Detecting Eating and Smoking Behaviors Using Smartwatches

Abhinav Parate and Deepak Ganesan

Abstract Inertial sensors embedded in commercial smartwatches and fitness bands are among the most informative and valuable on-body sensors for monitoring human behavior. This is because humans perform a variety of daily activities that impacts their health, and many of these activities involve using hands and have some characteristic hand gesture associated with it. For example, activities like eating food or smoking a cigarette require the direct use of hands and have a set of distinct hand gesture characteristics. However, recognizing these behaviors is a challenging task because the hand gestures associated with these activities occur only sporadically over the course of a day, and need to be separated from a large number of irrelevant hand gestures. In this chapter, we will look at approaches designed to detect behaviors involving sporadic hand gestures. These approaches involve two main stages: (1) spotting the relevant hand gestures in a continuous stream of sensor data, and (2) recognizing the high-level activity from the sequence of recognized hand gestures. We will describe and discuss the various categories of approaches used for each of these two stages, and conclude with a discussion about open questions that remain to be addressed.

Introduction

Human behaviors related to health such as eating, smoking, and physical activity levels, are widely regarded as among the best predictors of health and quality of life. An exciting opportunity that has emerged as a consequence of the growing popularity of wearable devices such as fitness bands, smartwatches, and smartphones is the ability to recognize and track these behaviors in a non-intrusive and ubiquitous

A. Parate (✉)
Lumme Inc., Amherst, MA, USA
e-mail: aparate@cs.umass.edu

D. Ganesan
University of Massachusetts—Amherst, Amherst, MA, USA

Lumme Inc., Amherst, MA, USA
e-mail: dganesan@cs.umass.edu

J.M. Rehg et al. (eds.), *Mobile Health*, DOI 10.1007/978-3-319-51394-2_10

manner. In turn, such real-time tracking has the potential to enable personalized and timely intervention mechanisms to alleviate addictive and unhealthy behavior.

A central question that needs to be addressed to enable this vision is whether we can reliably detect a broad range of behaviors using wearable devices. While many fitness bands monitor physical activity patterns such as walking, jogging, and running, this only represents a fraction of the range of behaviors we would like to monitor. Ideally, we should be able to use wrist-worn wearables to also detect patterns of hand movement that correspond to other key behaviors such as smoking and eating. Intuitively, this should be possible since behaviors that we perform with our hands involve characteristic *hand gestures*. For example, activities like eating food with a fork or smoking a cigarette seem to have a set of distinct hand-to-mouth gestures. For the case of eating, the distinct gesture is when a person takes the food from the plate using a fork towards the mouth, takes a food bite, and puts back the arm containing the fork to a relaxed position. Similarly, a set of distinct gestures appear to be present for the case of smoking a cigarette.

But the challenge is how to detect these gestures using the set of sensor modalities on a smartwatch, in particular, its inertial sensors (accelerometer, gyroscope, and compass). Our visual system can easily distinguish various hand gestures since it has the contextual information about the whole body. However, the inertial sensors on smartwatches and fitness bands only provide the movement patterns of the wrist, with no additional contextual information. Thus, the central challenge in recognizing gestures using a wrist-worn wearable arises from the need for *gesture spotting* i.e. matching a meaningful pattern for a gesture type in a continuous stream of sensor signals. While recognition of gestures from inertial sensors is commonplace in gaming devices (e.g. Nintendo Wii), these assume structured environments where the user intentionally makes a gesture that the system can interpret. In contrast, we need to spot gestures in natural settings where there are a wide range of hand movement patterns.

This raises three key challenges. The first is that there are many confounding gestures that have very similar arm movement patterns as the behavior that we want to detect. For example, a smoking gesture can be confounded by other gestures that involve hand-to-mouth movements such as eating and drinking. The second is that there are many irrelevant hand gestures that need to be filtered. For example, during an eating session, a person may employ a variety of hand gestures such as cutting food with a knife, switching knife and fork between hands, grabbing a bowl to fetch food, serving the food on plate and conversational gestures. The third is that hand gestures associated with the health-related activities like eating or drinking are sporadic in nature. For example, an eating session may last for half an hour and yet a person may have taken less than 25 food bites scattered across the session, with hundreds of irrelevant gestures in between. Fourth is that even for a single individual, there is variability in the gesture corresponding to the target activity due to the contextual circumstances, for example, smoking while walking, driving, or standing still.

In this chapter, we survey state-of-the-art approaches for robust detection of behaviors such as smoking and eating from the continuous stream of signals

generated by the embedded inertial sensors on a smartwatch. There are many pieces to this puzzle including extraction of distinguishing features from inertial signals (e.g. speed, acceleration and displacement of the arm or angular velocity, roll angle about the fore arm), robust methods to segment the data streams into potential gesture windows, and classification methods that leverage domain knowledge regarding the pattern of wrist movements corresponding to smoking or eating.

Gesture-Driven Activity Recognition: An Overview

The high level goal of gesture-driven behavior recognition is to find a temporal partition of a given time-series of sensor signals and assign a set of labels to each partition representing activities performed during that interval. Figure 1 gives an overview of the general computation pipeline used in detecting hand gestures and extracting sessions of high level activities. At the lowest layer of the pipeline is a sensing layer that obtains data from one or more inertial sensors, typically worn on the wrist for the hand gesture recognition task.

