Chapter 6
Activity Recognition from Trajectory Data

Yin Zhu, Vincent Wenchen Zheng and Qiang Yang

Abstract In today's world, we have increasingly sophisticated means to record the movement of humans and other moving objects in the form of trajectory data. These data are being accumulated at an extremely fast rate. As a result, knowledge discovery from these data for recognizing activities has become an important problem. The discovered activity patterns can help us understand people's lives, analyze traffic in a large city and study social networks among people. Trajectory-based activity recognition builds upon some fundamental functions of location estimation and machine learning, and can provide new insights on how to infer high-level goals and objectives from low-level sensor readings. In this chapter, we survey the area of trajectory-based activity recognition. We start from research in location estimation from sensors for obtaining the trajectories. We then review trajectory-based activity recognition research. We classify the research work on trajectory-based activity recognition into several broad categories, and systematically summarize existing work as well as future works in light of the categorization.

6.1 Introduction

Mining the activities and patterns from the moving objects' trajectory data is an important challenge in today's society. This problem is becoming increasingly necessary as we accumulate large amounts of trajectory data in our daily lives. Recent development of social networks and maturing sensing technologies makes it possible to record sequences of location related data. Activity information hidden in these data can power services such as location-based recommendation systems, e-health and intelligent transportation. Activity recognition is a process to extract high-level activity and goal related information from low-level sensor readings through ma-

Yin Zhu, Vincent Wenchen Zheng and Qiang Yang
Hong Kong University of Science and Technology
e-mail: {yinz,vincentz,qyang}@cse.ust.hk

chine learning and data mining techniques. In this survey, we review and summarize existing literature on trajectory-based activity recognition, and project into the future in this active research area.

A natural categorization of trajectory-based activity recognition can be done by considering the different levels in which we place the activities. First, at the lowest level, we consider location-estimation techniques that take into sensor readings and produces estimates on the object's locations. This layer provides the important information input for the further activity recognition. Second, we survey representative methods that can determine trajectory segments, transportation modes and activities. The target output includes actions and goals. At the third level, we consider various applications that can be built based on activity data. Figure 6.1 shows a process view of these three levels.

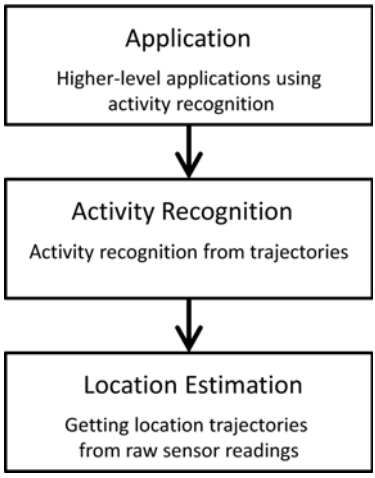


Fig. 6.1 A process view of trajectory-based activity recognition.

The activity recognition layer is essentially conducting information mining and compression from the huge spatial trajectories. To give a concrete impression the functionality of activity recognition, we use GeoLife, a location-based social-networking service developed by Microsoft Research Asia[1], as an example. GeoLife enables users to share their life experiences and build connections among each other using trajectory history data. One important component in Geolife is a transportation mode detection system [47], which classifies a segment of a GPS trajectory into one of { "walking", "driving", "biking" and "onBus"}. In this application, knowledge can be discovered from the trajectory data in the form of transportation mode. To the GeoLife system, users may ask queries like "how many times did I drive to work in this month?" Such questions can be answered by keeping a diary for the user. If an intelligent system can answer them directly from the user's GPS trajectories,

[1] http://research.microsoft.com/jump/79442

much manual effort is saved and mistakes avoided. Activity recognition also serves as an important component for higher-level applications in the GeoLife system. For example, a main goal of Geolife is to provide a trajectory sharing system that connects people using their trajectories as a medium. Transportation modes and other activities on the trajectories can be strong indicators for the similarity among people. This reveals another aspect of activity recognition: activity recognition serves as an important embedded component for other applications.

In this chapter, we consider **trajectory-based activity recognition** as a task that takes the sequences of sensor readings and context as input, and produces predictions of actions, goals, and plans. In particular, we first introduce location estimation techniques and focus on learning-based models. Then we consider various forms of activity recognition from trajectory data. We classify the current work into different categories, and survey existing research work under each category. We also present our views into the future.

6.2 Location Estimation for Obtaining the Trajectory Data

Before inferring activities for moving objects, we need to get their trajectory as a sequence of locations. In this section, we review location estimation techniques. In general, there are two categories of location estimation methods, including propagation models and learning based models.

6.2.1 Propagation Models for Outdoor Location Estimation

Global Positioning System (GPS) is the most widely used positing system in outdoor environments. It is based on a global navigation satellite system with at least 24 geo-synchronized satellites. A GPS receiver with four or more satellites in sight is able to know its location on the earth. In particular, the signal received from each satellite includes two pieces of information, including the coordinate of the satellite and the time stamp when the package is sent out. The distance between the GPS receiver and the satellite can be calculated by using the time difference between the sending time stamp and the receiving time stamp. After the GPS receiver knows the distances to four satellites, its location could be calculated by solving a distance equation system with three known parameters (latitude, longitude and altitude). This mathematical procedure is also called trilateration, where numerical root finding algorithms such as Newton-Raphson [4] are often used. The positioning accuracy for GPS can reach ten meters in an open area [14]. It is more accurate in an open area, and less accurate in cities with tall buildings. Because walls and other barriers block the satellite signals, GPS usually does not work in indoor environments. Besides G-

PS, there are also some other similar satellite positioning systems, such as Beidou[2] from China, and Galileo[3] from the European Union.

Though the GPS system is easy to use and relatively accurate in outdoor environments, it has obvious disadvantages. Most notably, it has difficulty working in indoor environments where signals from satellites are blocked or weakened. Other outdoor location estimation systems include GSM based localization, which uses cell-towers and their positions as basis for propagation model calculation.

6.2.2 Indoor Location Estimation using Learning-based Models

Since GPS hardly works in the indoor environment, researchers have considered alternative methods that work indoors and other complex environments. Among these methods, WiFi localization [1, 19] is one of the most mature and widely adopted solutions in research and practice. Even though, in theory, radio signal strength from a WiFi access point decays linearly with log distance, which allows a triangulation based method to identify the client mobile devices, in practice, it is difficult to obtain an accurate signal-propagation model in an indoor environment. This is because the physical characteristics of an environment, such as walls, furniture and even human activities, add significant noise to radio signal strength measurements. One particular problem in indoor environment is that the signals can reflect on the surfaces of certain materials, such as some walls. That means the signal strength the device receives is highly dependent on the local environment where the WiFi access points and the receiving device are. This scenario is referred to as the multiple path problem [1].

One intuitive idea to tackle this problem is to remember the signal strength behavior at different places in the environment, and train a learning-based method on these data. This method is commonly referred to as the fingerprinting technique. The following is an example illustrating this basic idea. When at place A, the device remembers the signal strength pattern. When this device arrives at A again, it knows that the scanned signal is most similar to one that is stored before. Therefore the device is able to infer its current location. However, there is much to be improved in this setting, e.g. how to collect fewer labeled points and achieve at the same accuracy, how to utilize the floor structure, and how to deal with the instability of WiFi signals.

A WiFi-enabled device, e.g. a laptop or a smartphone, can usually receive signals from multiple access points, which are installed at fixed positions in the environment, although their absolute positions may not be known. A WiFi access point is uniquely identified by its Media Access Control address (MAC address). The signal strength is called radio signal strength (RSS), which is a value usually spans from -120 dBm to -50 dBm. Once a mobile device scans the WiFi nearby, it gets a list of

[2] http://www.beidou.gov.cn/

[3] http://www.esa.int/esaNA/galileo.html

access points and the signal strength values to them. These received signal strength values can be represented as a vector $\mathbf{x} \in \mathbb{R}^d$, where d is the number of total access points and x_j is the RSS to the j-th access point. The fingerprints can be formally represented as a signal strength data set with locations as their labels $X = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where N is the total number of data examples. y_i denotes a location, which can be a discrete label indicating a "block" in an environment or a continuous coordinate pair such as $(p,q) \in \mathbb{R}^2$ in the 2-D case. There are some public WiFi data sets that can be used for indoor localization study. For example, one collection of data is provided by the IEEE 2007 ICDM data mining contest [43]. It contains a set of 5,333 fingerprint points collected from 247 locations in an office area of the HKUST academic building.

6.2.2.1 Nearest Neighbor Based Methods

RADAR [1] is one of the earliest system on WiFi localization. This project started as a localization system using only triangulation. But the multi-path problems led the researchers to use a fingerprint or learning-based method, which later became successful and highly influential. RADAR works in a test bed which is a 43.5m by 22.5m office area as shown in Figure 6.2. There are three base stations $BS1$, $BS2$ and $BS3$, which are similar to WiFi access points. A laptop is used as the mobile device to collect labeled signal strength at each location. Besides the signal strength SS, the device used in this project also uses one more strength called the signal-to-noise-ratio (SNR). For each location point (l_x, l_y) and each facing direction of the person holding the laptop $d \in \{N,E,S,W\}$, at least 20 pairs of SS and SNR to the three base stations are collected.

An important data preprocessing step in the RADAR system is that, it uses mean, standard deviation, and median of the signal strength values at each location for the location. After this, the data become less noisy. The data set contains many data tuples, each of which is represented as $(l_x, l_y, d, ss_i, snr_i)$ with $i = 1,2,3$. ss_i and snr_i are the mean values averaged over all the signal points collected under (l_x, l_y, d). The proposed location estimation method is based on nearest-neighbor search. A signal vector is matched to the nearest fingerprint vector sets in the feature space given some distance measure. Since these fingerprint points are already labeled, their location labels are used by average to predict the test signal vector's location. RADAR is an important early localization system with the meter-level accuracy, and it supports both learning-based method and radio propagation method for localization.

