3

Sensing

Sensors are devices that detect and respond to signals. A sensor converts a physical parameter such as temperature, blood pressure, or humidity into a signal that can be measured electrically and reported as a sensor event. A wide variety of sensors are available for activity learning and monitoring. In recent years, these sensors have also become low cost, wireless, and deployable in real-world, mobile settings. As a result, sensors are ubiquitous and are being integrated into more diverse computational settings including activity learning and monitoring. In order to design a sensor-based approach to activity learning, we need to consider both the types of sensors that will be used and how the data can be preprocessed for integration into data analysis and machine learning algorithms. In this chapter, we review sensors that are commonly used for activity learning and describe the type of data they generate. We also describe features that can be extracted from sensor data and illustrate how they are used on sample activity sensor data.

3.1 SENSORS USED FOR ACTIVITY LEARNING

Sensors differ in type, purpose, output signal, and technical infrastructure. Here, we provide an overview of sensors that are popular for activity learning. We separate the discussion into two categories of sensors: sensors on objects that are not attached to the individual performing an activity (i.e., sensors in the environment) and sensors on objects that are attached to the individual that is performing an activity (i.e., sensors attached to clothes or that are carried by the individual).

3.1.1 Sensors in the Environment

Some sensors that monitor activities are not affixed to the individuals performing the activity but are placed in the environment surrounding the individual. These sensors are valuable in passively providing readings without requiring individuals to comply with rules regarding wearing or carrying sensors in prescribed manners. Because they are not customized for each person, environment sensors can monitor activities for a group of individuals but may have difficulty separating movements or actions among individuals that are part of that group. Here we describe sensors that are commonly found installed in and around physical environments.

Passive Infrared (PIR) Sensor PIR sensors, or motion sensors, detect infrared radiation that is emitted by objects in their field of view through multiple slots, as shown in Figure 3.1. If the difference in the detected radiation between the multiple slots of a PIR sensor is greater than a predefined threshold (as would happen when a warm body moves into or out of the range of the sensor), it generates a message. These sensors are often used in security settings where a light may turn on or an alarm may sound when motion is detected. Humans and pets emit infrared radiation, so PIR sensors can be used to detect their movement within their coverage area. Because a passive infrared sensor (PIR) sensor is sensitive to heat-based movement, it operates in dim lighting conditions. On the other hand, it may not sense movement if an object is between the sensor and the moving person. A PIR sensor will sense movement from any object that generates heat, even if the origin is inorganic. As a result, the motion sensor may generate messages if a printer starts printing a large document in the vicinity or if a nearby baseboard heater suddenly turns on.

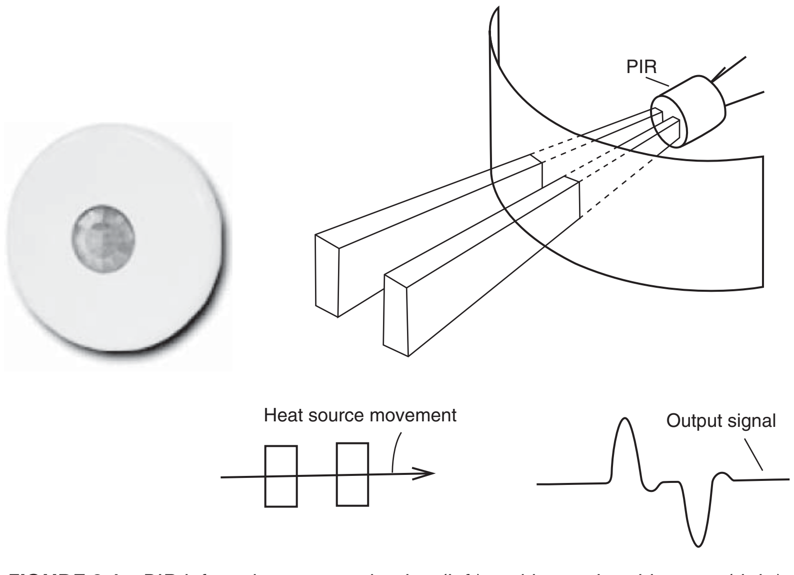


FIGURE 3.1 *PIR infrared sensor packaging (left) and internal architecture (right).*

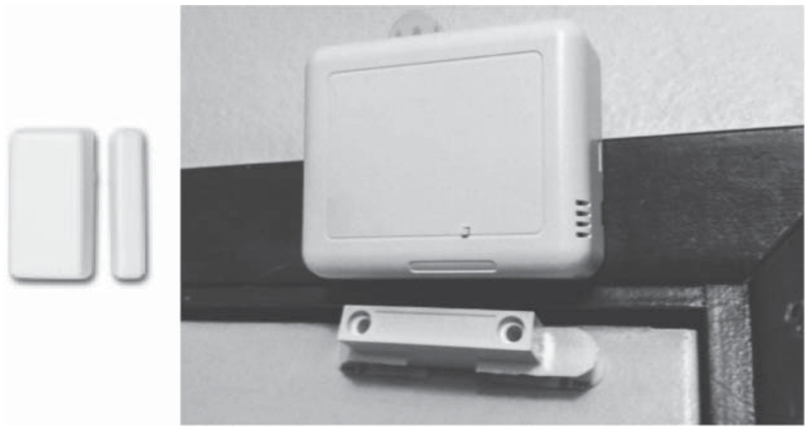


FIGURE 3.2 *The two components of a magnetic contact switch encased in plastic (left) and a magnetic contact switch acting as a door open/shut sensor (right).*

Magnetic Door Sensor A magnetic contact switch sensor consists of two components: a reed switch and a magnet, as shown in Figure 3.2. When the door shown in Figure 3.2 is closed, the magnet component pulls the metal switch in the second component closed so the electric circuit is complete, thus changing the state of the sensor. The sensor can report this change of state as a sensor event. When the magnet is moved by opening the door, the spring snaps the switch back into the open positive. This cuts off the current and closes the relay, again causing a change in the state of the sensor that can be reported as a sensor event. This feature is useful for detecting if doors, windows, drawers, or cabinets are open or shut, as shown in Figure 3.2.

Temperature Sensor, Light Sensor, Humidity Sensor Additional sensors can be placed in environments to measure ambient temperature, lighting, and humidity. These types of sensors are frequently bundled into one package. For example, the motion sensor shown on the left in Figure 3.1 also contains a light-level sensor in the same unit, while the magnetic door sensor shown in Figure 3.2 also contains a temperature sensor. Such sensors are calibrated to each periodically report their current status (e.g., light level, humidity reading, and temperature reading) or they report their current reading when there is a sufficiently large change in the value from the previous time point.

Vibration Sensors These types of sensors are often attached to items or placed on surfaces in order to detect interaction with the corresponding object. Some sensors are designed to be sensitive both to vibration (dynamic acceleration) and tilt (static acceleration). While they can be useful for generating events when the object they are attached to is handled, they may also generate events when they are accidentally bumped or when nearby surfaces shake.

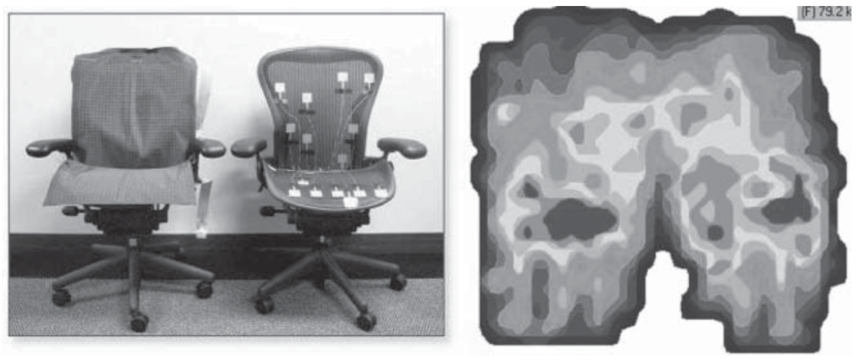


FIGURE 3.3 *Tactile pressure sensors positioned in a chair (left) and interpreted as a pressure map (right).*

Pressure Sensors There are many types of sensors that measure pressure. For monitoring of activities in a particular environment, tactile sensors are sensitive to touch, force, or pressure. These pressure sensors detect and measure interaction between an individual and a contact surface. For example, Figure 3.3 shows pressure sensors placed on a chair. The force distribution can be mapped as shown in the figure or the combined force can be compared with a threshold value to note that there is an object in contact with the sensor. Pressure sensors can be placed on or under chairs, door mats, floors, and beds to monitor the location and weight distribution of an individual in the space.

Global Positioning System (GPS) Sensors GPS sensors provide valuable location information and can be attached to objects in the environment or on a wearable or carryable device. They rely on communication with at least four global positioning system (GPS) satellites and thus are not usable in all settings. As an example, a smart phone can communicate with cell towers in the region to triangulate the position of the device based on time delay and signal strength in sending the message to and from the towers. Common reporting axes are latitude, longitude, and height. In addition to GPS, other signals including GSM, WiFi, and Bluetooth signals can be employed to provide improved localization, particularly indoors where GPS may not be consistently available.