The second layer in the pipeline is the segmentation layer that extracts segments containing candidate gestures from the continuous time-series of raw sensor signals, and filters out extraneous data. This is a non-trivial problem since the gesture durations can vary even for a single individual, hence canonical fixed window-based segmentation methods are inadequate. The window sizes need to be carefully chosen for each gesture to ensure that the extracted segments are neither too short to contain a complete gesture nor too large and contain extraneous gesture data, both of which can lead to classification errors. This layer should also act as an early-stage filter to remove segments containing gestures that are unlikely to be relevant for the higher-level activity recognition.

The third layer of the pipeline is a gesture recognition layer that recognizes and outputs the label for the recognized gesture type. This layer first computes a feature vector for each segment identified earlier, consisting of features that can discriminate hand gestures relevant to the target activity (smoking, eating, drinking, etc.) from a large number of other confounding gesture candidates. This layer computes the feature vector from the available sensor signals such as acceleration, angular velocity, etc. and from the derived signals such as *roll* and *pitch* (refer section "Sensing Layer"). The feature vector is then provided as an input to a supervised classification algorithm, that outputs the label for the type of gesture present in a segment ("smoking","eating with a fork","eating with a spoon", etc.) and a probability associated with the output.

The top-most layer in the processing pipeline is an activity recognition layer that identifies the whole sessions of recognized activities from a sequence of recognized gestures. The key intuition behind this layer is that each session of an activity such as smoking involves a continuous sequence of smoking gestures. Sessions for gesture-driven activities like eating and smoking are often characterized by features such as inter-gesture interval, session length, and number of relevant gestures in a session.

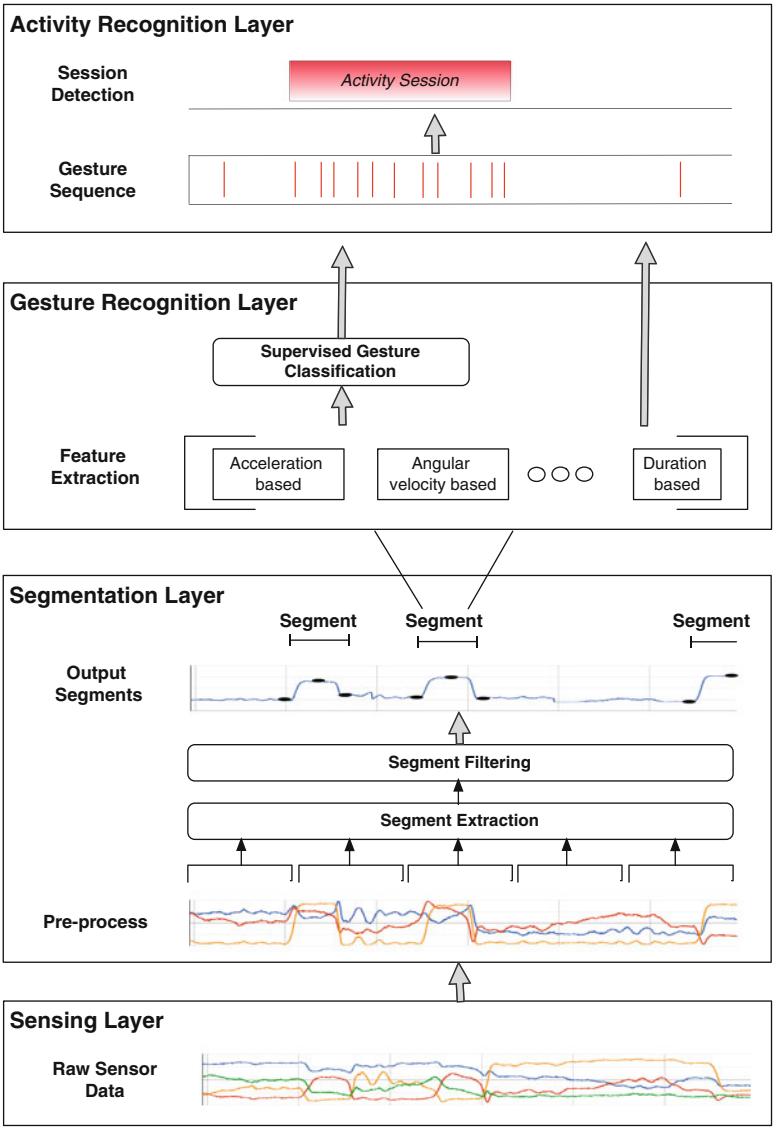


Fig. 1 Gesture-driven activity recognition pipeline

This layer utilizes such session characteristics to detect activity sessions and its boundaries (the start and the end of a session). Further, the detected activity sessions are used to filter out spurious and isolated gestures recognized at the lower layer, making the entire pipeline more accurate and robust.