6.2.2.2 Bayesian Methods

Ladd et al. provide a major improvement to the RADAR system by further reducing the localization error to one meter for a similar test bed [19]. They propose a Bayesian model for indoor localization. Assume that the indoor space consists of n location states $S = \{s_1, \cdots, s_n\}$, where each state represents a location coordinate

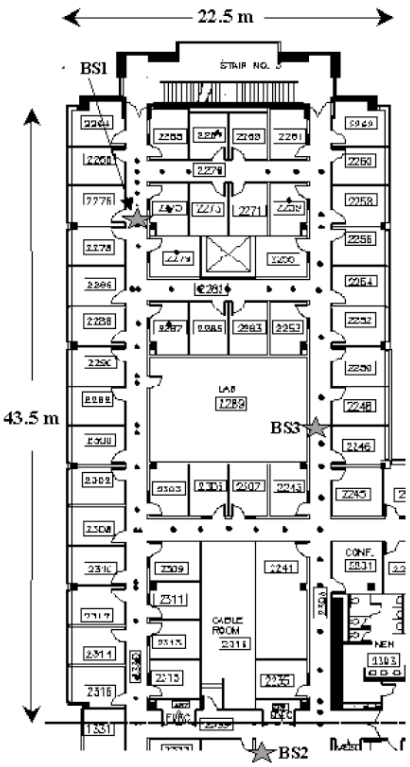


Fig. 6.2 The indoor map used in RADAR [1].

(l_x, l_y) with the facing direction d of the mobile user who holds a laptop in data collection. Given an observation o from a WiFi scan, the probability that it belongs to a state s_j is calculated as:

$$P(o|s_j) = \left(\prod_{j=1}^{N} P(f_j|s_i) \right) \cdot \left(\prod_{j=1}^{N} P(\lambda_j|b_j, s_i) \right),$$

where $P(f_j|s_i)$ is the probability that the frequency for j-th WiFi access point's frequency count f_j. $P(\lambda_j|b_j, s_i)$ measures the probability of received signal strength λ_j given the access point b_j and the state s_i. With this likelihood formula, one can calculate the posterior probability of the observation o on the location states:

$$P(s_i|o) = \frac{P(s_i) \cdot P(o|s_i)}{\sum_{j=1}^{n} P(s_i) \cdot P(o|s_i)}.$$

Ladd et al. further improve this Bayesian prediction by using the sequential constraints with a Hidden Markov Model (HMM) [34]. In practice, two consecutive

locations in an object's spatial trajectory are close to each other. Such a constraint can be naturally encoded in the HMM model as state transitions. An HMM is a state transition model $P(s_i|s_j)$ with observations $P(o|s)$. Given a sequence of observations, the states are hidden variables. The inference process gives the most probable state sequences, which is known as decoding. At inference time, observations and hidden states are both location states. HMM decoding is used as a smoothing procedure. In most applications, observations and hidden states are different. For example, sensor readings are the observations and the unknown activities are the hidden states. Other techniques such as Kalman filters and particle filters [37] are can also be used to model such sequential information. Interested readers are referred to Chapter 9 of [18] and [9, 8] for more details. The experimental results from this improved system also validate that using an HMM for post processing the predicted location states improves the error by 40%.

6.2.2.3 Summary

There are several directions to improve the above WiFi localization systems. For example, Pan et al. [31] present a manifold based method to recover both the mobile-device locations and the access point locations from labeled and unlabeled trajectories. Because unlabeled signal points are much easier to collect, this semi-supervised approach also reduces the manual labeling effort. One observation is that mobile devices and access points are spatially close to each other if their signal vectors are similar on some manifold structure. An illustration of the experimental result is shown in Figure 6.3. With only a few labeled WiFi signal points, the structure of the office area can be discovered by using many other unlabeled WiFi trajectories.

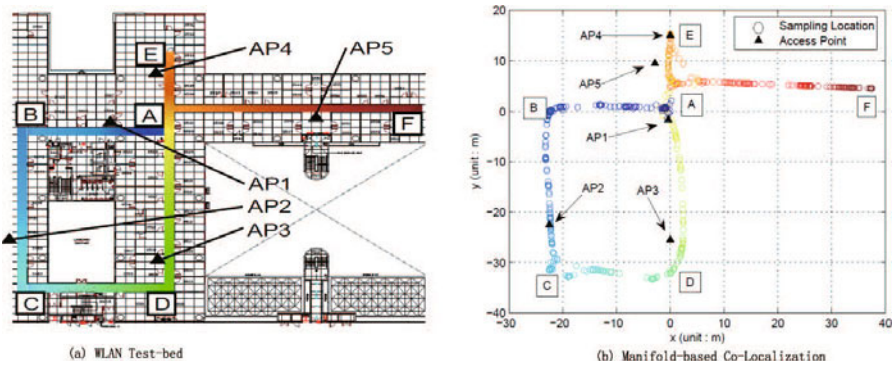


Fig. 6.3 Indoor environment layout and the manifold learning result in [31].

There has also been some work that uses additional sources such as signal sniffers with known locations or GPS to automatically annotate the data. For example, LANDMARC [30] is an indoor localization system that uses RFID sensors as refer-

ence points, whose locations are known. They are used to collect signal calibration data. The VTrack system [39] can use GPS to calibrate a WiFi localization system for outdoor usage. GPS is believed to be accurate in an outdoor environment. But there are cases where GPS is temporarily unavailable or the mobile device is forced to turn off GPS to save energy. Commercial systems such as Google WiFi localization and Skyhook also use the similar techniques. Table 6.1 compares the different learning-based localization methods regarding their calibration algorithms and the methods to collect labeled data.

Table 6.1 Comparison of the learning-based localization methods.

Project	Calibration algorithm	Method to collect data
RADAR [1]	KNN	manual
Improved RADAR [19]	Naive Bayes + HMM	manual
LANDMARK [30]	KNN	reference points
Geometry-based [31]	Semi-supervised learning	reduced manual
VTrack [39]	KNN + HMM	GPS Vehicle

6.3 Trajectory-based Activity Recognition

After getting the spatial trajectories, we need to build models to extract useful activity information from them. In this chapter, we categorize the existing work for trajectory-based activity recognition along two possible dimensions.

- The first dimension is the **user dimension**, where we categorize the existing work into **single-user activity recognition** and **multi-user activity recognition**. In particular, these two categories differ on whether the trajectory data are collected by multiple users and the user difference is modeled. If a model uses multiple users' data and considers the difference among the users in training the activity recognition model, then it belongs to multi-user activity recognition. Otherwise, it belongs to the single-user activity recognition category. Most of the existing work goes to the single-user activity recognition category, but we see a growing trend of multi-user activity recognition in the recent years as there are more and more users' activity data accumulate.
- The second dimension is the **learning method dimension**. Most of the existing work uses machine learning or data mining for activity recognition. For example, supervised learning methods use the trajectory data labeled with a predefined set of activity labels for training an activity recognition model. Unsupervised learning and frequent pattern mining methods aim to extract useful activity patterns directly from the trajectory data.

There are also some other possible criteria to categorize the existing trajectory-based activity recognition work, including venue (e.g. indoor and outdoor), sensors (e.g.

GPS, WiFi, cameras.), etc. However, the categories generated by using these criteria may not be mutually exclusive to each other. For example, an activity model that works in indoor environments may also work in outdoor environments. Therefore, we do not take these criteria into account, and only focus the user and learning method dimensions in categorization. We list some existing work that falls into these two categories in Table 6.2. In the following sections, we shall introduce some representative work within each category, and hopefully shed some light on the emerging research directions.

Table 6.2 Trajectory Activity Recognition Categorization.

User	Supervised methods	Unsupervised methods	Frequent pattern mining
Single-user	[44, 32, 25, 24, 45, 35, 47, 41]	[12, 10, 16]	[21, 29]
Multiple-user	[27, 42, 46, 23, 17]	?	?

6.3.1 Single-user Activity Recognition

In this section, we focus on single-user activity recognition and further categorize the existing work into three learning paradigms: supervised learning, unsupervised learning and frequent pattern mining.

6.3.1.1 Supervised Learning

In activity recognition, standard supervised learning algorithms take trajectory data, as well as their corresponding activity labels, as inputs. By training some classification (or regression) model with the given inputs, the supervised learning algorithm can use it to predict the activity labels for some test trajectory observations as output. We shall introduce some typical supervised learning models used by the existing work in this section.

Decision Tree with Sequence Smoothing and Hidden Markov Model

Sequence information is important in modeling the trajectory data for activity recognition. For example, Zheng et al. propose to use decision tree and sequence smoothing to detect a mobile user's transportation mode, such as "Drive", "Bus", "Bike" and "Walk", based on her GPS trajectory [47]. In order to predict the transportation modes, a GPS trajectory is segmented into a sequence of fix-sized time slices. Within each time slice t, raw GPS points denoted as latitude and longitude coordinates are used to calculate some features for a feature vector $\mathbf{x}_t$, such as

- *Heading change rate* measures the percentage of GPS points in a window that change their heading directions. As shown in Figure 6.4, for example, being constrained by the road, people driving a car or taking a bus cannot change their heading directions as flexible as if they are walking or cycling. Such heading direction changes can be different for different transportation modes.
- *Stop rate* measures the percentage of GPS points whose velocity values to their previous points are less than a threshold. This feature is inspired by the heuristic that the stop rate is usually higher for walking and lower for driving or taking a bus. Besides, a bus could also take more stops than a car, and thus taking a bus still has a higher stop rate than driving.
- *Velocity change rate* measures the percentage of GPS points with a velocity change percentage above a certain threshold within a unit distance as shown in Figure 6.5. This feature, together with the stop rate above, can be used to differentiate the transportation modes.

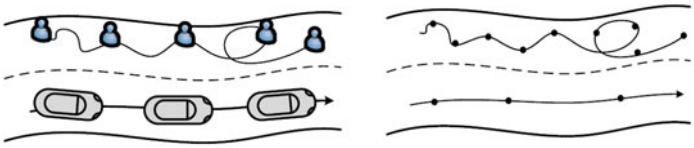


Fig. 6.4 Heading change rate of different modes.