Radio-Frequency Identification (RFID) Sensors Radio-frequency identification (RFID) employs radio-frequency electromagnetic fields to transfer data. RFID tags act as wireless barcodes that can be attached to objects. Figure 3.4 shows how RFID codes can be “hidden” under a visible object barcode. The RFID reader, or interrogator, queries a tag. A passive tag does not require a power source but collects energy from the interrogating electromagnetic (EM) field and then acts as a passive transponder to modulate the signal. The reader can interpret the signal to determine the tags that are in its area of coverage. As a result, RFID technology can be used to

FIGURE 3.4 *RFID tag positioned beneath visible object barcode.*

track the presence of objects or humans that are wearing a tag and are in close proximity to a reader. More complex tags are designed to be programmable and act as a power source for other sensors that are described in this chapter including light, temperature, and accelerometer sensors. RFID technology can be limited by the range of the reader, which is greater outdoors. For indoor environments, a reader typically needs to be placed in each room where tags will be monitored.

Power Meter Power meters provide readings indicating the amount of electricity that was consumed by a particular building for a unit of time. Electricity typically flows from a utility meter to the building's breaker panel. One method of monitoring usage is thus to clamp a measuring device around the main conductors inside a building's breaker panel. The meter calculates the amount of electricity currently being consumed and can report instantaneous values, accumulated values, or sufficiently changes in consumption.

3.1.2 Sensors on the Body

In addition to placing passive sensors around the environment in which an individual interacts, the individual can also wear, or carry, additional sensors that collect data about their gestures, movement, location, interactions, and actions. These sensors were originally sewn into garments, worn as watches or placed on the body. However, many of these sensors are now also found on smart phones that are routinely carried by individuals as they perform their daily activities. Here we describe some sensors that are commonly used as wearable or carryable mechanisms for monitoring activities.

Accelerometer An accelerometer is a common sensor that is worn or carried by an individual. Acceleration changes can indicate the beginning or ending of a motion (see Figure 3.5). As shown in Figure 3.6, this three-dimensional sensor measures acceleration along the x, y, and z axes. Acceleration is calculated as the change in velocity over time, or $a = \Delta V/\Delta t$. Acceleration can be detected when the person

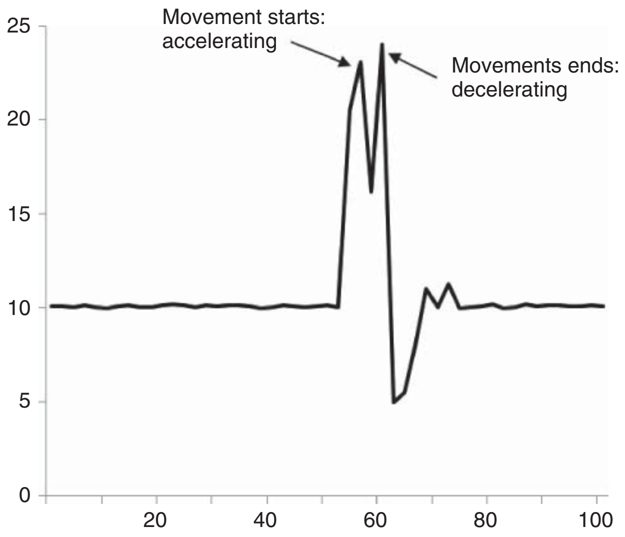


FIGURE 3.5 *Increased acceleration values can indicate a start or stop of a motion.*

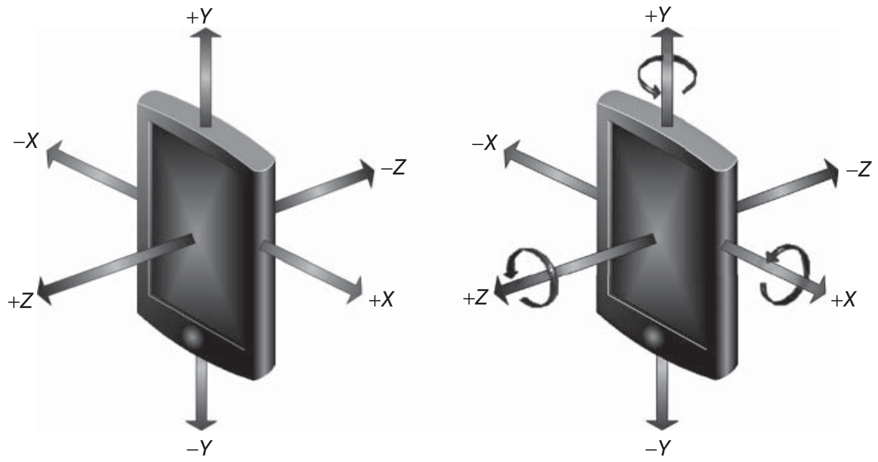


FIGURE 3.6 *A smart phone with a three-axis accelerometer (left) and a three-axis gyroscope (right).*

carrying the device initiates a change in direction or velocity, which makes these sensors ideal for detecting different types of movement.

Gyroscope A gyroscope sensor can enhance the movement detection provided by an accelerometer. In contrast to an accelerometer, the gyroscope measures the change in angular position over time, or angular velocity, calculated as $v = \Delta\theta/\Delta t$. Gyroscopes sometimes experience accumulated error due to drift but are frequently used in combination with other sensors to provide a more complete model of motion. For example, gyroscope readings are often packaged with accelerometer readings to provide six-dimensional motion data vectors.

Magnetometer This sensor measures the strength of the magnetic field in three dimensions. While it bears some similarity to a compass, it does not always behave the same as a compass because the magnetometer may not always point north when it is influenced by magnetic interference. A magnetometer is valuable for providing some sensor orientation and for detecting and locating metallic objects within its sensing radius. Human activities can then be modeled based on their proximity to these detected objects.

3.2 SAMPLE SENSOR DATASETS

Appendix 1 contains sample sensor data that are captured while humans perform routine activities. There are two tables in the Appendix, one for each of two activities that are performed: Hand Washing and Sweeping. Each table contains sensor messages that are generated while one individual performs the activity with no interruptions. The individual performed the activities in a smart home filled with environmental sensors. These sensors include infrared motion detectors, magnetic door sensors, temperature sensors, water flow and gas flow sensors, a whole-home power meter, and accelerometer-based vibration sensors placed on selected objects in the space. The layout of the instrumented main floor of the smart home and the position of the sensors is shown in Figure 3.7. In addition, the individual wore two accelerometer/gyroscope sensors: one on the arm and one on the hip, as shown in Figure 3.8. Each wearable sensor reports six values: acceleration in the x, y, z directions and rotational velocity around the x, y, and z axes.

In the Hand Washing activity, the participant washes her hands at the kitchen sink using hand soap that is located in a dispenser next to the sink. After her hands are washed, she uses a cloth towel also located in the kitchen to dry her hands. Appendix 1 shows the sensor data that were gathered during the performance of this activity for one participant and in Figures 3.9 and 3.10 the timing of sensor events corresponding to the sensor data are plotted.

A second example activity is plotted in Figures 3.11 and 3.12 for one participant and the corresponding sensor data are provided in Appendix 1. This is a Sweeping activity in which the participant is asked to sweep the kitchen floor and to dust the dining room and living room. All of the needed supplies, including the broom, duster, and dustpan are located in the kitchen closet at the bottom right of the space shown in Figure 3.6.

3.3 FEATURES

In Chapter 4, we will introduce a variety of machine learning algorithms that can be utilized to model human activities. Typically, these algorithms do not process