Table 1 gives an overview of the state of the art approaches used in spotting hand gestures and the algorithms used in the different stages of the computational pipeline

Table 1 State of the art approaches in gesture spotting

Reference	Activity classes	Segmentation[a]	Gesture recognition	Activity recognition	Sensors (sensor placement)
Parate et al. [11]	Smoking, eating	KB	Random forests	Conditional random fields	Accel, gyro, compass (wrist)
Saleheen et al. [13]	Smoking	KB	SVM	Rule-based	Accel, gyro, RIP (wrist and chest)
Varkey et al. [17]	Daily activities including smoking	TB	SVM	SVM	Accel, gyro (wrist, ankle)
Tang et al. [15]	Smoking	SB	Random forests	Distribution score based	Accel (two wrists)
Thomaz et al. [16]	Eating	SB	Random forests	DBSCAN	Accel (wrist)
Amft et al. [2]	Drinking, eating	TB	HMM	N/a	Accel, gyro (two wrists, two upper arms)
Junker et al. [7]	Activities including drinking and eating	TB	HMM	N/a	Accel, gyro (two wrists, two upper arms, torso)
Amft et al. [3]	Dietary activities: eating, chewing, swallowing	TB	HMM	N/a	Accel, gyro, compass (two wrists, two upper arms), microphone (ear, collar), EMG (collar)
Amft et al. [1]	Drinking	TB	Feature similarity	N/a	Accel, gyro, compass (wrist)
Scholl et al. [14]	Smoking	SB	GMM	Rule-based	Accel (wrist)
Dong et al. [6]	Eating	N/a[b]	Rule-based	N/a	Gyro (wrist)

[a]Segmentation techniques can be knowledge-based (KB), training-based (TB), fixed size sliding window-based (SB)
[b]These techniques do not require prior segmentation

discussed above. However, not all the approaches follow the computational pipeline exactly as shown in Fig. 1. For example, the approach used by Dong et al. [6] does not require explicit segmentation or the feature extraction. In some cases, one may use the classification algorithms like Hidden Markov Model (HMM) that can operate over the continuous sensor signals without the need to extract features. We discuss these variants when we get into details of each layer in the computational pipeline. We also note that some approaches employ multiple sensing modalities such as a microphone, a RIP sensor that monitors breathing waveforms, or multiple on-body inertial sensors. However, we limit our focus on techniques relevant to gesture spotting using commercial smartwatches i.e. using only the inertial sensors worn on a wrist.

Sensing Layer

The term *inertial sensors* is often used to represent a suite of sensors consisting of an accelerometer, gyroscope, compass and an altimeter (also known as barometer). An electronic device embedding a subset of inertial sensors is referred as an *Inertial Measurement Unit* (IMU). The term IMU is widely used to refer to a device containing 3-axis accelerometers and 3-axis gyroscopes. However, IMUs including 3-axis magnetometer(also called compass) and 1-axis barometer are increasingly available in the market. Most commercial smartwatches come fitted with IMUs including accelerometers, gyroscopes and often, magnetometers.

Frame of Reference

Let us first understand the frame of reference used by the inertial sensors before we get into the details of signals captured by these sensors. Figure 2 shows the frame of reference used by commercially available smartwatches based on the Android-wear operating system. This frame is defined relative to the screen/face of the watch and has three mutually perpendicular axes. The x and y axis are along the screen surface whereas the z axis points away from the screen. This frame of reference is *local* and is fixed with respect to the body of the watch.

Sensor Signals

An inertial measurement unit captures signals observing the linear translation as well as the rotation experienced by it. An accelerometer measures the physical acceleration experienced by an object. An object at rest experiences an acceleration equal to the earth's gravity ($g = 9.8\,\mathrm{m/s^2}$) in the direction pointing away

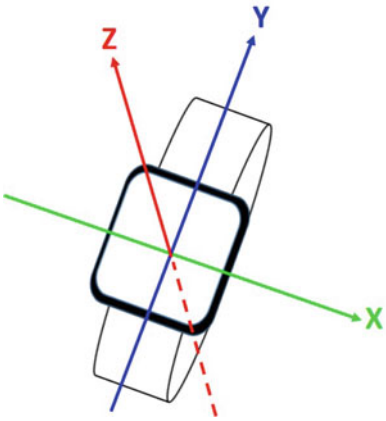


Fig. 2 Figure showing the frame of reference and the axis orientations used in Android-wear OS based smartwatches. The x and y axis of the frame of reference are along the face of the watch and are mutually perpendicular to each other. The z axis points towards the outside of the face of the watch. The coordinates behind the screen have negative z values

Table 2 Feature signals computed from the accelerometer signals $\langle a_x, a_y, a_z \rangle$

Signal	Notation	Definition
Tilt	ρ	$\arccos \frac{a_z}{\sqrt{a_x^2+a_y^2+a_z^2}}$
Pitch	ϕ	$\arctan \frac{a_y}{a_z}$
Roll	θ	$\arctan \frac{a_x}{\sqrt{a_y^2+a_z^2}}$

from the ground. On the other hand, a gyroscope sensor measures the angular velocity or the rate of rotation about an axis, expressed in *radians/second* or *degrees/second*. A compass measures the ambient magnetic field using μT as the unit of measurement. An IMU measures these signals along each of the three axis in its *local frame of reference*.