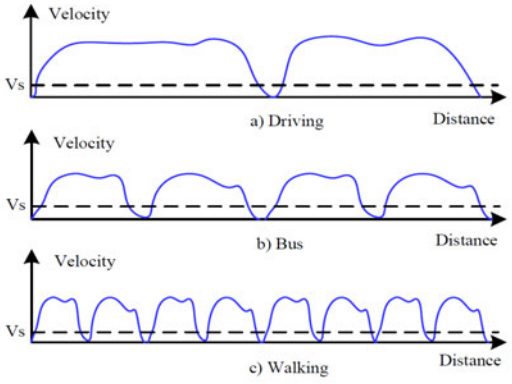


Fig. 6.5 Illustration for velocity change rate.

After the features $\mathbf{x}_t$ are extracted for each time slice, a decision tree is applied first to predict the transportation mode denoted as $y_t \in \{Drive, Bus, Bike, Walk\}$ in it. As such decision tree model does not take the sequence information into account, Zheng et al. further propose to model the location transition probability so that, if

two locations have high transition probability value, then the transportation modes for a mobile user to travel from one location to the other can be more consistent. Such location transition probabilities can be learned by clustering on the given GPS trajectory data in training. Finally, sequence smoothing is used to adjust possible misclassification in the inference step. The experimental results show that the system's overall accuracy archives 76.2% over the four transportation modes.

The proposed decision tree with sequence smoothing model can be seen as a variant of the Hidden Markov Model introduced in Section 6.2.2.2, except HMM learns the state emission probability $p(\mathbf{x}_t|y_t)$ and the state transition probability $p(y_{t+1}|y_t)$ at the same time. HMM is also extensively used in trajectory-based activity recognition. For example, HMM can be used to predict the taxi's activity state such as occupied and non-occupied. In particular, a taxi's GPS trajectory is segmented into a sequence of time slices. At each time slice, similar to the above mentioned transportation mode detection algorithm, some features such as the point-of-interest information around the trajectory, taxi velocity and heading change rate are extracted. Then, an HMM is used to formulate the taxi state transition probabilities and the state emission probabilities. An overall recognition accuracy of 75% is observed by using HMM, and it is close to the recognition accuracy from some human experts.

Dynamic Bayesian Networks

Dynamic Bayesian network is a natural extension to the standard Hidden Markov Model, and is also extensively used in many trajectory-based activity recognition problems. For example, Yin et al. develop a location-based activity recognition algorithm, which models the sensor-location-activity dependencies with some dynamic Bayesian network model [44]. In particular, as shown in Figure 6.6, they design a two-level Bayesian network that uses a sensor model to estimate the locations at the lower level and a goal recognition model to predict the user goals at the higher level. The shaded nodes *SS* denote some user's received signal strengths from the WiFi access points installed in the environment as shown in Figure 6.7. All the other variables, including the user's physical location L, the user action A and the user goal G, are all hidden. Their values are to be inferred from the raw signal data.

At the lower level of the DBN model, given the WiFi received signal strength values $SS_t =< ss_{t,1}, \ldots, ss_{t,N} > \in \mathbb{R}^N$ from N access points at time t, the proposed DBN model first predicts the mobile user's location L by using some Naive Bayes model $P(SS_t|L_{t,i}) = \prod_{j=1}^{N} P(ss_{t,j}|L_{t,i})$. Then, based on a series of location predictions, the model infers the user's action such as going to "Hall way 2" or "Elevator 3", etc. Such inference can be obtained by Expectation-Maximization as shown in [44]. At the higher level, the action sequences are taken as input to predict goals G. As the complexity of a DBN model is exponential in the number of hidden variables, the above estimations of these conditional probabilities can be computationally expensive. An alternative choice to infer these goals in the DBN is to use a separate N-gram model. Suppose that the actions $A_1, \cdots, A_T$ are inferred from the raw WiFi signal sequences. The most likely goal G^* can be obtained as follows:

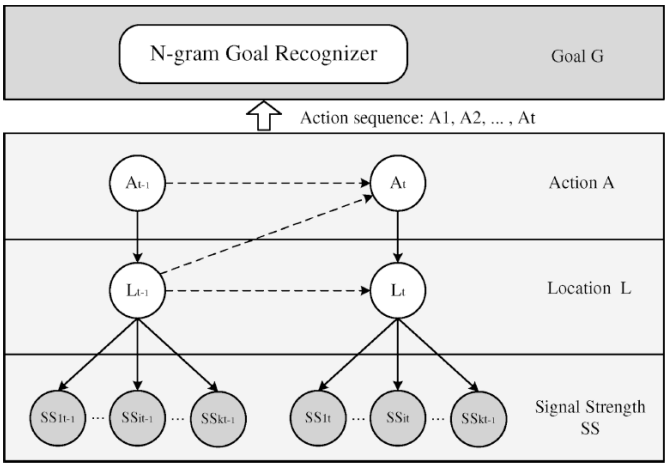


Fig. 6.6 The DBN + N-Gram model for activity recognition [44].

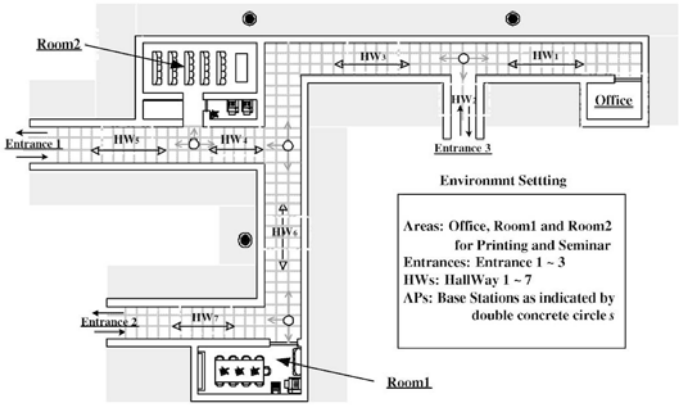


Fig. 6.7 The map for the office area of HKUST CSE department [45].

$$G^* = \underset{G_k}{\operatorname{argmax}} P(G_k|A_1, A_2, \cdots, A_t) = \underset{G_k}{\operatorname{argmax}} P(G_k|A_{1:t}).$$

By applying Bayes Rule, the above formula becomes:

$$G^* = \underset{G_k}{\operatorname{argmax}} \frac{P(A_{1:t}|G_k)P(G_k)}{P(A_{1:t})} = \underset{G_k}{\operatorname{argmax}} P(A_{1:t}|G_k)P(G_k),$$

where the conditional probability $P(A_{1:t}|G_k)$ can be simplified with a N-gram model:

$$P(A_{1:t}|G_k) = P(A_t|A_{t-1}, \ldots, A_1, G_k) \cdot P(A_{t-1}|A_{t-2}, \ldots, A_1, G_k) \cdots P(A_1|G_k).$$

Hence, an action A_t only depends on the goal G and its previous actions A_{t-1}, A_{t-2}, $\ldots, A_{t-n+1}$. When $n = 2$, the Bigram model is as follows:

$$G^* = \underset{G_k}{\operatorname{argmax}} P(G_k)P(A_1|G)\prod_{i=2}^{T} P(A_i|A_{i-1}, G_k).$$

The above inference's computational complexity is reduced to be linear in the number of goals and in the length of an action sequence, and thus can be efficiently performed.

Conditional Random Fields

Conditional Random Fields (CRF) is another important sequence learning model. Different from the generative models such as Hidden Markov Model and Dynamic Bayesian Networks, CRF is a discriminative model, and it is shown to perform well in various sequence labeling tasks including natural language processing and activity recognition. In this part, we introduce how CRF can be used in location-based activity recognition with GPS data [24].

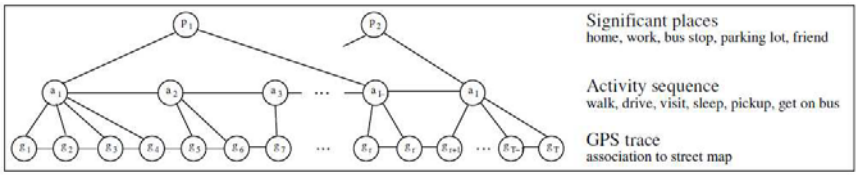


Fig. 6.8 Recognizing activities and significant places from the GPS trajectories.

As shown in Figure 6.8, Liao et al. show that they can employ a hierarchical model for location-based activity recognition. There are three levels in the model:

- **GPS trajectories** are sequences of some latitude and longitude coordinates in the map. As the raw GPS points can be noisy and less informative, a map-matching algorithm based on linear-chain CRF is employed to group the consecutive raw GPS points within every 10 meters into some segment, and further align it to some street patch in the map. Later, these street patches are used for feature extraction in recognizing the activities and significant places.
- **Activities** are estimated for the street patches obtained from the segmented GPS trajectory. The activities such as "walk", "drive" and "sleep", after being recognized, are further used to infer the significant places.
- **Significant places** are those locations that play a significant role in the activities of a person. Such places include a person's home and work place, the bus stops and parking lots the person typically uses, the homes of friends, stores the person frequently shops in, and so on.

The whole activity and significant place recognition task is decomposed into two sub-tasks, including a map matching process which groups the raw GPS data into some informative street patches, and a joint recognition process which detects the activities and the significant places together from the trajectory data. In each subtask, a CRF model is employed to formulate the dependencies among the GPS trajectory features and the hidden variables of activities and significant places.

Step 1: Map matching.

As the raw GPS data can be noisy, the observed user location may not be always accurate. For example, a user's position can be shown to be deviated from a street even she is in the middle of it. Consequently, some data processing with map matching becomes necessary. Liao et al. propose to employ a CRF model to implement the map matching. As a discriminative model, one of the most important things in using the CRF model is to design reasonable feature functions, which can capture the domain knowledge. Each feature function is defined on a variable node clique of the graphical model as shown in Figure 6.9.

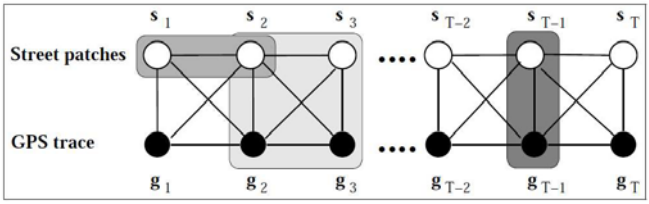


Fig. 6.9 Three types of features functions designed for map matching.