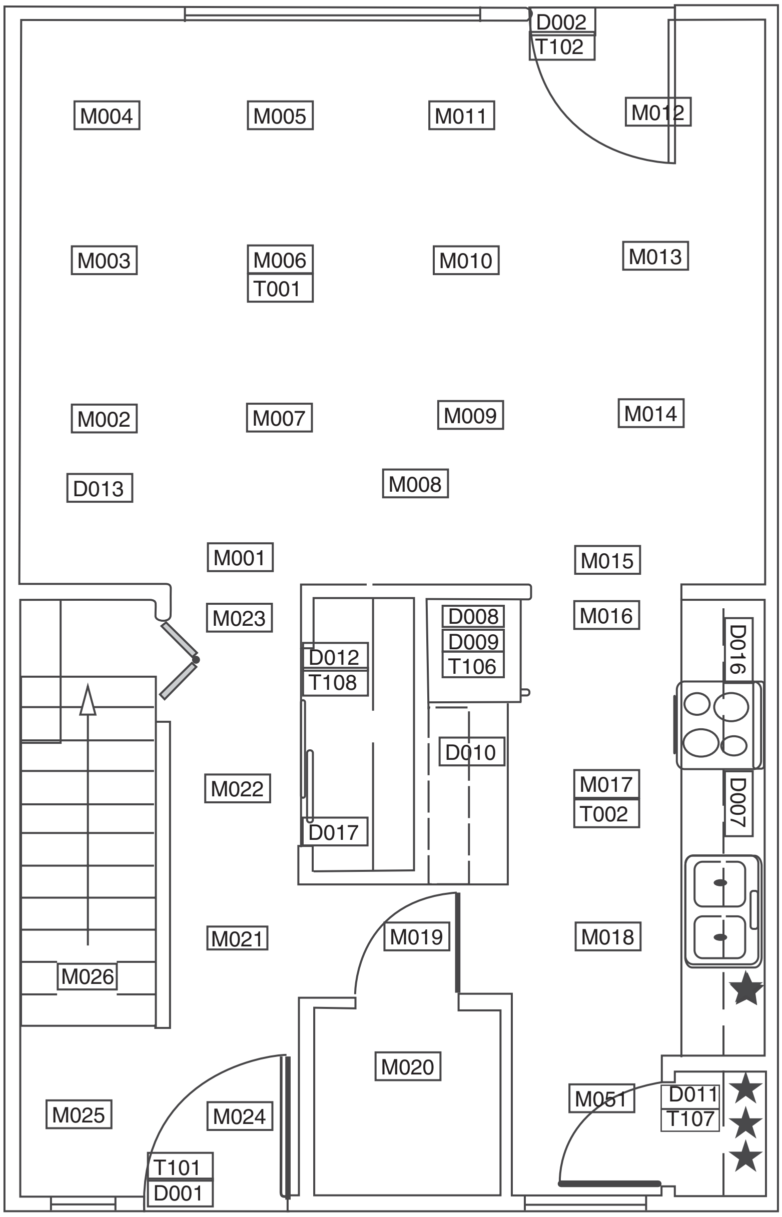


FIGURE 3.7 *Floor plan and sensor layout for the first floor of a smart home. Sensor identifiers starting with "M" are infrared motion sensors, sensors starting with "D" are magnetic door closure sensors, and sensors starting with "T" are ambient temperature sensors. The hand soap object sensor is located near the sink and the additional object sensors are in the kitchen closet. All object sensors are indicated with stars. Additional sensors included in these datasets are* `A001` *(gas usage sensor attached to burner),* `A002` *and* `A003` *(hot and cold water consumption sensors, respectively), and* `P001` *(whole-home power usage meter).*

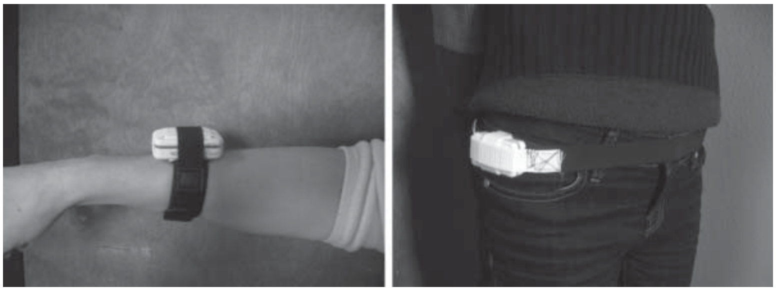


FIGURE 3.8 *Positioning of wearable accelerometers on an individual's dominant arm (left) and hip (right).*

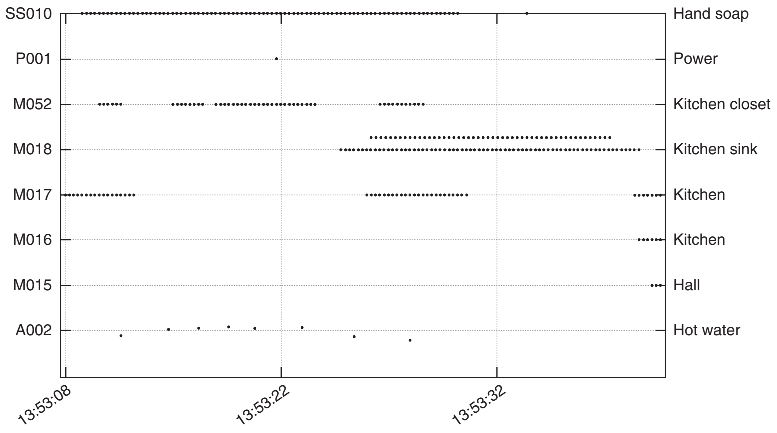


FIGURE 3.9 *Time plot of sensor activity during the Hand Washing activity. The x axis shows the time of day when the sensor message was generated. The y axis shows the identifier (on the left) and functional location (on the right) for each sensor generating messages.*

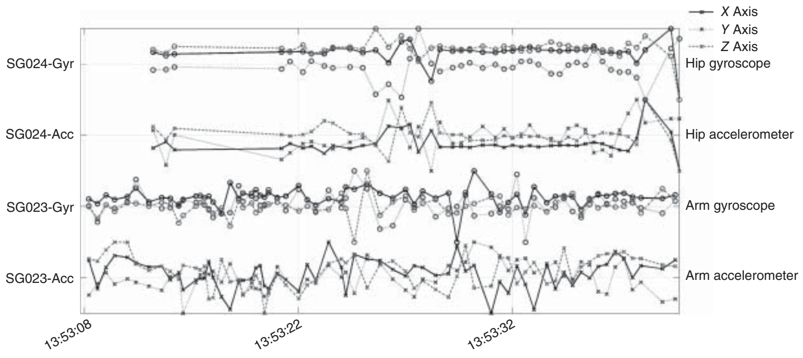


FIGURE 3.10 *Time plot of normalized accelerometer readings during the Hand Washing activity. The x axis shows the time of day when the sensor message was generated. The y axis shows the identifier (on the left) and placement (on the right) for each sensor. There are three lines for each sensor, corresponding to the x, y, and z axes.*

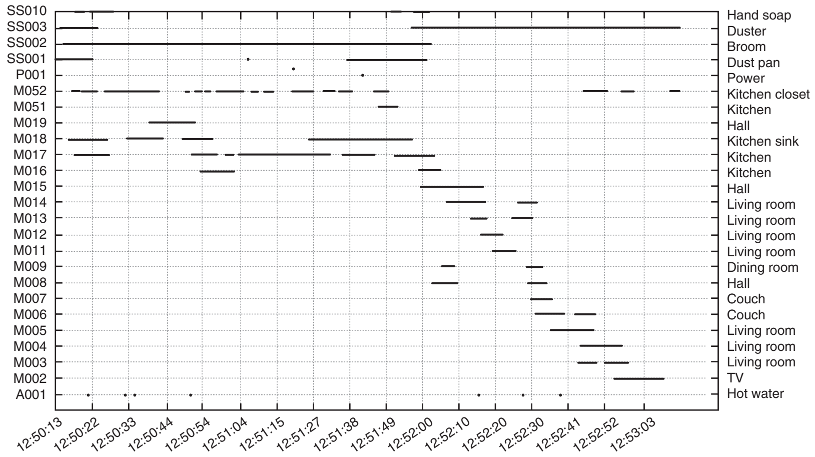


FIGURE 3.11 *Time plot of sensor activity during the Sweeping activity. The x axis shows the time of day when the sensor message was generated. The y axis shows the identifier (on the left) and functional location (on the right) for each sensor generating messages.*

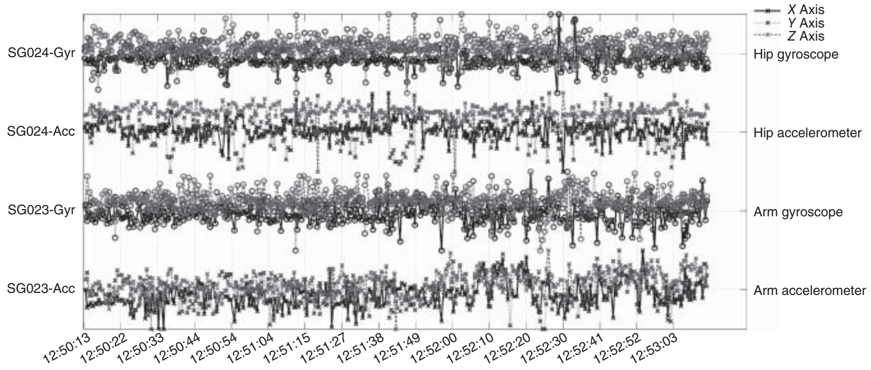


FIGURE 3.12 *Time plot of normalized accelerometer readings during the Sweeping activity. The x axis shows the time of day when the sensor message was generated. The y axis shows the identifier (on the left) and placement (on the right) for each sensor. There are three lines for each sensor, corresponding to the x, y, and z axes.*

raw sensor event data generated by sensors we reviewed in this chapter. Instead, the learned model is often a function of higher-level features that are used to describe the sensor data. As a result, the quality of the learned model varies with the expressiveness of the input feature vector. Because so many varied sensor platforms have been used to model activities, there is a wealth of possible features available to represent an activity. In this discussion, we categorize sensor data features into four groups: (i) Features describing characteristics of the sensor event sequence, (ii) Features describing characteristics of discrete sensor values, (iii) Features describing characteristics of continuous sensor values, and (iv) Activity context.

We will use a subsequence of sensor events from the Sweeping activity example provided in Appendix 1 to illustrate the methods of calculating feature values from sensor data. The corresponding sample of sensor events is shown in Table 3.1.