Tilt, Pitch, Roll, Yaw

Apart from the raw sensor signals, we can derive more useful signals using one or more of the inertial sensors. For example, *tilt* i.e. the angle ρ between the z axis of the device and the direction of the gravity, can be computed using the accelerometer signals. *Roll, pitch* and *yaw* are the angles that capture the orientation of a device in a 3D space. *Yaw*(ψ), *pitch*(ϕ) and *Roll* (θ) angles at a particular orientation indicate the amount of rotation around z, y and x axis respectively applied in that order to reach the particular orientation from the default orientation of the device. Pitch and roll angles can be computed using the accelerometer signals when there is no linear acceleration i.e. the device is stationary. Yaw angle can be computed using sensor fusion algorithms [10] that combine the signals from accelerometer, gyroscope and compass (optional) to accurately obtain all the orientation angles. Table 2 gives the mathematical functions to compute these signals using the accelerometer signals.

Android wear operating system provide APIs to obtain the signals such as tilt and orientation using several combinations of inertial sensors. Use `SensorManager` with the appropriate sensor type to get these signals. For example, sensor type `Sensor.TYPE_GEOMAGNETIC_ROTATION_VECTOR` uses accelerometer and compass, `Sensor.TYPE_GAME_ROTATION_VECTOR` uses accelerometer with gyroscope whereas `Sensor.TYPE_ROTATION_VECTOR` uses accelerometer, gyroscope and compass to compute the same orientation information.

An Example of Sensor Signals for a Hand Gesture

Figure 3 shows the sensor signals obtained using an Android-wear smartwatch for a hand gesture representing taking a cigarette puff. We can break this gesture into the following three stages: (1) the arm carrying the cigarette begins from a relaxing position and moves to bring the cigarette towards the mouth, (2) the arm remains stationary at the mouth when the person holding the cigarette takes a puff, and (3) the arm falls back to the original relaxing position. The watch is worn on the person's dominant hand (right hand in this case). At the beginning of this gesture, the arm is hanging vertically down such that the positive X axis of the smartwatch is in the direction opposite to the gravity, the acceleration along Y and Z axis is close to zero as the arm is not moving and these axes being parallel to the ground do not experience any gravity, and the angular velocities measured by the gyroscope are close to zero as the arm is not moving. We make the following observations about the three stages of this gesture:

- The acceleration along the watch's X axis goes from value closer to $+g$ to a value close to $-g$ when the arm reaches the mouth. The same transition is observed in reverse when the arm moves away from the mouth and falls back to the relaxing position.
- The acceleration along Y and Z axis hovers around zero except for the transitions when the arm is moving between the relaxed position and the mouth. The transition when arm is moving towards the mouth is similar but in the reverse order of the transition when the arm is moving away from the mouth.
- The angular velocities reach its peak when the arm is moving between the two positions. The reverse of the towards-mouth movement trend is observed when the arm goes away from the mouth. However, the sign of the angular velocity changes because the angular movement is now in the opposite direction.
- The transitions are similarly visible in the magnetometer readings. However, unlike accelerometer and gyroscope signals, the values of the signals depend on the direction the person is facing. If the person repeats the exact same gesture

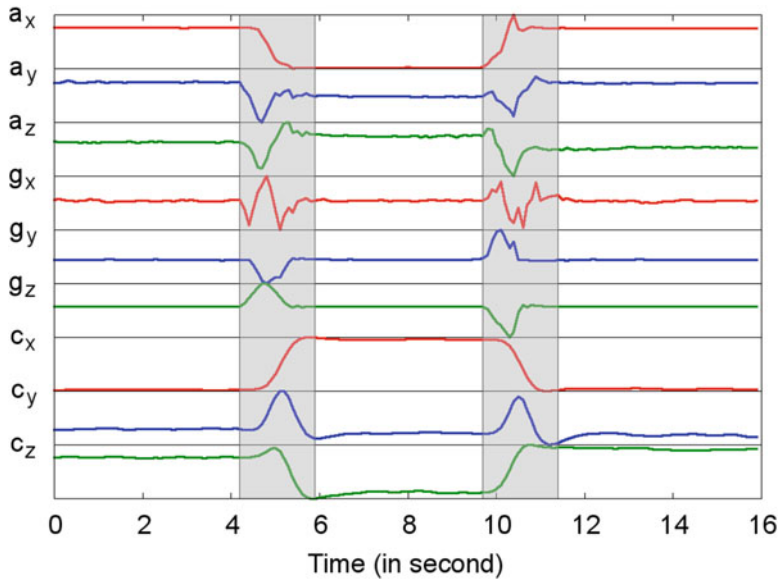


Fig. 3 Sensor signals observed for smoking a cigarette puff hand gesture. The value triplets $\langle a_x, a_y, a_z \rangle$, $\langle g_x, g_y, g_z \rangle$ and $\langle c_x, c_y, c_z \rangle$ present the signals of accelerometer, gyroscope and compass respectively. The *first gray shaded area* marks the interval when the hand holding the cigarette is moving towards the mouth. The *second gray shaded area* marks the interval when the hand falls back to a resting position. The period in between the two areas is when a person is taking a puff without moving the hands

once facing north and once facing east, the accelerometer and the gyroscope signals will look the same but the signals from the magnetometer will look very different.