In particular, three types of feature functions are designed for this map matching subtask:

- **Measurement feature function** formulates the distance between a street patch s_{T-1} and a GPS point g_{T-1} for some time moment $t = T-1$, as shown in the dark grey part in the figure:

$$f_{meas}(g_t, s_t) = \frac{|g_t - s_t|^2}{\sigma^2},$$

  where $sigma > 0$ is used to control the scale of the distance. Generally, if the GPS point g_t is closer to the street patch center s_t, then the distance f_{meas} is smaller.
- **Consistency feature function** captures the temporal consistency of sequential GPS points, as shown in the light grey part in the figure:

$$f_{cons}(g_t, g_{t+1}, s_t, s_{t+1}) = \frac{|(g_{t+1} - g_t) - (s_{t+1} - s_t)|^2}{\sigma^2}.$$

This feature means that, if two GPS points (g_{t+1} and g_t) are close, their associated street patches (s_{t+1} and s_t) should be close too.

- **Smoothness feature function** constrain two consecutive street patches to be consistent, as shown in the medium grey part in the figure:

$$f_{smooth}(s_t, s_{t+1}) = \delta(s_t.street, s_{t+1}.street) \cdot \delta(s_t.direction, s_{t+1}.direction),$$

where $\delta(u,v)$ is an indicator function which equals 1 when $u = v$ and 0 otherwise.

With these feature functions, a CRF model formulates the likelihood function as:

$$p(s|g) = \frac{1}{Z} \exp\left\{ \sum_{t=1}^{T} w_m \cdot f_{meas}(g_t, s_t) + \sum_{t=1}^{T-1} (w_c \cdot f_{cons}(g_t, g_{t+1}, s_t, s_{t+1}) + w_s \cdot f_{smooth}(s_t, s_{t+1})) \right\},$$

where the model parameters w_m, w_c and w_s are to be learned by maximizing this likelihood function w.r.t. all the GPS sequences.

Step 2: Activity and significant place recognition.

After the map matching is ready, various useful features, such as average velocity, time of the day and so, can be extracted from each street patch. Such features are denoted as *evidence* in Figure 6.10, and further used to infer the activities and significant places with a new CRF model. This CRF, different from the one-layer CRF model used in map matching, contains two layers of labeling information about the activities and the significant places. Similarly, some feature functions are defined to capture the data dependencies. For example, one can design a measurement feature function for an activity a_i and the evidence's time feature t_i by $f(a_i, t_i) = \delta(a_i, Work) \cdot \delta(t_i, Morning)$. Similarly, a smoothness feature function can be defined on the transition of different activities: $f(a_i, a_{i+1}) = \delta(a_i, OnBus) \cdot \delta(a_{i+1}, Car)$. After having these features functions, one can train the CRF model by maximizing the data likelihood like map matching.

However, a major challenge in training this new CRF model is that in practice only the activity labeling is available while the significant place labeling is not, because the user usually does not label the significant places for his data. In other words, the training data for this CRF model only contain GPS street patches as the observations and the activities as the labels. In order to discover the significant places as well, Liao et al. propose to use an expectation-maximization (EM) algorithm to iteratively solve the problem. In each iteration, some initial guesses of the significant places are provided based on clustering with the observed activities. Then, these discovered significant places, together with the activities and the street patch evidences, are used to train the whole CRF model. The iteration continues until the inferred activities and the significant places do not change much.

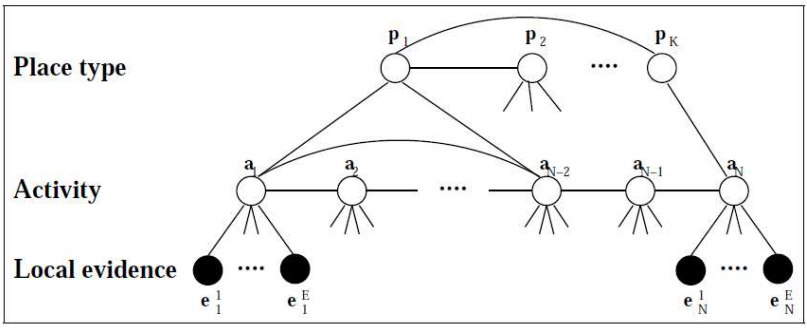


Fig. 6.10 The CRF for activity and significant place recognition [24].

Summary

Early work on supervised learning focuses on designing sophisticated learning models, e.g. the multi-layer DBNs in [25, 45]. A recent trend is paying more attention to incorporating domain knowledge as features, e.g. the heading change feature in [47] and the accelerometer data features [35]. It is believed that good features can lead to good activity recognition results, such as higher classification accuracy. Taking transportation mode as an example, Liao et al. designed a baroque DBN model to encode domain knowledge [25] while the recent research [47, 35] uses more domain-specific features. An additional advantage of the latter approach is that it is easier to implement and standardized.

6.3.1.2 Unsupervised Learning

Unsupervised learning is also used in trajectory-based activity recognition in order to discover some useful activity patterns. In general, this category of learning models take only the trajectory data as input and do not use any activity label information. Therefore, the outputs of these models are some data clusters [16] or low-dimensional feature representations [11]. Such outputs may not correspond to exact activity labels, but may imply some underlying activity patterns. In this section, we introduce three typical methods in this category: K-means clustering, Principal Component Analysis (PCA), and Latent Dirichlet Allocation (LDA). We use a representative project to introduce each kind of methods.

Clustering Methods

One intuitive unsupervised learning model is clustering, which can uncover structure in a set of data examples by grouping them according to some distance metric. Clustering methods are also used in activity recognition. For example, Huynh et al.

study how to use K-means clustering to rank individual features of accelerometer trajectory data according to their discriminative power in activity recognition [16]. In particular, the movement trajectory sensed by 3D accelerometers is usually segmented into a sequence of time slices, and the basic statistics values such as mean and variance are calculated as the features for each time slice. For different window sizes, many features could be defined. Huynh et al. use K-means clustering method to evaluate the effectiveness of each feature. Ideally, the data examples in each cluster should come from the same activity class. Such an intuition can be defined as:

$$P_{i,j} = \frac{|C_{i,j}|}{\sum_j |C_{i,j}|},$$

where $C_{i,j}$ is the set of examples in cluster i labeled with activity j. The cluster precision for activity j is defined as the weighted sum over different clusters:

$$p_j = \frac{\sum_i p_{i,j}|C_{i,j}|}{\sum_i |C_{i,j}|}.$$

If an activity has a cluster precision close to one, this indicates that there are many clusters mainly consisting of samples for this activity. Therefore, such a clustering process can help to decide which features to use for further activity recognition.

Principal Component Analysis

The MIT RealityMining project [11] aims to model conversation context, proximity sensing, and temporospatial location throughout large communities of individuals. Among others, it provides a very valuable dataset, which contains cellphone usage logs of one hundred MIT faculty members and students over nine months. The dataset is available online for other researchers to use[4]. For each user, the log records phone calls and short messages, phone status information, proximity to Bluetooth devices, and celltower IDs. The celltower ID indicates a coarse location of each user, based on which a user spatial trajectory for each user can be built. Figure 6.11 illustrates a single user's daily trajectories over 113 days. The celltower IDs are further associated with activity and status labels: Work, Home, No Signal, etc.

The Eigenbehavior project studies the patterns in the location trajectories for each user. For each user, a behavior dataset $\Gamma = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_D\}$ is built where D is the number of days and Γ_i is a H-dimensional binary vector. Each bit in the behavior vector Γ_i indicates whether one of the above five status states is one in a specific hour. Similar to face recognition research such as eigenface learning [40], we can get the average behavior of a user via Principle Component Analysis (PCA). In this analysis, we have $\Phi = \frac{1}{D}\sum_{i=1}^{D} \Gamma_i$, and we build difference behavior vectors as $\Phi_i = \Gamma_i - \Phi$. An eigenbehavior is then the eigenvectors of the covariance matrix AA^T where the

[4] http://reality.media.mit.edu/download.php.

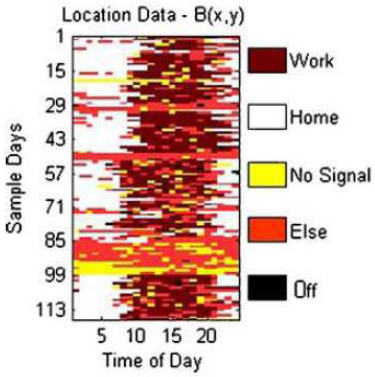


Fig. 6.11 A user's daily trajectories over 113 days [10].

matrix $A = [\Phi_1, \Phi_2, \ldots, \Phi_D]$. Figure 6.12 shows the top three eigenbehaviors of one subject in the dataset. The length of each vector is H, and is decomposed into five segments for visualization. We can clearly see that Eigenbehavior #1 represents a normal routine: at home from 20pm to 8am, at work from 8am to 20pm, and occasionally being at elsewhere. Thus, Eigenbehavior #1 represents a typical workday pattern. In contrast, Eigenbehavior #2 has large values at "elsewhere" part of the vector, which represents a weekend pattern.

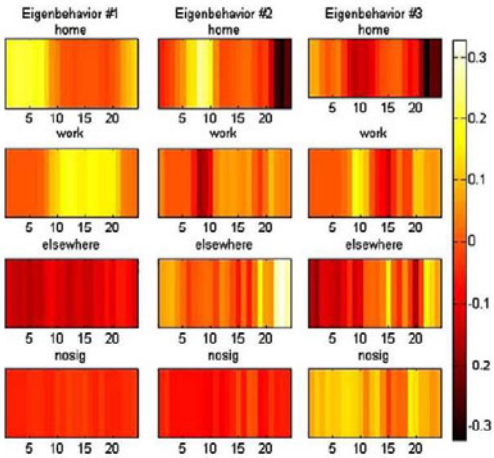


Fig. 6.12 The three eigenvectors generated by PCA [10].