3.3.1 Sequence Features

We assume in this book that activities are modeled based on a sequence of sensor events. The sequence may represent an entire activity occurrence if the data have been pre-segmented into individual, nonoverlapping activity streams. On the other hand, the sequence may also represent a captured window of time if activities are learned or recognized in real time as sensor events arrive. In either situation, there exist characteristics of the sequence itself that can be useful for modeling activities. We describe such sequence features here.

Time Humans and animals are creatures of habit. As a result, many of our daily activities revolve around specific times. For example, we typically go to sleep in the evening, wake up in the morning, and eat at equally spaced intervals throughout our waking hours. Because individuals follow fairly similar schedules, the notion of time allows us to coordinate group activities such as meetings and sporting events around time units such as the time of the day, the day of the week, the day of the month, and the date. For this reason, the time of the event sequence represents an important feature. A challenge is deciding the precision of time that should be included in the feature vector. The sequence could be represented by the date, the day of the month, the day of the week, and the wall clock time with any level of granularity. For our Sweeping example, we capture time in terms of three granularities: the hour of the day, the number of minutes past midnight on the current day, and the number of milliseconds past midnight on the current day. For sliding-window methods, the focus is on the last sensor event in the sequence and the events before this provide context for the last sensor event. As a result, we capture these time features for the last event in the sequence.

Example 3.1 The time features for our sample activities are:

Hour: 12
Minute: $(12 * 60) + 50 = 770$
Milliseconds: $((((12 * 60) + 50) * 60) + 22) * 1000) + 8739 = 46,230,739$

Sequence Size The size of the sequence can provide insights on the type of activity that is being performed and how it is being performed. For pre-segmented data, the sequence size not only provides an indication of the activity that is being performed but also how much time is being taken to complete the activity.

Example 3.2 As an example, we see that the size of the Sweeping activity is much longer in both duration and number of sensor events than the Hand Washing activity.

TABLE 3.1 Sample of Window Events from the Sweeping Activity

Time	ID	Message
12:50:13.102682	Dustpan	MOVED
12:50:13.149904	HIP	(1019,1921,1520,1850,1800,1898)
12:50:13.367222	ARM	(1584,2402,2318,2040,1838,1736)
12:50:14.128257	HIP	(973,1951,1545,1839,1918,1900)
12:50:14.217971	Duster	MOVED
12:50:14.357487	ARM	(948,1851,1837,1886,1820,2028)
12:50:15.129119	HIP	(1055,1745,1792,1840,1814,1872)
12:50:15.873883	ARM	(926,1867,2119,1751,1853,1863)
12:50:16.134036	HIP	(1047,2137,1487,1819,1594,1992)
12:50:16.235281	Broom	MOVED
12:50:16.36758	ARM	(1055,1677,2182,1705,1613,1862)
12:50:17.001771	M018	ON
12:50:17.11736	HIP	(997,1970,1625,1752,1438,1907)
12:50:17.345406	ARM	(1189,1935,2232,1840,1682,1813)
12:50:18.119112	HIP	(1072,2052,1559,1880,1796,1949)
12:50:18.341054	ARM	(1071,1889,1978,1780,1721,1982)
12:50:18.452039	M017	ON
12:50:18.790212	HandSoap	MOVED
12:50:19.103787	HIP	(1159,2026,1598,1863,1744,1951)
12:50:19.349545	ARM	(1043,1330,1688,1676,1825,1872)
12:50:20.101233	HIP	(986,1966,1573,1838,1727,1921)
12:50:20.334268	ARM	(1007,1674,1700,1702,1868,1916)
12:50:20.741536	HandSoap	STILL
12:50:21.097739	HIP	(981,2074,1527,1853,1724,1930)
12:50:21.342635	ARM	(1208,2346,1888,2217,1234,1483)
12:50:21.895284	Burner	2.8227
12:50:22.090522	HIP	(1149,2003,1701,1857,1615,1949)
12:50:22.339887	ARM	(1134,2594,2088,1894,1845,1872)
12:50:22.412985	HandSoap	MOVED
12:50:22.8739	Dustpan	STILL

If the length of the sliding window is fixed, then measuring the sequence size in terms of duration provides an indication of how much movement and interaction is occurring (or not occurring) during the window. If the time duration of the window is fixed, then the corresponding sequence length serves as this indicator. For our example, we are using a sliding window of length = 30 events, so we add a sequence size feature that represents the time duration, or the time stamp of the last event in the sequence minus the time stamp of the first event in the sequence. As with the *time* feature, the precision of the time duration needs to be determined. We add the sequence duration feature to our vector in terms of the milliseconds that elapsed during the sequence window. For our example, duration is calculated using the smallest measured time unit, milliseconds. The time-based features are summarized in Table 3.2.

$$\begin{aligned}\text{Duration: } 12:50:22.8739 - 12:50:13.102682 &= 46{,}230{,}739 \\ &- 46{,}214{,}027 = 16{,}712\end{aligned}$$

TABLE 3.2 Window-Based Features for the Sweeping Activity

Window-Based Features			
Hour	12	Milliseconds	46,230,739
Minute	770	Duration	16,712

3.3.2 Discrete Event Features

In some settings, sensors report activity behavior as a well-defined event using a string-based message rather than a numeric value. Many of these sensors also report messages only when events are detected, rather than sampling the state of the sensor at near-constant time intervals. As an example, motion sensors may report a message of "`ON`" when movement is detected in their field of view, and report "`OFF`" when movement ceases to be detected. These discrete-value sensor messages can be converted to numeric-value features by mapping string messages onto a set of binary-valued numeric values. Event-based sensors could also emulate sampling-based sensors by reporting their state at constant time intervals. In this case, the motion sensor would report "`ON`" at every sampling time interval if movement has been detected within the time interval, otherwise it would report "`OFF`" for the current time interval. As discrete-value, event-based sensors offer convenience in terms of simple representations with minimal reported and stored messages, sensor features can be incorporated that are calculated based on these types of sensors. Here we describe common features used for this purpose.

Bag of Sensors The bag-of-sensors feature is analogous to the bag-of-words model commonly used in text or image mining. In text mining, a document can be represented as a set of words that appear in the document together with the associated word frequencies in the document. The representation is independent of the sequence of words in the document, thus the words and frequencies are thrown together into a "bag." We can use a bag-of-words approach for discrete, event-based sensors by assembling the list of sensors that fall into this category together with the number of times they generate messages in the current window. The number of messages they generate will vary according to the sensitivity of the sensor. For example, the motion sensors used in our Sweeping example are designed to generate an "`ON`" message when movement is detected corresponding to an individual weighing at least 40 pounds. The sensor generates an "`OFF`" message when no movement from the individual has been detected for 1.25 seconds. If the sensitivity remains consistent across all possible activities, the bag-of-sensors model can be effective at modeling activities and is appealing in its simplicity.

Note that a visualization of the bag of sensors feature produces a heat map showing how a particular smart environment was utilized over a period of time, as shown in Figure 3.13 (left). For this type of heat map, the pixel intensities or choice of colors indicates the amount of messages or the recency of messages that were generated by the corresponding sensors. The bag of sensors feature is analogous to the motion

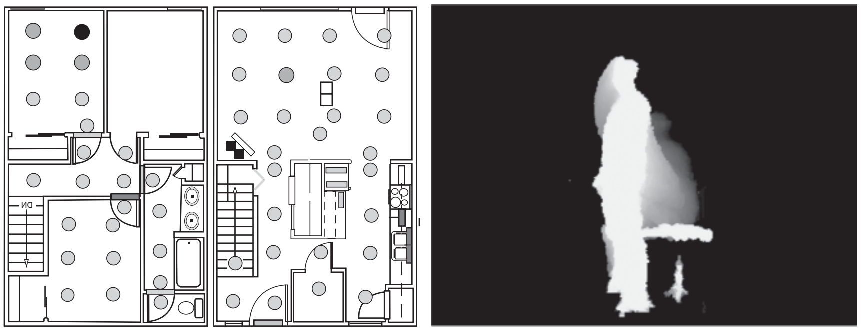


FIGURE 3.13 *Heat map of discrete event sensor usage in a smart home (left, motion sensor locations represented by circles in the image) and motion history image of a sit-to-stand transition movement (right). In both images, the pixel intensities indicate activities occurring in the corresponding region of the space. In the left picture, darker circles indicate areas in which more time was spent. In the right picture, lighter regions indicate more recent movement occurring in the region.*

history images used in vision-based activity recognition. As shown in Figure 3.13 (right), the pixel intensity can be a function of the recency of motion in a sequence of the amount of motion detected in the corresponding region. In both cases, the feature provides a static snapshot of recent activity that can be used to compare with templates for known activities.