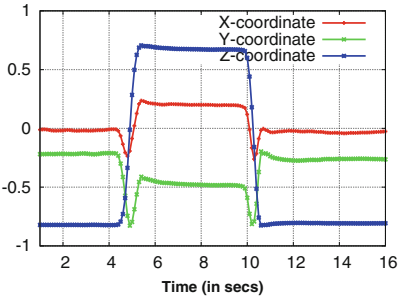
Time-Series Segmentation of Sensor Signals

In this section, we present a survey of techniques used in the extraction of segments from the time-series containing raw sensor signals. The aim of the segmentation process is to identify temporal segments that are likely to contain candidate gestures and filter out any extraneous data. This is a critical step in the gesture-driven activity recognition pipeline because if segments are too long, they result in noisy features, and if segments are too short, they result in inaccurate features. In the following, we look at the various categories of segmentation approaches studied in the literature.

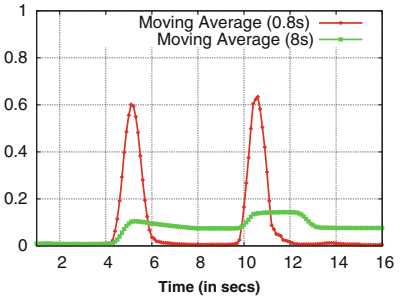
Knowledge Based Segmentation

Some of the most promising approaches for recognizing behavioral activities such as smoking and eating, rely on the domain knowledge about the gestures in the activities being observed. In general, hand-to-mouth gestures such as smoking a cigarette, taking a food bite, or drinking from a cup tend to have different characteristics in terms of the duration of the gesture and the pace of the wrist movement while performing the gesture. But one common characteristic is that a person performing the gesture starts from "a rest position" in which the arm is relaxed, then move their arm towards the mouth, keep the arm stationary at the mouth for a short duration, and finally, move their arm back to a possibly different rest position in the end. Thus, hand to mouth gestures tend to lie between these resting positions. In the following, we discuss the two approaches that use this observation to extract gesture segments.

Parate et al. [11] use the characteristic property of hand-to-mouth gestures in the extraction of segments containing gestures like taking a food bite or taking a cigarette puff. In this approach, the authors track the rest positions of an arm by computing the spatio-temporal trajectory taken by the wrist in a 3D space from the wrist orientation data. Figure 4a shows an example of a spatio-temporal trajectory of the wrist performing a smoking gesture. Using this spatio-temporal trajectory, the rest point can be identified as the centroid of the extremely slow moving points in the trajectory time series. In any hand-to-mouth gesture, the spatial distance of the wrist from the rest point first increases rapidly when the arm is moving towards the mouth and away from the rest point, plateaus for a short period when the hand is at the



(a) Wrist coordinates



(b) Moving average of gyroscope magnitude

Fig. 4 A person performing the smoking gesture starts from "a rest position" in which the arm is relaxed, then move their arm towards the mouth, and move their arm back to a possibly different rest position in the end. Thus, hand to mouth gestures tend to lie between these resting positions. (**a**) The segment between the resting positions can be identified from the time-series of wrist-coordinates by tracking the periods of large arm movements. (**b**) The period of large arm movements can also be obtained by using two moving averages of the gyroscope magnitude computed over windows of sizes 0.8 s and 8 s respectively

mouth, and then decreases rapidly when the arm falls back to the rest point. Thus, we can extract the segments containing a hand-to-mouth gesture by computing a time-series of spatial distance of the wrist from the rest point and looking for a tell-tale pattern of rise, plateau and fall in the time-series.

Saleheen et al. [13] identify the periods of quick arm movements before and after the smoking puff from the accelerometer and the gyroscope signals. They use the observation that the arm movements around the stationary smoking puff period is associated with a change in the angular velocity and can be observed as peaks in gyroscope signals. The example of such peaks can be seen in the gyroscope signals in Fig. 3. In this approach, the task of extracting segments between two consecutive peaks in gyroscope is accomplished using two moving averages computed over windows of different sizes to detect the rise and fall in the gyroscope magnitude given by $\sqrt{g_x^2 + g_y^2 + g_z^2}$). The first moving average is computed over a small window (0.8 s) that adapts to the changing values faster than the second slower moving average computed over a larger window of 8 s. Figure 4b shows an example of these moving averages for the example smoking gesture from the Fig. 3. Next, the segment extraction is done by identifying the following points: (1) P_R: time when the fast moving average drops below the slow moving average, and (2) P_F: time when the fast moving average rises above the slow moving average. In a smoking gesture, P_R marks the event when the cigarette reaches the mouth and P_F marks the event when the cigarette moves away. Thus, the segment between consecutive P_F and P_R gives a potential stationary-at-mouth segment of the hand-to-mouth gesture. An additional check is made to ensure that the arm is facing up during the identified segment by verifying that the acceleration along the x axis is negative.

Training Based Segmentation

In this section, we look at a set of approaches towards segmentation that extract dynamic size segments by learning the segment characteristics from the training data. The main intuition behind the approaches in this category is that one can exhaustively search for a candidate gesture segment among all the possible segments of varying sizes in a time series of sensor signals. The candidate gesture segment can be identified by matching each possible segment with the characteristics of true gesture segments observed in the training data. Segments with a matching score higher than a certain threshold can be selected as the candidate gesture segment. However, an exhaustive search over all the segments is not practical as a time-series containing n data points can have $n(n-1)/2$ segments. Thus, we need to limit the number of segments to search. In the following, we look at the approaches describing (1) a segment search-space reduction method, and (2) an algorithm to compute a matching score for a segment.