Latent Dirichlet Allocation

Farrahi and Gatica-Perez [12] apply Latent Dirichlet Allocation (LDA), which is an unsupervised model, to extract the activity routines from the RealityMining data set. LDA is a probabilistic modeling method for unsupervised learning. It is originally developed to perform topic modeling on a text corpus [3]. Later, researchers found it useful for non-text data mining as well; e.g., it can be used for tagging images [2] and for accelerometer based routine recognition [15]. LDA is usually explained in the language of a text domain. Given a collection of text documents, where each document contains a set of words, the output of a LDA algorithm is to find K latent topics $Z = \{z_1, z_2, \ldots, z_K\}$, which each topic z_i is a distribution over words w_j via a conditional probability $P(w_j|z_i)$. Different topics favor different words. For example, the *politics* topic may have a larger value for the probability for words such as "president" and "war", while the *computer* topic may have a larger probability value for words such as "java" and "Microsoft".

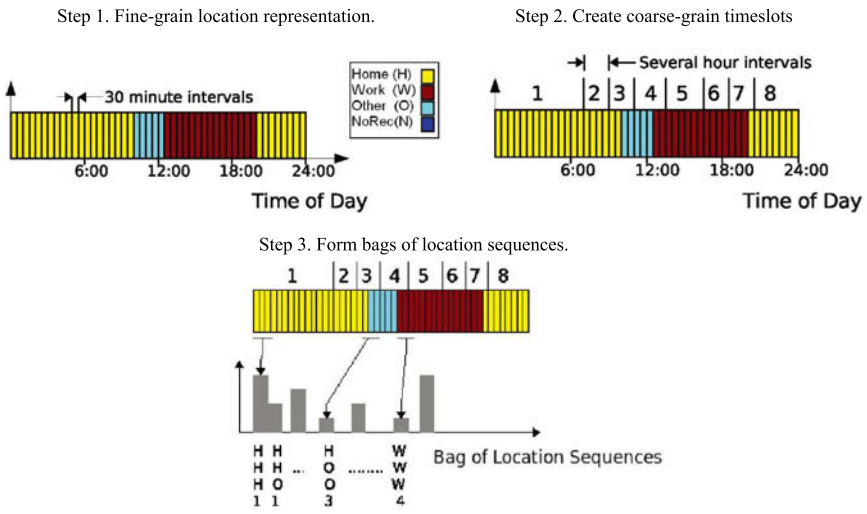


Fig. 6.13 Location words and documents building procedures [12].

In activity recognition, each day's sensor data is analogous to a document and each time slot represents a word. As shown in Figure 6.13, Farrahi and Gatica-Perez divide a day into 30-minute time slots and assigned each slot with a single location label that occurred for the longest duration. Here, the location labels include home, work, other and no-reception. Consequently, they treat each day's location data as a document, which consists of several location words. Each location word is a four-dimensional vector, containing three consecutive locations within a coarse time slot as well as the time slot ID. Then, the LDA algorithm was applied to find the activity

routines, or so-called topics, from the data. Each day can be seen as a mixture of the topics. The transition pattern is encoded as a word. The more frequent a word occurs, the more likely the pattern it encodes. Table 6.3 shows an example result of topic modeling. For example, the pattern "WOO7" means that there is a high transition probability from Work to other activities during the time of 7pm to 9pm. The words in each topic are also similar; for example the pattern words in topic 2 mainly represent Work-to-Other activity during 7-9pm.

Table 6.3 Activity routines illustrated in four topics [12].

Topic 2 - LDA		Topic 3 - LDA		Topic 23 - LDA		Topic 183 - LDA		Topic 171 - LDA	
Word	$p(w\|z)$	Word	$p(w\|z)$	Word	$p(w\|z)$	Word	$p(w\|z)$	Word	$p(w\|z)$
W W W 6	0.548	W W W 5	0.462	H H H 2	0.528	W W W 1	0.920	W W O 4	0.300
W O O 7	0.212	W W O 6	0.255	H W W 3	0.212	W W W 2	0.020	O W W 4	0.290
W W O 7	0.196	W O O 6	0.231	H H W 3	0.201	W W O 2	0.013	W O W 4	0.273
O O W 3	0.003	O W W 4	0.003	H H H 3	0.022	W O O 2	0.008	O O W 3	0.046
Work-Out 7-9pm		Work-Out 5-7pm		Home-Work 9-11am		Work in morning		Work-Out 9am-2pm	

Summary

Comparing the different unsupervised learning methods, we can see some interesting differences among them. From an algorithmic perspective, the working principle under the eigenbehavior method is PCA, which is a well-known statistical learning method for learning latent factors. LDA is a recent algorithm from the Bayesian learning community, which model may be easier to explain to people. From an application perspective, the input to eigenbehavior is numerical coding for the trajectory. The output is also numerical, which requires good knowledge of the inner side of the PCA algorithm to explain the result. Topic modeling uses preprocessed short trajectory words. Therefore, the output of the model is much easier to explain. The unsupervised segmentation method in [45] is different as it is only a data compression step to the higher level of supervised activity recognition. Thus, it does not require the output of the model to be explainable.

Although unsupervised activity recognition does not require labeled data, it is a challenge to explain the model and outcome of the unsupervised learning algorithm to people. For example, in [10], the outcome of the PCA model is a set of eigenvectors, which indicates that each day's user behavior can be seen as a linear combination of the some activity routines such as midnight to 9:00 at home, 10:00 to 20:00 at work, etc. This is a key insight why PCA can be used to extract and explain the behaviors. The transformation of daily activity representation in Figure 6.13 to the binary vectors is the key idea for PCA. In LDA , a challenge lies in how to define the pattern word.

6.3.1.3 Frequent Pattern Mining

Over the years, there has been much research work using association rule and frequent pattern mining for spatial and temporal data [7, 29]. Sequential activity pattern mining from trajectory data also belongs to this category. Here we shall take a recent study on mining the animal periodic patterns from trajectories [22, 21] as an example to start introducing this area. Figure 6.14 shows the location history of a bald eagle. Li et al. aim to analyze such a spatial trajectory and find some periodic activity patterns of the eagle [22]: for example, from December to March the eagle stays in the New York area, and some time later, it may leave to some other place etc.

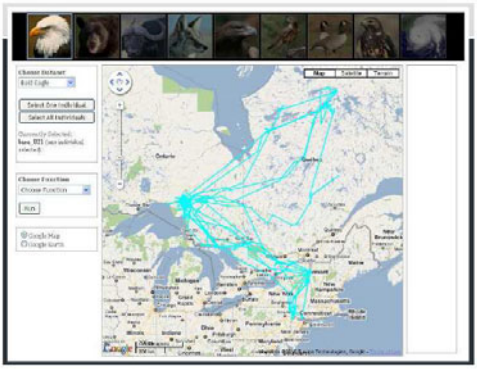


Fig. 6.14 One trajectory from the bald eagle [22].

In order to achieve this goal, Li et al. propose to first discover meaningful reference spots where the eagle often stays, and then try to detect the periods from the binary movement sequence (indicating whether the eagle is in this reference spot at some time t) within each reference spot. Finally, all the movement sequences across the reference spots with the same period will be put into some clustering algorithm in order to find the frequent patterns. The details for each step are given as follows:

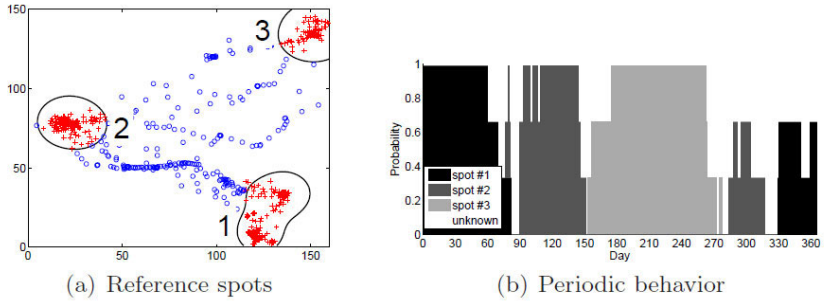
- *Finding reference spots*. In order to get the reference spots, they divide the map into $w \times h$ cells of the same size. For each cell c, the GPS data density can be calculated as:

$$f(c) = \frac{1}{n\gamma^2} \sum_{i=1}^{n} \frac{1}{2\pi} \exp(-\frac{|c - loc_i|^2}{2\gamma^2}),$$

  where $|c - loc_i|$ is the distance between cell center c and location loc_i. γ is a smoothing factor defined as $\gamma = \frac{1}{2}(\sigma_x^2 + \sigma_y^2)^{1/2} n^{-1/6}$. Therefore, if there are more GPS points from the eagle in this cell, the density is higher. Besides, if the GPS points are closer to the cell center, the density is also higher. After obtaining the density values, an reference spot can be defined by a contour line on the map, as shown in Figure 6.15(a), which joins the cells of the equal density

value, with some density threshold. The threshold can be determined as the top-$p\%$ density value among all the density values of all cells. The larger the value p is, the bigger the size of reference spot is.

- *Detecting periods on binary movement sequence*. Given a single reference spot, the eagle's movement sequence can be transformed into a binary sequence $B = b_1 b_2 \ldots b_n$, where $b_i = 1$ when the eagle is within the reference spot at time i and 0 otherwise. Such a sequence can be used to find the sequence period patterns (e.g. in days) by combining Fourier transform and autocorrelation. We refer readers to the details of such period discovery in the paper [22].
- *Mining periodic patterns*. Let $O_T = \{o_1, \ldots, o_d\}$ denote reference spots with the same period T such as a day. Given $LOC = loc_1 \ldots loc_n$, one can generate the corresponding symbolized movement sequence $S = s_1 \ldots s_n$, where $s_i = j$ if loc_i is within o_j. S is further segmented into $m = \lfloor \frac{n}{T} \rfloor$ segments so that each segment can represent a period such as a "day" if $T = 24$. Finally, the periodic activities can be found by using hierarchical agglomerative clustering to group these segments.



(a) Reference spots (b) Periodic behavior

Fig. 6.15 Reference spots and pattern distribution [22].

Figure 6.15(b) shows the daily behavior pattern of the bald eagle over one year, indicating that this eagle stays in New York area (i.e., reference spot 1) from December to March. In March, it flies to Great Lakes area (i.e., reference spot 2) and stays there until the end of May. It flies to Quebec area (i.e., reference spot 3) in the summer and stays there until late September. Then it flies back to Great Lake again staying there from mid-October to mid-November and goes back to New York in December.