Example 3.3 For our Sweeping activity example, we add sensor counts for the event-based sensors, which include vibration sensors (for the dustpan, duster, broom, and hand soap objects), infrared motion sensors (`M017` and `M018`), and the burner sensor. Although the burner sensor reports a numeric value, it is calibrated to generate a message only when the change in burner level exceeds a threshold amount and therefore can be appropriately included in the sensor counts. We do not count sensor events that contain messages corresponding to a lack of activity, such as motion sensor "`OFF`" and object "`STILL`" messages. The additions to our feature vector are listed in Table 3.3. As can be seen from this example, there are many features generated by a bag-of-sensors model and the resulting vector will often be sparse. This situation can be addressed through feature selection or dimensionality reduction techniques, described in Chapter 4.

Elapsed Sensor Time Another feature that can be valuable for discrete, event-based sensors is to calculate the time that has elapsed for each such sensor since it last generated a message. The delay can be calculated based just on events found within the current window or can extend to earlier windows as well. This type of feature provides context for the current activity. In cases where the current window contains more than one activity, it can also provide information on how relevant messages early in the window are to the activity occurring at the end of the window. For example, if an individual took a nap in a chair and a few hours later got

TABLE 3.3 Discrete-Based Features for the Sweeping Activity

Discrete-Based Features			
Sensor Counts		Sensor Elapsed Times	
Dustpan	1	Dustpan	97,712
Duster	1	Duster	86,559
Broom	1	Broom	66,386
Hand soap	2	Hand soap	40,837
`M017` (Kitchen)	2	M017 (Kitchen)	44,219
`M018` (Sink)	1	M018 (Sink)	58,721
Burner	1	Burner	9,786
All other sensors	0	All other sensors	?

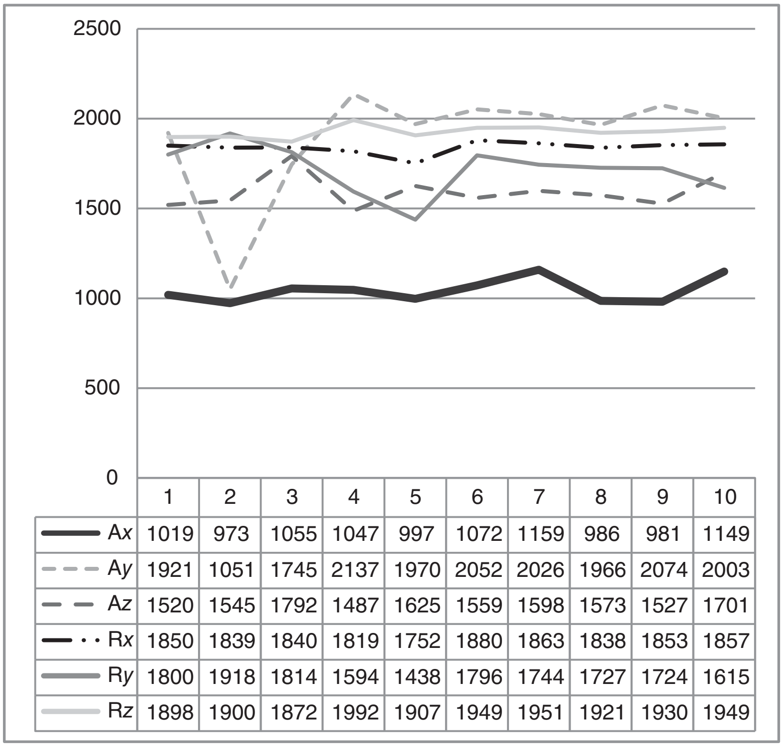
up and left the home, a sequence window could contain sensor messages from both activities. However, the time that elapsed for sensors near the chair will be much greater than those near the front door. The elapsed sensor time feature will capture this relevant information that can be used to more accurately model the Nap and Leave Home activities.

Example 3.4 In our Sweeping example, we add elapsed sensor times for all of the discrete event-based sensors. For those sensors that do not occur within the current window, we can report the values as missing, we can calculate the time since they fired before the current window, or we can use a time value that is greater than the current window duration. Here we report the elapsed sensor times, in milliseconds, only for those event-based sensors that appear in the current window. The elapsed sensor times for all other sensors are reported as missing, denoted by "?". As before, we consider events that indicate activity and not a lack of activity.

3.3.3 Statistical Features

While some features generate messages only when a particular condition is detected, many others report numeric state values at near-constant time intervals. Such data constitutes a sequence of sensor values measured at uniform time intervals, or a time series. As a result, many of the methods used to extract features from time series data can be used in this context. Examples of this type of sensor that are used to monitor activities include accelerometers, gyroscopes, light sensors, temperature sensors, and power usage monitors.

Example 3.5 In our Sweeping activity data, the "`ARM`" and "`HIP`" sensors are accelerometers that reported acceleration and rotational velocity in the x, y, and z axes at uniform time intervals. We use the "`HIP`" accelerometer data to illustrate the feature calculations and add the results to our activity feature vector. Plots of the values for these axes over our sample window are shown in Figure 3.14. Note that sensor data from discrete event sensors can be converted to sampling-based messages by forcing them to periodically report their current state. The state may correspond to



	1	2	3	4	5	6	7	8	9	10
Ax	1019	973	1055	1047	997	1072	1159	986	981	1149
Ay	1921	1051	1745	2137	1970	2052	2026	1966	2074	2003
Az	1520	1545	1792	1487	1625	1559	1598	1573	1527	1701
Rx	1850	1839	1840	1819	1752	1880	1863	1838	1853	1857
Ry	1800	1918	1814	1594	1438	1796	1744	1727	1724	1615
Rz	1898	1900	1872	1992	1907	1949	1951	1921	1930	1949

FIGURE 3.14 *Plot of acceleration (Ax, Ay, Az) and rotational velocity (Rx, Ry, Rz) values for HIP accelerometer from the Sweeping activity data sample.*

the most recent message and persists until an event occurs that generates a different message.

- *Max*, *Min*. These values, as the others listed below, are calculated from the sensor values over the specific window of time. The values are calculated separately for each feature dimension. We let S refer to the set of values for a particular sensor feature and $s_i \in S$ refers to one particular value in the set. In our example, there are six such features being considered. *Max*(S) thus refers to the maximum of the numbers $s_i \in S$ and Min(S) refers to the minimum value in the set of numbers. These features can help distinguish between activities that are characterized by different ranges of movements.
- *Sum*, *Mean*. As before, these are computed based on the set of values for a particular sensor feature found in the current window. In these definitions, we let N refer to the number of values found in the set S for the current window.

$$\text{Sum}(S) = \sum_{i=1}^{N} s_i \tag{3.1}$$

$$\text{Mean}(S) = \bar{s} = \frac{\text{Sum}(S)}{N} \tag{3.2}$$

- *Mean Absolute Deviation*, *Median Absolute Deviation*, *Standard Deviation*. While mean absolute deviation (and the related measure of standard deviation) provide a measure of the variability of the sample, in some cases the median absolute deviation provides a more robust measure. This is particularly true in situations where outliers can bias the mean value.

$$\text{MeanAbsDev }(S) = \frac{1}{N}\sum_{i=1}^{N} |s_i - \bar{s}| \tag{3.3}$$

$$\text{MedAbsDev}(S) = \frac{1}{N}\sum_{i=1}^{N} |s_i - \text{Median}(S)| \tag{3.4}$$

$$\text{StDev}(S) = \sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (s_i - \mu)^2} \tag{3.5}$$

- *Coefficient of Variation*. Another indication of variability in the values is the coefficient of variation. This measure shows the relationship of the standard deviation to the mean of the set of values. This value is typically only computed for features that do not have negative values, as it may not have useful meaning for features with negative values.

$$\text{CV}(S) = \frac{\sigma}{\mu} \tag{3.6}$$

- *Zero Crossings*. Zero crossings are typically calculated only for time series data with a mean of 0. However, zero crossings can be computed for an arbitrary sequence of values to represent the number of times the signal crosses its median. This feature is useful for distinguishing between activities involving rapid and slow movements (e.g., distinguishing Walking from Running).

Example 3.6 In the Ax values for the Hip accelerometer shown in Figure 3.14, this happens between points 2 and 3 ($973.00 < 1034.50 < 1055.00$), points 5 and 6 ($1047.00 > 1034.50 > 997.00$), points 6 and 7 ($997.00 < 1034.50 < 1072.00$), points 8 and 9 ($1159.00 > 1034.50 > 986.00$), and between points 10 and 11 ($981.00 < 1034.50 < 1149.00$), so the number of zero crossings is 5.

$$\text{ZC}(S) = |s_i < \text{Median}(S) < s_{i+1}| + |s_i > \text{Median}(S) > s_{i+1}| \tag{3.7}$$

- *Percentiles*, *Interquartile Range*. If the set of values for a particular feature is sorted in nondecreasing order, then the values that appear at various points along the sorted list provide insights in how the values are distributed over the sequence. A percentile reports a value below which a specified percentage of the feature values fall. Percentiles provide insights on how the values are distributed across the sequence.