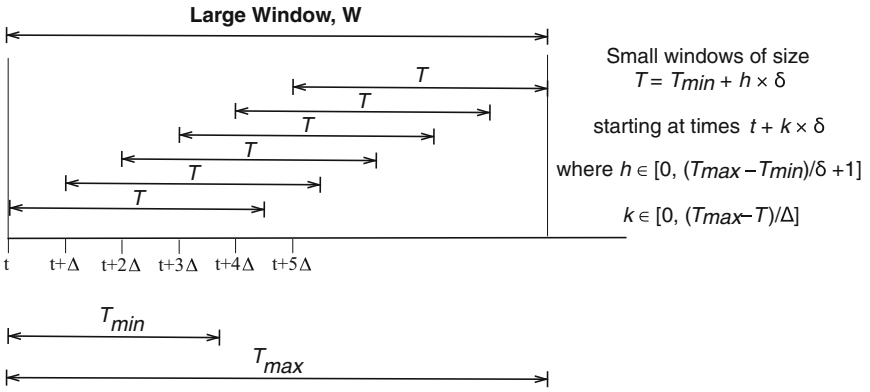


Fig. 5 The search space for segments is reduced by limiting the size of the segment and the points where these segments can begin

Search-Space Reduction

We first look at the ways we can reduce the segment search space.

Varkey et al. [17] propose a simple way to reduce the search space by limiting the size of the segments being considered. Most gestures have a lower bound T_{min} and an upper bound T_{max} on the duration needed to complete a gesture. Thus, we can search the segments of duration T where the value of T begins with T_{min} and is incremented by a value δ. This limits the number of possible sizes to $(T_{max} - T_{min})/\delta + 1$. Initially, this approach restricts the search of segments in a larger window W in the time series of sensor signals where the size of window W is equal to T_{max}. Within this window, the segments of size T can be obtained by starting from various possible positions. The number of such positions can still be large leading to a large number of segments to search from. To further reduce the search space, we avoid the segments that are too close to each other. This is done by selecting the starting points for the segments shifted by a period Δ. This approach limits the number of segments of size T to $\lfloor \frac{T_{max}-T}{\Delta} + 1 \rfloor$. Figure 5 illustrates the segment generation process.

Amft et al. [2] use a technique based on the natural partitioning of a gesture into "motion segments". These smaller segments are described as non-overlapping, atomic units of human movement, characterized by their spatio-temporal trajectories. For example, a smoking gesture can be divided into three natural partitions: hand approaching the mouth, hand remaining stationary at the mouth, and hand moving down towards a relaxed position. Thus, a gesture segment is composed of two or more consecutive motion segments. Hence, the search for a gesture segment can be limited to those candidate segments in the data whose boundaries coincide with the boundaries of the motion segments. To divide a continuous stream of data into non-overlapping consecutive atomic segments, an algorithm by Keogh et al. [8] called sliding-window and bottom-up (SWAB) algorithm is used. This algorithm

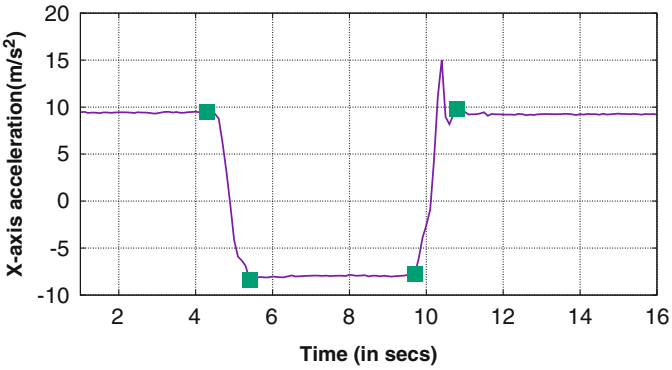


Fig. 6 Motion segments generated using the SWAB algorithm [8] over x-axis acceleration observed for a smoking gesture. A gesture segment is composed of two or more consecutive motions segments

partitions a time series into segments such that each segment can be approximated using a line segment. Figure 6 shows an example of such partitioning obtained from the time-series of acceleration along the x axis of a wrist band for a smoking gesture. Each segment in this figure can be approximated using a line segment such that the error in the approximation is below a chosen threshold.

Having identified the motion segments, a coarse search based on the motion segment boundaries can be used to efficiently find sections that contain relevant gestures. The search is performed by considering each motion segment's endpoint as a potential end of a gesture. For each endpoint, potential start points can be derived from preceding motion segment boundaries. To further confine the search space, the following two constraints learnt from the gesture training data are used to limit the section to be searched:

1. The duration T of a section spanning consecutive motion segments must lie in the range $[T_{min}, T_{max}]$ given by the minimum and the maximum duration for a gesture observed in the training data.
2. The number of motion segments n in a section must lie in the range $[n_{min}, n_{max}]$ where n_{min} and n_{max} give the minimum and maximum number of motion segments observed for the gesture in the training data.