6.3.2 Multiple-user Activity Recognition

As mobile devices and sensors become extensively available, there are more and more data collected from different users. In the past, people have considered using

all the users data together to train an activity recognition model. For example, Liao et al. pool all the users' GPS trajectory data together and train a hierarchical activity model to predict some transportation routine activities [25]. Such a method may work well if there is no big difference among different users' activities. However, this is not always true. For example, in WiFi-based activity recognition, a user visits the coffee shop for meal and the other just enjoys sitting in its outdoor couches to read research paper. These two users are very likely to observe similar WiFi signals, but their activities are quite personalized. This motivates us to consider the user-user relationship in utilizing such multi-user data for activity recognition. In the following, we introduce some existing work that falls into this multi-user activity recognition category.

Coupled Hidden Markov Model for Concurrent Activity Recognition

Wang et al. [42] provide a concurrent activity recognition system based on coupled Hidden Markov Model. In particular, each user $u_i, 1 \leq i \leq n$, has a sensor observation sequence $\{o_1, o_2, \cdots, o_T\}$ of length T. Each observation o_i has a label from a set of m activities $L = \{y_1, y_2, \cdots, y_m\}$. One can use a standard HMM to model each user's activity with the hidden states as the activity labels and the corresponding observations as the observations. To model the interaction and influence between users along the time, Wang et al. use the Coupled Hidden Markov Models (CHMM) to formulate the state transitions among different users. Figure 6.16 shows a CHMM for two users A and B. The two hidden state sequences $\{a_i\}$ and $\{b_i\}$ represent the activity sequences of user A and user B. The states from A and B are cross linked to capture the interactions between the two users. For example, the activity of B at time t is influenced not only by B's state at time $t-1$, but also A's state at time $t-1$.

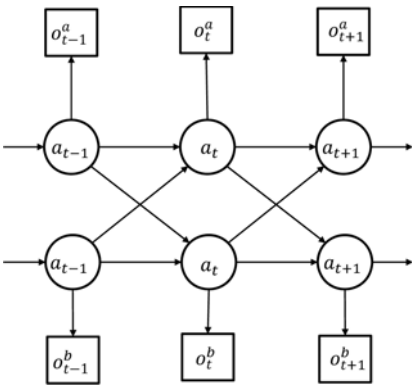


Fig. 6.16 A two-user Coupled Hidden Markov Model (CHMM) for activity recognition [42].

After such a CHMM model is built from the multi-user training data, the most probable label sequence for an observation sequence is $\arg\max_S P(S|O)$, which can be effectively solved by dynamic programming. Notice that in this work, although the training phase takes the concurrent activities of different users into consideration, the inference procedure only works for a single observation sequence.

Ensemble Learning

The DarwinPhone project takes a different approach towards multi-user activity recognition [27]. Different from [42], its inference is based on multi-user, i.e. when doing online activity recognition on one user's phone, it also uses the information from all the other nearby phones' data. In practice, different phones may have very different sensing information even at the same environment because of their positions and settings. For example, a phone inside a handbag may sense that its environment is quiet, but the other phone open in the air may sense a louder environment. In this case, using the activity recognition models on other nearby phones is very helpful to get more stable and more accurate recognition result. In the proposed method, each phone has a separate model for some activity. For some activity event, its nearby phones can pass the extracted features from their own phone sensors to it. Then, each received feature vector are used to generate an activity prediction, and finally a voting mechanism is used to decide which activity it is.

Transfer Learning

An important problem in activity recognition is that, every time when we want to build an activity recognition model for a new environment, we have to collect a labeled data set in it. This is because most activity recognition models are specific to each data set on which they are trained. If the training data set changes, for example given different sensors installed and different users having different activity patterns, the learned model in one house may not be applicable in another house. An interesting question to ask is that, is it possible to use some existing labeled data sets of various houses to help learn the parameters of a model applied in a new house? If possible, many data labeling efforts can be saved. Kasteren et al. give a positive answer to the above question by providing a transfer learning solution [17].

In the transfer learning based activity recognition task, there is a target house for which there is little or no training data available, and a number of source houses for which there is a lot of labeled data. The activities to be recognized in each house are the same, while the sensors used in each house are different. The task's goal is to use the source houses' data, together with the target house's possibly available training data, to train an activity recognition model for the target house. In order to achieve this goal, two challenges need to be addressed, including:

- The sensors used in different houses can be different;
- The activity patterns from multiple users in different houses can be different.

The first challenge leads to different feature spaces in different houses' data. In order to address this challenge, Kasteren et al. propose to introduce some meta-feature mapping function, which can map different sensor sets into a single common feature set that can be used for all houses. In particular, each sensor is described by one or more meta features, for example, a sensor on the microwave might have one meta feature describing that the sensor is located in the kitchen, and another that the sensor is attached to a heating device. The second challenge describes the activity differences given a series of similar sensor observations. For example, one person might often have cereal for breakfast, while another prefers toast, though they can trigger a series of similar sensor events w.r.t. their similar moving trajectories in the houses. Such activity differences require different sets of parameters to allow the model to recognize the corresponding activities. Therefore, Kasteren et al. propose to use a separate model for each house, and further assume that the target house's model and the source houses' model parameters share some common prior distribution. By learning the prior distribution from the source houses, one can use it to provide a reasonable initial value for the target house's model, and meanwhile make the new model be able to capture the unique activity patterns in the target house.

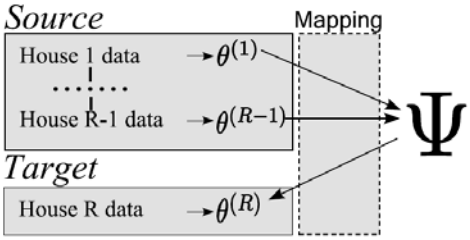


Fig. 6.17 The illustration for the transfer learning algorithm [17].

The detailed transfer learning model is illustrated in Figure 6.17. In each house, a Hidden Markov Model (HMM) is used to model the sensor sequence data together with the activities. Let us denote $\mathbf{x}_t$ as a sensor feature vector at time t, and y_t as its corresponding activity label. Here, each house's sensors are mapped to some common features; in other words, the feature spaces of $\mathbf{x}$ at each house, after the meta-feature mapping, are now the same. A HMM formulates the generative probability

$$p(\mathbf{y}_{1:T}, \mathbf{x}_{1:T}) = p(y_1) \prod_{t=1}^{T} p(\mathbf{x}_t | y_t) \prod_{t=2}^{T} p(y_t | y_{t-1}),$$

where $p(y_1)$ is the prior state distribution parameterized by π, $p(\mathbf{x}_t | y_t)$ measures the emission probability parameterized by B and $p(y_t | y_{t-1})$ measures the transition probability parameterized by A. Therefore, a HMM is denoted as $\theta = \{\pi, A, A\}$. Notice that each house has a HMM parameterized as θ_j, the proposed transfer learning model assigns some common prior distributions to these HMM parameters θ_j. For example, Kasteren et al. design each prior state probability as $p(y_1) = \prod_{i=1}^{K} \pi_i^{\delta(y_1 - i)}$,

where K is the number of states, $\delta(s) = 1$ if $s = 1$ and 0 otherwise. Then, they force such prior state probabilities from different HMM's to share a same Dirichlet prior distribution, which is given by $Dir(\pi|\eta) = \frac{\Gamma(\sum_{k=1}^{K}\eta_k)}{\Gamma(\eta_1)...\Gamma(\eta_K)}\prod_{k=1}^{K}\pi_k^{\eta_k-1}$ with parameters η. Here, $\Gamma(\cdot)$ is a Gamma distribution and we refer readers to the details in the paper. As the parameters η are shared across different houses, one can use different houses' data to learn them and thus be able to estimate the prior state distribution $\pi^{(j)}$ at each house j. Similarly, Kasteren et al. also assign a beta prior distribution to the emission probability and a Dirichlet prior distribution to the transition probability. These prior distributions' parameters are then learned using multiple houses' data and used to estimate the HMMs for each house [17].

Finally, three real world data sets are used to evaluate the proposed transfer learning solution. The experimental results show that, the proposed method can give good performance in activity recognition for a house with little or no labeled data, and generally outperforms some competing baselines. The datasets collected by the authors are also available on line[5].

Factorial Conditional Random Fields

One important application of using multiple users' data at the same time is to model the concurrent activities among the users. For example, Lian and Hsu develop a methodology to recognize concurrent chatting activities from multiple users' audio streams [23]. In order to capture the dynamic interactions, they adopt a factorial Conditional Random Fields model to learn and recognize concurrent chatting activities.

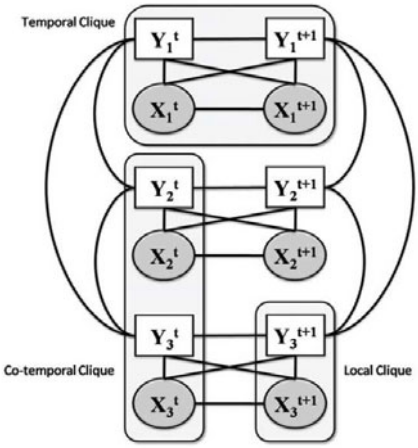


Fig. 6.18 A sample FCRF of 3 concurrent chatting activities [23].

[5] https://sites.google.com/site/tim0306/datasets

Figure 6.18 shows a sample FCRFs model for the recognition of concurrent chatting activities among three users. Denote X as the acoustic feature variables and Y as the set of chatting activity variables. Let $x_i^t \in X$ denote an observed acoustic sensor feature value at time t for chatting activity i from a user. Let y_i^t be the corresponding state of chatting activity i. Let $G = (V, E)$ be an undirected graph structure shown in Figure 6.18, with V as the node set and E as the edge set. For any given time slice t and chatting activity i, Lian and Hsu build edges $(Y_i^t, Y_i^{t+1}) \in E$ to represent the possibility of activity state transition across time slices. They also build edges $(X_i^t, Y_i^t) \in E$ to represent the possible relationships between activity labels and acoustic observations. Besides, edges $(Y_i^t, Y_j^t) \in E$ are built to represent the possibility of co-temporal relationships between any two concurrent chatting activities i and j. That is, all the hidden nodes within the same time slice are fully connected.

