, N., Rubin, D., et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological) 39*, 1 (1977), 1–38.

3. Gordon, D., Hanne, J., Berchtold, M., Miyaki, T., and Beigl, M. Recognizing group activities using wearable sensors. *MobiQuitous 2012* (2012), 350–361.

4. Gu, T., Wu, Z., Wang, L., Tao, X., and Lu, J. Mining emerging patterns for recognizing activities of multiple users in pervasive computing. In *MobiQuitous 2009* (2009), 1–10.

5. Maekawa, T., and Watanabe, S. Unsupervised activity recognition with user's physical characteristics data. In *Int'l Symp. on Wearable Computers* (2011), 89–96.

6. Rabiner, L. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE 77*, 2 (1989), 257–286.

7. Wirz, M., Roggen, D., and Tröster, G. A methodology towards the detection of collective behavior patterns by means of body-worn sensors. In *UbiLarge Workshop at Pervasive 2010* (2010).

8. Zhang, D., Gatica-Perez, D., Bengio, S., and McCowan, I. Modeling individual and group actions in meetings with layered hmms. *IEEE Transactions on Multimedia 8*, 3 (2006), 509–520.