Segment Matching

In the previous sub-section, we described how to obtain a reduced number of segments from the large search space. Now, we look at the process of obtaining a score for these segments to be used for identifying the gesture segments. As we described earlier, this approach requires training data with labeled gesture segments. Since we have the labeled gesture segments, we can potentially use any supervised

classification approach. For example, Varkey et al. [17] use a support vector machine (SVM) classification algorithm for identifying segments containing gestures for activities like smoking, eating, and brushing teeth. The trained SVM classier can be used to compute a matching score, d, as follows:

$$d(\mathbf{w}, b; \mathbf{x}) = \frac{\mathbf{w}.\mathbf{x} + b}{||\mathbf{w}||}$$

where $\mathbf{x}$ is a feature vector computed for a segment and $\mathbf{w}$ is a normal vector learnt using the training data. The segment with the best score is extracted as the gesture segment.

Another popular approach for matching segments is using a score based on the *feature similarity*. To accomplish this, a feature vector $\mathbf{f} = f_1, f_2, .., f_k$ is computed from the sensor data for the search segment. Using the training data, the parameters μ_i and σ_i representing the mean and the standard deviation of the i^{th} element of the feature vector of a true gesture are obtained. Now, a distance metric d is computed as follows:

$$d(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\sigma}) = \sqrt{\sum_{i=1}^{k} \left(\frac{f_i - \mu_i}{\sigma_i}\right)}$$

Any segment with distance less than the pre-computed threshold is accepted as a candidate gesture segment. The threshold is typically chosen using the sensitivity analysis over the training data containing relevant as well as irrelevant gestures. A lower value of threshold reduces the number of false positive segments but discards many true gesture segments. A larger value of threshold improves the number of true gesture segments identified but also increases the number of false positive segments. A threshold that results in highest recall of true gestures segments with minimum number of false positives is chosen. The choice of features is dependent on the type of sensors being used and the target gesture activity [1–3, 7]. For example, Junker et al. [7] used relatively simple features such as minimum and maximum values for pitch and roll angles obtained from the four on-body sensors (one sensor on each wrist and one sensor on each upper arm) for drinking and eating detection. Amft et al. [1] used a set of 200 features derived from the 3-axis accelerometer, 3-axis gyroscope, and 3-axis compass signals.

Sliding Window Based Segmentation

Some recent efforts have studied the problem from a gesture containment perspective i.e. they seek to verify if a given segment of data contains a partial or a complete gesture. Unlike the previous two segmentation approaches, this approach does not give us the precise boundary of the gestures but just seeks to contain it.

In this approach, a segment is obtained by placing a window of size $|W|$ at the beginning of the sensor signal stream and selecting the data from that window as the segment. The next segment is obtained by sliding the window over the sensor signal stream by a parameter Δ. This results in removal of the oldest data and adds new data to the latest segment.The process is continued to get more segments. Typically the value of Δ is chosen to be smaller than the window size $|W|$, and hence, the consecutive segments obtained using this protocol overlap with each other. One limitation of this segmentation approach is that if two consecutive segments are recognized to contain a gesture, one cannot estimate if these segments contain two distinct gestures or one gesture spread across both the segments. Although this limitation does not impact activity recognition performance, it can result in miscounting the number of gestures.

Selecting the Sliding Window Size

The size of a sliding window is an important parameter as it can impact the performance of later stages in the activity recognition pipeline. For example, a very small window size may not be sufficient to capture enough gesture characteristics to solve the gesture containment problem. In [16], Thomaz et al. use an approach where the sensitivity of the window size is analyzed. In this approach, the window size is varied between the minimum and the maximum duration of the gesture observed in the training data. The window size that results in the best classification results over the training data is chosen as the window size to be used on the test data. Scholl et al. [14] use a window of size 5.4 s to generate non-overlapping segments to detect smoking gestures. The window size is fixed based on the domain knowledge and is equal to the mean length of two subsequent hand-to-mouth gestures (raising a hand to take a cigarette to the mouth).

Gesture Classification

Having discussed approaches to extract relevant segments from the continuous time-series signals from the inertial sensors, we now look at the third layer in the gesture-driven activity recognition pipeline. In this layer, we recognize the gesture contained in a segment extracted from the lower layer. This gesture recognition task can be executed by first extracting a feature vector for each segment and then, by using a supervised classification algorithm to get the label of the gesture in the segment represented by the feature vector.

Features

In Table 3, we describe the set of features proposed in literature for the gesture recognition task using the inertial sensors.

Acceleration-Based Features Many state of the art approaches for detecting eating and smoking gestures use a 3-axis accelerometer. Most of these features are generic features that have been shown to be useful in a range of activity recognition contexts (e.g. statistical features and zero-crossings). Some look for correlations across axes and crossings between axes that are observed during hand-to-mouth gestures.

Orientation and Angular Velocity Based Features Wrist orientation, i.e. pitch and roll, is often used to extract a range of features. One key challenge in extracting these features is to determine the specific window during which they need to be computed—for example, [11] computes these features during the ascending stage (when the hand is moving towards the mouth) and the descending stage (when the hand is moving away from the mouth), and [13] computes features over the period when the hand is stationary at mouth. Several approaches also extract information about wrist rotation that is useful for gesture-based detection.

Gesture Duration Based Features Gesture duration is an important feature and is found to be useful in the segmentation: either to define the search space for the segments [14, 17] or for early filtering of the candidate segments that lack relevant gestures [2, 3, 7, 13, 16]. For example, Parate et al. [11] found that the duration of the sub-gestures can be a useful feature in hand-to-mouth gesture classification.