An FCRF model allows us to design various feature functions in formulating the conditional probabilities $p(y|x)$ for a sensor feature x and its chatting activity state y. In particular, based on the propoesd FCRF model shown in Figure 6.18, Lian and Hsu propose to design three types of feature functions:

- Local potential function $\phi_i^A(x_i^t, y_i^t, t) = \exp\left(\sum_{p=1}^{P} w_i^{(p)} f_i^{(p)}(x_i^t, y_i^t, t)\right)$, where $f_i^{(p)}$ is a function indicating whether the state values are equal to the p-th state combination within the local clique (x_i^t, y_i^t). $w_i^{(p)}$ are some weights to learn later.
- Temporal potential function

$$\phi_i^B(x_i^t, y_i^t, x_i^{t+1}, y_i^{t+1}, t) = \exp\left(\sum_{q=1}^{Q} w_i^{(q)} f_i^{(q)}(x_i^t, y_i^t, x_i^{t+1}, y_i^{t+1}, t)\right),$$

  where $f_i^{(q)}$ is a function indicating whether the state values are equal to the q-th state combination within the temporal clique $(x_i^t, y_i^t, x_i^{t+1}, y_i^{t+1})$. $w_i^{(q)}$ are some weights to learn later.
- Co-temporal potential function

$$\phi_i^{\Delta}(x_i^t, y_i^t, x_j^t, y_j^t, t) = \exp\left(\sum_{r=1}^{R} w_{ij}^{(r)} f_{ij}^{(r)}(x_i^t, y_i^t, x_j^t, y_j^t, t)\right),$$

  where $f_{ij}^{(r)}$ is a function indicating whether the state values are equal to the r-th state combination within the co-temporal clique $(x_i^t, y_i^t, x_j^t, y_j^t)$. $w_{ij}^{(r)}$ are some weights to learn later.

Finally, an FCRF model formulate the following conditional probability for N users' concurrent chatting activities:

$$p(y|x,w) = \frac{1}{Z(x)} \cdot \left(\prod_{t=1}^{T}\prod_{i=1}^{N} \phi_i^A(x_i^t, y_i^t, t)\right) \cdot \left(\prod_{t=1}^{T-1}\prod_{i=1}^{N} \phi_i^B(x_i^t, y_i^t, x_i^{t+1}, y_i^{t+1}, t)\right) \cdot \left(\prod_{t=1}^{T}\prod_{i,j} \phi_i^{\Delta}(x_i^t, y_i^t, x_j^t, y_j^t, t)\right),$$

where w denote the parameter set of $w_i^{(p)}$, $w_i^{(q)}$ and $w_{ij}^{(r)}$. $Z(x)$ is a normalization factor. By learning the parameters w based on the observed audio sequence data, one can predict the chatting activity state of a particular user at each time stamp. We refer interested readers to the technical details of learning and inference of such an FCRF model in [23]. Some experiments based on two concurrent chatting activity data sets show that, the proposed FCRF model is consistently better than some competing baselines such as Coupled Hidden Markov Model [5] in the comparison of F-score.

Latent Aspect Model

In [46], Zheng and Yang propose a user-dependent aspect model to help the users collaboratively build an activity recognition model that can give personalized predictions. Rather than simply pooling multiple users' data together, the proposed model introduces user aspect variables to capture the user grouping information from their data. As a result, for a targeted user, the data from her similar users in the same group can also help with her personalized activity recognition. In this way, one can greatly reduce the need of much valuable and expensive labeled data required in training the personalized recognition model.

In a WiFi environment, multiple users collect wireless signal data with activity annotations for around a month. The data format is in a set of quads: $\{\langle a_i, u_i, f_i, t_i\rangle | i = 1, ..., L\}$, where a is an activity, u is a user, and f is a feature observed at time t. In the WiFi case, a feature corresponds to a wireless access point (AP) that the mobile device can detect. A data record quad indicates that a user u is doing an activity a at time t, and meanwhile her wireless device detects some AP f. The goal is to build a personalized activity recognition model by using these data, so that with a user's WiFi observations at some time, we can predict what she is doing.

The proposed model extends the standard aspect model [13] by introducing user aspects, as well as time aspects and feature aspects to model personalized activity recognition from time-dependent sensor data. Figure 6.19 depicts the graphical model. The shadow nodes for user variables u, time variables f, feature variables f and activity variables a are observations. The blank nodes inside the rectangle are latent aspect variables. The user latent aspects $Z_u \in \{z_u^1, z_u^2, ..., z_u^{D_u}\}$ are discrete variables, indicating D_u user clusters. The model adopts such a *user-cluster-activity* hierarchy to help the users to collaboratively build an activity recognizer. In contrast to a two-tier *user-activity* hierarchy where each user can only rely on herself to do activity recognition, the proposed model can make the users from a same cluster to contribute together to train the recognizer from their feature and time observations. Therefore, even if some user has limited data to train an activity recognition model, she can still benefit from other similar users in the same group(s). As each user can belong to multiple user clusters at the same time with different probabilities, they actually contribute differently to each user cluster in training the recognition model and consequently get different predictions in real-time recognition. This helps to achieve the model personalization.

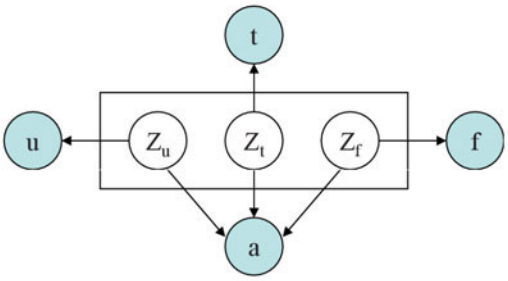


Fig. 6.19 User-dependent aspect model.

Some other latent aspects $Z_f \in \{z_f^1, z_f^2, ..., z_f^{D_f}\}$ and $Z_t \in \{z_t^1, z_t^2, ..., z_t^{D_t}\}$ are also introduced to encode the data observations on feature and time. They are used to capture the dependency between activities and observations, considering that similar feature observations at similar time periods are likely to imply some same activity. Note that these aspects do not necessarily rely on users. One can also take all the users' data as input, and only use them (i.e. Z_f and Z_t) to build a user-independent model for general activity recognition. However, such a user-independent model, as shown in their empirical experiment, does not perform as well as the user-dependent model. The user latent aspects help to achieve personalization, so it is called a user-dependent aspect model.

In general, the proposed aspect model is a generative model, which uses the latent aspect variables to explain the observations. It specifies a joint probability of the observed random variables:

$$P(a,u,f,t) = \sum_{Z_u,Z_f,Z_t} P(a,u,f,t,Z_u,Z_f,Z_t), \tag{6.1}$$

where $P(a,u,f,t,Z_u,Z_f,Z_t)$ is expanded, according to the graphical model, as follows:

$$\begin{aligned} P(a,u,f,t,Z_u,Z_f,Z_t) &= P(Z_u)P(Z_f)P(Z_t) \\ &\quad P(u|Z_u)P(f|Z_f)P(t|Z_t)P(a|Z_u,Z_f,Z_t). \end{aligned} \tag{6.2}$$

Here, the user variables u, feature variables f and activity variables a are all discrete in nature, so their conditional probabilities on latent aspects can be modeled easily by multi-nominal distributions. One exception is the time, which could be continuous. To formulate $P(t|Z_t)$, we discretize the time t with two possible strategies. One is "ByHour", which segments the time into hours. The other is "ByPeriod", which segments it into larger time periods; for example, one can define five periods, including morning (7am∼11am), noon (11am∼2pm), afternoon (2pm∼6pm), evening (6pm∼12am) and night (12am∼7am).

Summary and Outlook

We have introduced some recent research in multiple-user activity recognition. Multiple-user activity recognition is still an ongoing research topic. Most of the existing research work falls into the supervised learning category, leaving a lot of future work to do in the unsupervised learning category and so on. We give a summary in Table 6.4. The summary focuses on whether the information from multiple users is used during the training or testing phase. For example, Wang et al. only model concurrent activities during the training phase [42]. Zheng et al. model multiple users, but the activities do not need to happen concurrently [46]. In Miluzzo et al.'s work [27], the multiple user's data are only used in the inference (or, testing) phase. Lian et al.'s work [23] [46] supports multi-user modeling for training and testing phases. However, the model in [46] is more general as it does not require the activity data to be observed at the same time. Kasteren et al.'s work [17] is different from the other four. It follows the transfer learning stetting, where training data and testing data are of different distributions.

Table 6.4 Comparison of different multi-user activity recognition projects.

Research work	Training	Testing
Wang et al. [42]	Multi-user	Single-user
Miluzzo et al. [27]	Single-user	Multi-user
Kasteren et al.[17]	Multi-user	Single-user
Lian et al.[23]	Multi-user	Multi-user
Zheng et al.[46]	Multi-user	Single-user

Much future work can be done in multi-user activity recognition. One particularly interesting direction is social activity recognition, which aims to use the social media information as input and predict the users' physical activities. In social activity recognition, the scales on both users and data can be much larger than those using limited sensor or spatial trajectory data. For example, Sakaki et al. treat the Twitter users as social sensors, and use these sensors for event detection [36]. In their work, the words and links in the Twitter messages are used as sensor readings, and a spatial-temporal model is built on the Twitter message streams. They build an earthquake notification system which uses these Twitter messages as the input, and outputs the earthquake notification when there is one.

6.4 Summary

This chapter surveys the recent research in trajectory-based activity recognition. We start with introducing the learning-based localization methods, which aim to generate the spatial trajectories. Many machine learning techniques used in localization also reappear in the trajectory activity recognition. For example, HMM can

be either used to smooth the location estimations [19] or used in sequential classifications for transportation mode detection [35]. The main part of this chapter discusses trajectory-based activity recognition for a single user and for multiple users. We have also categorized the previous research work according to the learning paradigms: supervised learning, unsupervised learning and frequent pattern mining.

Table 6.5 gives a summary of the applications based on trajectory activity recognition. Some of the applications are direct activity recognition problems, while the others can be more high-level and are based on activity recognition such as GeoLife. The second category of applications are usually built on the first category.