Example 3.7 For our sample dataset, we report the 20th, 50th, and 80th percentiles:

$$Pt(S, 20) = 981.01,\ Pt(S, 50) = 1019.01, \mathrm{Pt}(S, 80) = 1072.01$$

Based on these percentiles, the interquartile range can be calculated, which is the difference between the 75th and 25th percentiles. For our Sweeping activity example, the interquartile range is

$$\mathrm{IQ}(S) = 1072.01 - 981.01 = 91.00$$

- *Square Sum of Percentile Observations*. Once the percentiles are defined, the square sum of observations that fall below each percentile (or alternatively, above the percentile) can be reported as a separate feature.

Example 3.8 The corresponding square sum of observations for our example is:

$$\mathrm{SqSumPt}(S, 20) = 1,909,090, \quad \mathrm{SqSumPt}(S, 50) = 4,913,656,$$
$$\mathrm{SqSumPt}(S, 80) = 8,272,047$$

- *Binned Distribution*. This measure represents the fraction of values that fall within equally-spaced bins that span the entire range of sensor values. Binning also provides a mechanism for mapping continuous-valued attributes onto discrete values, by mapping the original value onto a range number or bin number into which the value falls.

Example 3.9 For our example, we consider three equally spaced bins. For k bins, the upper bin margins are defined as shown in Equation 3.8.

$$\bigcup_{j=1}^{k} \frac{j \times (\mathrm{Max}(S) - \mathrm{Min}(S))}{k} \tag{3.8}$$

In our scenario, then, the three bins represent the values ranges [973.00 .. 1035.00], (1035.00 .. 1097.00], and (1097.00 .. 1159.00]. The fraction of values that falls into these bins, or the binned distribution, is 0.5, 0.3, and 0.2.

- *Skewness*. This feature provides an indicator of the degree of asymmetry in the distribution of values for the feature. The greater the skewness, the greater is the lack of symmetry below and above the sample mean. A symmetric dataset will have skewness near 0.

Example 3.10 Applying Equation 3.9 to our sample, we see that the skewness of the data sample is 0.83. This indicates that the data are fairly asymmetric. Looking at the values plotted in Figure 3.14, we observe that in fact about 2/3 of the data points fall below the mean value for , so data are skewed toward lower values.

$$\text{Skewness}(S) = \frac{\frac{1}{N}\sum_{i=1}^{N}(s_i - \mu)^3}{\left(\frac{1}{N}\sum_{i=1}^{N}(s_i - \mu)^2\right)^{\frac{3}{2}}} \tag{3.9}$$

- *Kurtosis*. Similar to skewness, kurtosis provides an indication of the shape of the feature value distribution. While skewness considers the symmetry of the distribution, kurtosis considers the amount of peakedness (or conversely, the amount of flatness) of the distribution toward the mean. Equation 3.10 provides the formula for calculating kurtosis. If the kurtosis is high then the distribution has a distinct peak near the mean and decline quickly with heavy tails. A normal distribution, for example, has a kurtosis of 3, while a uniform distribution has a kurtosis close to 0. In contrast, distributions with low kurtosis have a flat top near the mean.

Example 3.11 We observe as shown in Figure 3.14 that the data are not concentrated around the mean, but have a rather flat distribution. Applying Equation 3.10 to our data, we get a kurtosis of −0.48, which is consistent with this observation.

$$\text{Kurstosis}(S) = \frac{\frac{1}{N}\sum_{i=1}^{N}(s_i - \mu)^4}{\left(\frac{1}{N}\sum_{i=1}^{N}(s_i - \mu)^2\right)^{3}} - 3 \tag{3.10}$$

- *Correlation*. The amount of correlation that exists between multiple sensors or between the dimensions of a multidimensional sensor, such as a multiple-axis accelerometer, can provide important insights on the type of activity that is being monitored. For example, an accelerometer on a hand may see similar acceleration along all three dimensions if the individual is swimming, but may see little or no acceleration for one axis while the others axes do indicate acceleration for actions such as wiping a table surface. Correlation is measured here between two dimensions at a time, S and V, and can be calculated between all pairs of dimensions.

Example 3.12 Using Equation 3.11, we calculate the correlation between acceleration in the x direction and acceleration in the y direction for our sample data as 0.37. This value indicates that there is a slight, but not strong, positive correlation between the two dimensions and therefore when one changes value the other will tend to change value in the same direction.

$$\text{Corr}(S, V) = \frac{\sum_{i=1}^{N}(s_i - \overline{s})(v_i - \overline{v})}{\sqrt{\sum_{i=1}^{N}(s_i - \overline{s})^2 \sum_{i=1}^{N}(v_i - \overline{v})^2}} \tag{3.11}$$

TABLE 3.4 Continuous-Based Features for Sweeping Activity Hip Accelerometer Data

Hip A*x* Continuous-Based Feature Values for Sweeping Activity					
Max	1,159.00	CV	0.09	SqSumPt(80)	8,272,047
Min	973.00	ZC	5	BD(1)	0.5
Sum	10,438.00	PT(20)	981.01	BD(2)	0.3
Mean	1,043.00	PT(50)	1019.01	BD(3)	0.2
Median	1,033.00	PT(80)	1072.01	Skewness	0.83
MeanAbsDev	19.12	IQ	91.00	Kurtosis	−0.48
MedAbsDev	25.50	SqSumPt(20)	1,909,090	Corr(A*x*,A*y*)	0.37
StDev	66.98	SqSumPt(50)	4,913,656	AC_1	−0.20

- *Autocorrelation*. Just as correlation can be measured between two different dimensions, so also correlation can be calculated for a single dimension but displaced in time. By comparing sensor values with the values that occur at previous time steps, we can observe the amount of persistence in the data, or the tendency for the system to remain in the same state over a period of time. While any time delay can be considered, in Equation 3.12 we compute autocorrelation for a lag of 1, which compares a data point at time *i* with the data point at time i + 1. While an autocorrelation value approaching 1 indicates a high amount of persistence, a value close to 0 indicates near-randomness in the values.

Example 3.13 For the A*x* dimension of the sweeping activity sample, the lag−1 autocorrelation is −0.20. As this is for an accelerometer worn on the hip, the value may indicate many shifts between starting and stopping movement or changes in direction, both of which are common movement patterns while sweeping a floor.

$$\mathrm{AC}_1(S) = \frac{\sum_{i=1}^{N-1}(s_i - \bar{s})(s_{i+1} - \bar{s})}{\sum_{i=1}^{N}(s_i - \bar{s})^2} \tag{3.12}$$

Table 3.4 summarizes the continuous-based sensor features that we extracted from the raw hip accelerometer data for the Sweeping activities. We next turn our attention to features commonly used in signal processing of continuous-valued data sources.

- *Signal Energy*. Signal energy refers to the area between the signal curve and the time axis. For sensor values, this can be computed as the sum of the squared values. Related features are *log energy*, which is the sum of the log squared values, and *signal power*, which is a time-based average of energy.

$$\mathrm{E}(S) = \sum_{i=1}^{N} s_i^2 \tag{3.13}$$

$$\text{Log}E(S) = \sum_{i=1}^{N} \log(s_i^2) \tag{3.14}$$

$$\text{Power}(S) = \frac{1}{N}\sum_{i=1}^{N} s_i^2 \tag{3.15}$$

- *Signal Magnitude Area*. For sensors with multiple dimensions, the sensor values may be summed over all of the dimensions and averaged over time (in our case, over the current window). Equation 3.16 shows how to calculate the signal magnitude area for one such sensor, an accelerometer with x, y, and z dimensions.

$$\text{SMA}(S) \ = \sum_{i=1}^{N}(|x_i| + |y_i| + |z_i|) \tag{3.16}$$

- *Peak-to-Peak Amplitude*. This value represents the change between the peak (highest value) and trough (lowest value) of the signal. For sensor values, we can compute the difference between the maximum and minimum values of the set.

$$\text{P2PA}(S) = \text{Max}(S) - \text{Min}(S) \tag{3.17}$$

- *Time Between Peaks*. This value represents the time delay between successive occurrences of a maximum value. When processing sensor values that are not strictly sinusoidal signals, special attention must be paid to determine what constitutes a peak. A peak may be a value within a fixed range of the maximum value, or it may be a spike or sudden increase in values. The *number of peaks* in the window for different thresholds of the peak detector may also be included as a separate feature value.

Example 3.14 Consider the data plotted in Figure 3.14. In this case, the values at time points 3, 7, and 10 could be considered peaks. The number of peaks = 3 and time between peaks = $\{4, 3\}$ could be added to our feature vector.

Table 3.5 summarizes the signal-based sensor features that we extracted from the raw hip accelerometer data for the Sweeping activity. These features are commonly extracted from continuous-valued, sampled data sources.

3.3.4 Spectral Features

Many sensors used in activity monitoring are typically viewed as a signal, or a function of varying amplitude over time. The frequency spectrum provides an alternate view of the signal. This view shows how much of the signal lies within each given frequency band over a range of frequencies. A commonly used mathematical operator for transforming the time varying signal into its frequency spectra is the Fourier Transform, which is efficiently implemented using the Fast Fourier Transform (FFT). This transform decomposes the signal into the sum of a number of sine wave frequency components.