Table 6.5 Applications of Trajectory-based Activity Recognition

Applications
transportation mode detection. [47, 35]
goal recognition. [45]
animal moving patterns. [26]
daily living patterns. [12]
earthquick detection. [36]
smart houseing. [17]

Collecting the experimental data in trajectory activity recognition is usually tedious and expensive. For example, the MIT RealityMining data set has 100 subjects to collect their daily behavior over a period of nine months, which costs a lot of money and time. Fortunately some of data sets have been made public, and Table 6.6 lists some of them related to this chapter.

Table 6.6 Open datasets

Dataset
Geolife [48]
RealityMining [11]
SmartHome [29]
HouseTransfer [17]
WiFiLocalization [43]

Trajectory-based activity recognition is a great application area for many machine learning algorithms. Table 6.7 lists the algorithms used in the previous research work covered by this chapter. Most of the algorithms listed in the table are for sequential data. Some non-sequential algorithms such as KNN and Decision trees can also be used to give predictions for each trajectory segment independently, and also shown to have reasonable performances.

One contribution of this article is the categorization of trajectory-based activity recognition. We believe that multiple-user activity recognition will be an important future direction in this area. Besides digital sensors, social media such as Twitter contains very rich information about the users' activities, and thus can be used for further activity recognition. Introducing the social media data can greatly relieve

Table 6.7 Machine learning and data mining algorithms

Algorithm	Used in	Classical reference
KNN	[1]	
Decision Tree	[35]	[33, 6]
HMM	[19, 47, 35]	[34]
CHMM	[42]	[5]
DBN	[45]	[28]
CRF	[24]	[20]
FCRF	[23]	[38]

the data sparsity in activity recognition, but it also brings the large-scale data computation problem. Some cloud computing solution could be an option. Besides, as the social media data are very noisy, more careful data cleaning and information retrieval are vital.

References

1. Bahl, P., Padmanabhan, V.N.: Radar: An in-building rf-based user location and tracking system. In: Proc. The Annual IEEE International Conference on Computer Communications (INFOCOM), pp. 775–784 (2000)
2. Barnard, K., Duygulu, P., Forsyth, D.A., de Freitas, N., Blei, D.M., Jordan, M.I.: Matching words and pictures. Journal of Machine Learning Research **3** (2003)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research **3**, 993–1022 (2003)
4. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge Univ Pr (2004)
5. Brand, M., Oliver, N., Pentland, A.: Coupled hidden markov models for complex action recognition. In: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), CVPR '97, p. 994. IEEE Computer Society, Washington, DC, USA (1997)
6. Breiman, L.: Classification and regression trees. Chapman & Hall/CRC (1984)
7. Cao, H., Mamoulis, N., Cheung, D.: Mining frequent spatio-temporal sequential patterns. In: Proc. of IEEE International Conference on Data Mining (ICDM), pp. 8–pp. IEEE (2005)
8. Doucet, A., De Freitas, N., Gordon, N.: Sequential Monte Carlo methods in practice. Springer Verlag (2001)
9. Doucet, A., Godsill, S., Andrieu, C.: On sequential monte carlo sampling methods for bayesian filtering. Statistics and computing **10**(3), 197–208 (2000)
10. Eagle, N., Pentland, A.: Eigenbehaviors: Identifying structure in routine. Behavioral Ecology and Sociobiology **63**(7), 1057–1066 (2009)
11. Eagle, N., (Sandy) Pentland, A.: Reality mining: sensing complex social systems. Personal Ubiquitous Comput. **10**, 255–268 (2006)
12. Farrahi, K., Gatica-Perez, D.: Discovering routines from large-scale human locations using probabilistic topic models. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(1), 3 (2011)
13. Hofmann, T., Puzicha, J.: Latent class models for collaborative filtering. In: Proc. of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99), pp. 688–693 (1999)
14. Hofmann-Wellenhof, B., Lichtenegger, H., Collins, J.: Global positioning System. Theory and Practice. (1993)
15. Huynh, T., Fritz, M., Schiele, B.: Discovery of activity patterns using topic models. In: Proc. of International Conference on Ubiquitous Computing (UbiComp), pp. 10–19 (2008)

16. Huynh, T., Schiele, B.: Analyzing features for activity recognition. p. 159C163. Smart objects and ambient in- telligence: innovative context-aware services (2005)
17. van Kasteren, T., Englebienne, G., Kröse, B.J.A.: Transferring knowledge of activity recognition across sensor networks. In: Proc. of the International Conference of Pervasive Computing, pp. 283–300 (2010)
18. Krumm, J. (ed.): Ubiquitous Computing Fundamentals. Chapman and Hall/CRC, Boca Raton, FL (2010)
19. Ladd, A.M., Bekris, K.E., Rudys, A., Kavraki, L.E., Wallach, D.S., Marceau, G.: Robotics-based location sensing using wireless ethernet. In: Proc. of The Annual International Conference on Mobile Computing and Networking (MobiCom), pp. 227–238 (2002)
20. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. The International Conference on Machine Learning (ICML), pp. 282–289 (2001)
21. Li, Z., Ding, B., Han, J., Kays, R., Nye, P.: Mining periodic behaviors for moving objects. In: Proc. of the ACM Conference on Knowledge Discovery and Data Mining (KDD), pp. 1099–1108 (2010)
22. Li, Z., Han, J., Ji, M., Tang, L.A., Yu, Y., Ding, B., Lee, J.G., Kays, R.: Movemine: Mining moving object data for discovery of animal movement patterns. ACM Transactions on Intelligent Systems and Technology (ACM TIST) (Special Issue on Computational Sustainability) (Aug. 2010)
23. Lian, C.C., Hsu, J.Y.j.: Probabilistic models for concurrent chatting activity recognition. In: Proc. of the 21st International Jont Conference on Artifical Jntelligence (IJCAI '09) (2009)
24. Liao, L., Fox, D., Kautz, H.A.: Extracting places and activities from gps traces using hierarchical conditional random fields. I. J. Robotic Res. **26**(1), 119–134 (2007)
25. Liao, L., Patterson, D.J., Fox, D., Kautz, H.A.: Learning and inferring transportation routines. Artif. Intell. **171**(5-6), 311–331 (2007)
26. Liu, S., Liu, Y., Ni, L.M., 0002, J.F., Li, M.: Towards mobility-based clustering. In: Proc. of the ACM Conference on Knowledge Discovery and Data Mining (KDD), pp. 919–928 (2010)
27. Miluzzo, E., Cornelius, C., Ramaswamy, A., Choudhury, T., Liu, Z., Campbell, A.T.: Darwin phones: the evolution of sensing and inference on mobile phones. In: Proc. The Annual International Conference on Mobile Systems (MobiSys), pp. 5–20 (2010)
28. Murphy, K.: Dynamic bayesian networks: representation, inference and learning. Ph.D. thesis, UC Berkeley, Computer Science Division (2002)
29. Nazerfard, E., Rashidi, P., Cook, D.J.: Using association rule mining to discover temporal relations of daily activities
30. Ni, L.M., Liu, Y., Lau, Y.C., Patil, A.P.: Landmarc: Indoor location sensing using active rfid. Wireless Networks **10**(6), 701–710 (2004)
31. Pan, J.J., Yang, Q., Pan, S.J.: Online co-localization in indoor wireless networks by dimension reduction. In: Proc. of National Conference on Artificial Intelligence (AAAI), pp. 1102–1107 (2007)
32. Patterson, D.J., Liao, L., Fox, D., Kautz, H.A.: Inferring high-level behavior from low-level sensors. In: Proc. of International Conference on Ubiquitous Computing (UbiComp), pp. 73–89 (2003)
33. Quinlan, J.: C4. 5: programs for machine learning. Morgan Kaufmann (1993)
34. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE **77**(2), 257–286 (1989)
35. Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M.H., Srivastava, M.B.: Using mobile phones to determine transportation modes. ACM Transactions on Sensor Networks (TOSN) **6**(2) (2010)
36. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: Proc. of International World Wide Web Conference, pp. 851–860 (2010)
37. Schulz, D., Fox, D., Hightower, J.: People tracking with anonymous and id-sensors using rao-blackwellised particle filters. In: Proc. of International Joint Conferences on Artificial Intelligence (IJCAI), pp. 921–928 (2003)

38. Sutton, C.A., Rohanimanesh, K., McCallum, A.: Dynamic conditional random fields: factorized probabilistic models for labeling and segmenting sequence data. In: Proc. The International Conference on Machine Learning (ICML) (2004)
39. Thiagarajan, A., Ravindranath, L., LaCurts, K., Madden, S., Balakrishnan, H., Toledo, S., Eriksson, J.: Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones
40. Turk, M., Pentland, A.: Eigenfaces for recognition. Journal of cognitive neuroscience **3**(1), 71–86 (1991)
41. Vail, D.L., Veloso, M.M., Lafferty, J.D.: Conditional random fields for activity recognition. In: Proc. the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), p. 235 (2007)
42. Wang, L., Gu, T., Tao, X., Lu, J.: Sensor-based human activity recognition in a multi-user scenario. In: Proc. of the International Joint Conference on Ambient Intelligence (AmI), pp. 78–87 (2009)
43. Yang, Q., Pan, S.J., Zheng, V.W.: Estimating location using wi-fi. IEEE Intelligent Systems **23**(1), 8–13 (2008)
44. Yin, J., Chai, X., Yang, Q.: High-level goal recognition in a wireless lan. In: Proc. of National Conference on Artificial Intelligence (AAAI), pp. 578–584 (2004)
45. Yin, J., Shen, D., Yang, Q., Li, Z.N.: Activity recognition through goal-based segmentation. In: Proc. of National Conference on Artificial Intelligence (AAAI), pp. 28–34 (2005)
46. Zheng, V.W., Yang, Q.: User-dependent aspect model for collaborative activity recognition. In: In Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11) (2011)
47. Zheng, Y., Li, Q., Chen, Y., Xie, X., Ma, W.Y.: Understanding mobility based on gps data. In: Proc. of International Conference on Ubiquitous Computing (UbiComp), pp. 312–321 (2008)
48. Zheng, Y., Xie, X.: Learning travel recommendations from user-generated gps traces. ACM Transactions on Intelligent Systems and Technology (TIST) **2**(1), 2 (2011)