TABLE 3.5 Signal-Based Features for Sweeping Activity

A*x* Signal-Based Features for Sweeping Activity			
Signal energy	10, 935, 556.0	P2PA	186.0
Log signal energy	200.5	TBPeaks	3.5
Power	1, 093, 555.6	NumPeaks	3

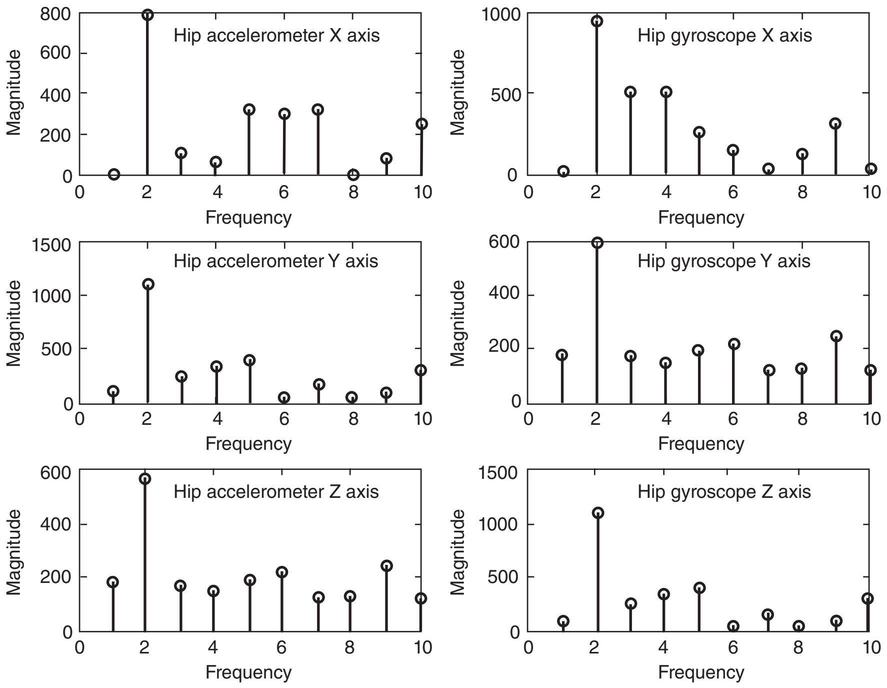


FIGURE 3.15 Frequency content in the accelerometer and gyroscope signal for the sensor placed on the hip while performing a Sweeping activity.

Example 3.15 Figure 3.15 illustrates the frequency content for our working example consisting of accelerometer and gyroscope data from the sensor located on an individual's hip. The corresponding spectral feature values are provided in Table 3.6. If we consider a window of 10 sensor readings, the frequency spectrum covers 10 frequencies. Note that converting the signal into its frequency components hides information regarding the frequency changes over time. Therefore, the time series signal is divided into short-duration overlapping windows from which the frequency content is extracted. More expressive features characterizing the signal can be extracted from the frequency spectra. We highlight a few such features here.

Let $\mathcal{F}$ represent the frequency spectrum for the window of sensor readings, s, of length N. Then $\mathcal{F} = \text{FFT}(s)$, which estimates the frequency content in the signal by applying FFT. Here, $\mathcal{F}(n)$ corresponds to the magnitude of the n^{th} frequency component.

- *Spectral Centroid*: This corresponds to the balancing point or the center of mass of the spectral power distribution. This value is calculated as the weighted mean of the frequency components present in the signal, as shown in Equation 3.18.

$$S_C = \frac{\sum_{i=1}^{N} \mathcal{F}(n) \times n}{\sum_{i=1}^{N} \mathcal{F}(n)} \tag{3.18}$$

- *Spectral Energy*: The equivalent to the energy of the signal as discussed before in the frequency domain is the spectral energy. This value is computed as the sum of the squares of the magnitude of the frequency content, as shown in Equation 3.19.

$$S_E = \sum_{i=1}^{N} \mathcal{F}(n)^2 \tag{3.19}$$

The energy can also be computed from the normalized frequency spectrum, or $\widehat{\mathcal{F}}(n) = \frac{\mathcal{F}(n)}{\sum_{i=1}^{N} \mathcal{F}(n)}$, in which case the equation is modified as shown in Equation 3.20.

$$NS_E = \sum_{i=1}^{N} \widehat{\mathcal{F}}(n)^2 \tag{3.20}$$

- *Spectral Entropy*: This refers to the entropy of the signal in the frequency domain. The frequency content of the signal is normalized before the entropy computation. Let $\widehat{\mathcal{F}}(n) = \frac{\mathcal{F}(n)}{\sum_{i=1}^{N} \mathcal{F}(n)}$ represent the normalized frequency spectrum. Then spectral entropy is calculated as shown in Equation 3.21.

$$S_{EN} = -\sum_{i=1}^{N} \widehat{\mathcal{F}}(n) \times \log(\widehat{\mathcal{F}}(n)) \tag{3.21}$$

As with the other features, we extract the spectral features for our Sweeping activity example. Table 3.6 summarizes these spectral sensor features.

TABLE 3.6 Spectral Features for Sweeping Activity

Spectral Features from Hip A*x* Data for the Sweeping Activity Window of 10 Events

Spectral centroid	4.9550	Spectral energy	1007864
Spectral entropy	1.8238	Normalized spectral energy	0.2002

3.3.5 Activity Context Features

All of the features that we discussed in this chapter are extracted from a single window of sensor events. However, the context that is defined from events outside the window can be influential as well in modeling and learning the current activity.

- *Previous Activity*. The activity label for the previous window can be very helpful in understanding the activity in the current window. For example, opening the front door often signifies that an individual is returning home if the previous activity was that person leaving home. The difficulty here is knowing the correct activity label for the previous window. The output of an activity recognition algorithm can be used here, although if the label is incorrect than the error will propagate to the feature vector describing the current window. Another approach is to input a probability distribution over possible activity labels for the previous window into the feature vector for the current window.
- *Previous Dominant Sensor*. Another context feature that is helpful is to note the sensor that generated a message most frequently in the previous window. For sampling-based sensors, these could also be the median value of the sensor messages.
- *Weighted Features*. In theory, all of the features extracted from earlier windows can be used as input to the current feature vector. However, the number of such windows may not be fixed and the resulting feature vector will be quite large. An alternative approach is to compute aggregates of the features from previous windows, such as a weighted sum of values. The values can optionally be weighted by a decay factor based on how far back in time the value was observed.

3.4 MULTISENSOR FUSION

Multisensor fusion refers to the synergistic combination of sensory data from multiple sensors to provide more reliable and accurate information. The potential advantages of multisensor fusion are redundancy, complementarity, and reduction in uncertainty of the information. The integration or fusion of redundant information can reduce overall uncertainty and thus serve to increase the accuracy with which the information is perceived by the system. Multiple sensors providing redundant information can also serve to increase reliability in the case of sensor error or failure. For example, if there are multiple wearable accelerometers that are gathering movement data related to activities, multisensor fusion aids in scenarios when some of the accelerometers fail or provide noisy data. The raw sensor data from all the accelerometers can be fused to obtain a more reliable estimate of the movement measurement.

Complementary information from multiple sensors allows features in the environment to be perceived that are difficult or impossible to perceive using just the information from each individual sensor operating separately. For example, in a smart home setting multiple sensors such as motion, temperature, and pressure sensors gather complementary data about a Cooking activity. Motion sensors can provide data about a human presence in the kitchen area, temperature sensors provide clues to whether the stove is on and pressure or vibration sensors can indicate whether any kitchen objects are being used. While these three sensor classes may independently be weak at characterizing the Cooking activity, fusing them together leads to a stronger model.

Information fusion is worthwhile for reducing uncertainty. Data gathered from sensors can be noisy and incomplete. As a result, activity learning methods that directly utilize raw sensor data are also prone to yield erroneous results. However, when different sensors and methods produce varying levels of errors and each method is reasonably accurate independently, a combination of multiple experts should reduce overall classification error and as a consequence yield more accurate outputs.

Typically, fusion of data/information can be carried out on three levels of abstraction closely connected with the flow of the activity learning process: data, feature, and classifier fusion. Of these, classifier fusion is more popular with the research community, due to sound theoretic principles that support different algorithms.

Data Fusion Multisensor data fusion refers to the stage in the integration process where there is an actual combination of different sources of sensory information into one representational format. Given a group of sensors, a weighted average of the redundant sensor data provides a simple approach to fuse the events generated by different sensors. Let x_1 and x_2 represent measurements from two sensors. The noise variances of the two sensors can be represented as σ_1^2 and σ_2^2, respectively. The fused sensor value that is generated by a weighted average algorithm can be calculated as shown in Equation 3.22. In this equation, $\sigma_3^2 = (\sigma_1^{-2} + \sigma_2^{-2})^{-1}$ represents the variance of the combined estimate.

$$x_3 = \sigma_3^2(\sigma_1^{-2}x_1 + \sigma_2^{-2}x_2) \tag{3.22}$$

While this is a simple approach that employs the noise variances of the sensors as the weights, applying a Kalman filter is more often the preferred approach as it results in estimates for the fused data that are optimal in a statistical sense. Additionally, the Kalman filter does not incur heavy computational expense. Kalman filtering assumes that the sensors can be modeled as a linear system and sensor noise can be modeled as a Gaussian distribution. The algorithm operates using two steps. In the first step, referred to as the prediction step, the Kalman filter estimates the current sensor values along with their uncertainties. Once the actual sensor values are observed, the fused estimate is obtained by the weighted average of the sensor values, with more weights given to the values with higher certainty. Kalman filtering can be performed in real time using the current sensor measurements and the previously-calculated sensor states with their uncertainty values.

Feature Fusion Feature fusion refers to the process of combining features extracted from multiple sensors. The naïve technique for fusing features from multiple sensors is to concatenate the features extracted from each sensor into a single, longer feature vector. This is an effective technique when the multiple sensors and features extracted from the raw sensor data are independent of each other. When the features extracted from different features are not independent, then dimensionality reduction or feature selection techniques that are discussed in Chapter 4 can be used to fuse the different features.

Classifier Fusion At the third level of sensor fusion, multiple individual classification models are first learned using data from each sensor independently. In the second step, the individual classifier outputs are combined to determine the actual output value. This type of fusion is also referred to as decision fusion or mixture of experts, and has been investigated with a number of proposed approaches. There are two common approaches for fusing output from the classifiers. The goal of the first approach is to find a single best classifier or a group of best classifiers whose outputs are then considered for making the final prediction. The second group of methods perform simple fusion functions. Such functions operate directly on the outputs of the individual classifiers to determine the best combination to make the prediction.

One popular technique, Dynamic Classifier Selection (DCS), extracts the single best classifier that is most likely to produce the correct classification label for a particular data point (such as a sensor event sequence). DCS partitions samples from sensor training data and selects the best classifier for each partition. Given a data point, we first determine the most likely partition to which it belongs and generate its label using only the output of the best classifier for the corresponding partition. The idea of using partition-based DCS is to estimate each classifier's accuracy in a local region of the feature space and generate a final output from the most locally accurate classifier. Different metrics can be employed for partitioning the training samples. These include partitioning samples into a subgroup for which there is agreement between the classifier output and the ground truth and a subgroup for which there is no such agreement and groupings based on the features of the input samples. DCS approaches strongly rely on available sensor event training data and hence can be unduly influenced by noisy data samples. Furthermore, choosing only the locally best classifier will result in loss of additional information that could have been gained from other classifiers for making a more accurate prediction.

Simple fusion functions are the most obvious choice when constructing a multiple classifier system. Common fusion functions are listed below.

- *Simple Voting Strategy*. Voting strategies can be applied to a multiple classifier system assuming that each classifier gives a single class label as output. Voting strategies include unanimous voting in which all classifiers agree on the same class label, simple majority in which more than half the number of classifiers agree on the same class label, and majority voting in which the class label is a label that is most often predicted by the classifiers.
- *Weighted Majority Voting*. If the classifiers in the ensemble do not have identical performance, then classifiers that are more accurate are assigned higher weights when making the final prediction. The accuracy of the classifiers can be used to determine the weights.

- *Highest Average Classification Probability*. When the classifiers also output posterior classification probabilities, a simple strategy is to pick the class label with the highest average probability of classification.

Bayesian methods are also applicable for classifier fusion when classifiers output posterior probabilities. For example, the classification output of multiple models can be combined using a naïve Bayes technique as described in Chapter 4. If $S_1, S_2, \ldots, S_K$ represent the classification output of K models that were independently trained using the data from K sensors, the fused prediction C can be estimated as shown in Equation 3.23.

$$P(C|S_1, \ldots, S_K) = \frac{P(S_1, \ldots, S_K|C)P(C)}{P(S_1, \ldots, S_K)} \tag{3.23}$$

Since classification models are independent of each other, we can rewrite this rule as shown in Equation 3.24.

$$C^* = \text{argmax}_c \left\{ P(C = c) \prod_{k=1}^{K} P(S_k = s_k|C = c) \right\} \tag{3.24}$$

The prior $P(C = c)$ can be estimated by computing the distribution of samples in every class c. The posterior likelihood of a particular output can be computed from the confusion matrix of model S_k as shown in Equation 3.24, where t_C is the number of training instances for which both the class $C = c$ and the classifier output $S_k = s_k$. The term t represents the number of total training instances.

$$P(S_k = s_k|C = c) = \frac{t_c}{t} \tag{3.25}$$

Another approach to classifier fusion is to formulate it as a supervised learning problem. This approach simply treats the outputs of multiple classifiers as input to a second-level classifier, and uses classical supervised learning techniques, such as those described in Chapter 4, to generate the final classification output. The selection of the second-level classifier depends on the distribution of its input, which in this case are probabilities that are skewed toward 0 and 1. As a result, algorithms that assume the data follows a normal distribution are not likely to fare well.

Classifier fusion is useful when activity information is gathered from a redundant set of noisy sensors and data from each sensor alone is sufficient to model the activity. The meta-classifier, a classifier that combines the output of multiple classifiers, exploits the redundancy intrinsic in the sensor network. An example of this situation is a setting where there are multiple accelerometers gathering movement information while an individual performs an activity. Accelerometers can be selected according to their contribution to classification accuracy as assessed during system training.

3.5 ADDITIONAL READING

The activity learning literature is filled with case studies that use environmental sensors including motion, temperature, door usage, and lighting sensors. Similarly, accelerometers, gyroscopes, GPS, and magnetometers are ubiquitous in smart phone-based activity modeling. Because of the inherent range limitations in RFID technology, researchers have experimented with individuals wearing an RFID reader such as the iGlove[18] and the iBracelet[19]. Ongoing research also exists that couples programmability with RFID tags in order to imbue them with additional sensing ability.[20] Zhu et al.[21] and Reddy et al.[22] explore the mining of GPS trajectory data to recognize activities, and Pu et al.[23] have tested wireless signals for action recognition.

We focus this discussion on sensors that are most commonly used for activity learning from sensor data. However, the number and variety of sensors that are developed for activity modeling grow as sensor technology matures and the applications that make use of activity learning become more diverse. Some examples are recent research that utilizes the Doppler effect to sense body movement[24] and research that uses voltage and measurement disaggregation to monitor device usage[25,26]. Bulling et al.[27] investigate a novel direction of eye movement analysis as a modality for learning activities. The examples in Appendix A illustrate the use of special-purpose sensors to monitor gas flow and water flow. Infrared proximity sensors are now found on mobile devices and are gaining traction for gesture recognition. Additional types of sensor modalities emerge each year and they can be incorporated into activity models as well.

While this chapter does not discuss video-based approaches to activity learning in depth, many of the techniques described here can be applied to video processing as well. Some additional features have been designed that are fine-tuned for processing information from CCD, depth, or infrared cameras[2,28–30] and from microphones[31]. While these sensors provide a dense grid of information that is valuable for analyzing the activity, the resulting wealth of data may also require substantial preprocessing in order to extract high-level features for modeling activities.

The discussion in this chapter highlights the many diverse sensors and features that can be employed for learning activities. Effective use of this information can depend on subtle choices of how to tune the sensors and utilize their data. For example, many different sampling rates have been reported for continuous-valued sensors. Yan et al.[32] investigate how the choice of sampling frequency and classification features affects accuracy and energy usage in different ways for each activity class. While we describe a method to convert discrete-event sensors to sampling-based sensors, some time series research, such as the work by Cardinaux et al.[33], converts binary-valued times series data to a start time/stop time representation that approximates the representation of the discrete event sensors. Electricity consumption by wireless sensors has also been investigated by Liu et al.[34], who propose a design that allows sensors to be effectively powered by ambient RF in the environment. Sensor placement is another important factor. Alternative placement locations on the body have been explored by Bao and Intille[35] and by Keally et al.[36], while the effect of placement on activity modeling has been evaluated by Hoseini-Tabatabaei[37] and by Kunze et al[38]. Environmental sensor selection and placement techniques have also been proposed and evaluated by Philipose et al.[39] and by Cook and Holder[40].

In addition to defining features that can be applied to raw sensor data, hierarchical features can also be defined. As an example, raw sensor data can be clustered using a technique such as density-based clustering or k means clustering. Features can be defined that describe the cluster characteristics[41,42]. By transforming the raw data provided by these sensors into higher-level features, the insights gleaned from a tremendous diversity of sensors can be utilized to create more robust models of human behavior.

Luo et.al.[43] describe the three levels for multisensor fusion and discuss techniques for fusion at each level. Durrant-Whyte and Henderson[44] provide a resource on techniques for fusing raw sensor data and Kalman filtering for data fusion. Ruta and Gabrys[45] provide an overview of methods for combining classifier outputs for multisensor fusion. Kuncheva[46] presents a thorough discussion on the theoretical underpinnings of classifier fusion strategies. Gao et al.[47] and Zappi et al.[48] employ multisensor fusion for combining data from multiple accelerometers for recognizing human locomotion. Hong et al.[49] discuss a methodology for fusing information gathered from uncertain smart home sensor readings using Dempster–Shafer theory.