Trajectory-Based Features This class of features were studied in [11] where the explicit trajectory of the wrist performing the gestures in a 3D space is computed. From the 3D trajectory, several features were extracted corresponding to different segments of a hand-to-mouth gesture.

Gesture Classification

We now look at some of the classification algorithms used in recognizing gestures using the feature vectors computed for a gesture segment. In general, many popular classification methods including random forests, support vector machines, and hidden markov models have been used for the classification task.

Random Forest A Random Forest [5] is a popular classification method and has been shown to achieve high gesture recognition accuracy in many gesture classification scenarios [11, 15, 16]. This is because hand gestures show variations in the shape and duration even when performed by the same person. Thus, the type of gestures are likely to be correlated with a certain range of values of segment features, which makes decision tree-based methods a good choice. Unfortunately,

Table 3 The set of features proposed in the literature for gesture classification using inertial sensors

Feature set		
Acceleration features		
Statistical	Mean, variance, maximum, minimum, median, kurtosis, skew, signal-to-noise ratio (SNR), root mean square (RMS) computed for each axis	[1, 15–17]
Peak-peak amplitude	This feature gives the difference between the maximum and the minimum acceleration observed over a window. This feature is computed for each axis	[15, 17]
Correlation coefficients	This feature measures the correlation between the acceleration readings for any pair of axes	[15]
Mean-level crossing rate	This feature computes the rate at which the signal crosses the mean value over a segment	[1]
Crossing rate between axes	This feature is computed for each pair of axes and can be computed as the number of times accelerometer readings in these axes cross each other. The crossing behavior is often observed for hand-to-mouth gestures	[15]
Regression features	*Slope, mean squared error (MSE), R-squared* are used as the features capturing the relative trend change within a segment	[15]
Angular velocity features		
Statistical	Mean, variance, maximum, minimum, median, quartile deviation computed for each axis. [13] computes these features for the magnitude of the angular velocity (l^2*-norm* of the three velocities) as well. Parate et al. [11] compute a subset of these features separately for the ascending stage (when the hand is moving towards the mouth) and the descending stage (when the hand is moving away from the mouth)	[2, 7, 11, 13]
Orientation features		
Statistical	Mean, variance, maximum, minimum, median, quartile deviation, nine decile features from the distribution computed for the orientation values (pitch and roll)	[2, 3, 7, 11, 13]
Net change in roll	This feature computes the net angular change about the axis of the arm while performing a gesture	[2, 7, 11]
Duration features		
Gesture duration	Gesture duration is found to be useful in defining the search space for the segments [14, 17], and for early filtering of the candidate segments that lack relevant gestures [2, 3, 7, 13, 16]. In [11], Parate et al. found that the duration of a sub-gesture when the arm is ascending towards the mouth is a useful feature to distinguish between smoking and eating gestures	[2, 3, 7, 11, 13, 16]
Trajectory features		
Velocity features	In [11], Parate et al. use the spatio-temporal trajectory to compute the instantaneous speeds of the wrist. Using these, they extract mean, maximum, and variance of the wrist speed for the various stages in a gesture as features	[11]
Displacement features	Maximum vertical and horizontal displacement of the wrist during a hand-to-mouth gesture	[11]

The table describes each feature type and gives the relevant references

fitting a single decision tree to the training data results in poor performance for gesture recognition when used across the general population. The reason is that a single decision-tree yields predictions that are suited only for segments whose features are very similar to the ones of the training segments. However, there exist small variations in the way gestures are performed across the population. Thus, the decision-making thresholds used in a single decision tree do not work for the general population. Random Forests offer a popular alternative i.e. to build an ensemble of decision trees where each decision tree is fitted to small different random subsets of the training data. This leads to decorrelating the individual tree predictions and, in turn, results in improved generalization and robustness [5]. This ensemble of decision trees is called random forest classifier.

Support Vector Machine (SVM) SVM [4] is another popular supervised classification algorithm and has been used in some work on gesture recognitions [13, 15, 17]. In general, SVM is a good choice where the types of classes can be separated by a set of hyperplanes in a finite-dimensional space defined by the features i.e. where the training instances of classes are linearly separable. However, this is not the case with gesture recognition tasks as the several types of gestures cannot be separated linearly in a space defined by the features. To address this problem, a kernel function is used that maps the original finite-dimensional space into a much higher-dimensional space where a set of hyperplanes can be found to perform the linear separation, but this method increases computational complexity.

Hidden Markov Model (HMM) HMM is a statistical Markov model that can be applied to analyzing time-series data with spatial and temporal variability. Hand gestures typically have a relatively strict temporal order of sub-gestures in it but the same gesture, when repeated by an individual, shows some variability in the shape of the hand trajectory and in the duration of the sub-gestures. Since HMMs allow a wide range of spatio-temporal variability, it is a good fit for matching the gesture data with the reference patterns. Moreover, with a long history of use in various domains, HMM has elegant and efficient algorithms for learning and recognition.

While there are various topologies used for modeling HMM states, for the task of gesture recognition, a left-to-right topology (Fig. 7a) is used. In this topology, a state can transition to itself or can go forward to the following states on the right but can never go back to a previously visited state. This topology is suitable for hand gesture modeling as it imposes the temporal order observed for hand gestures. Another frequently used topology is a left-right-banded topology (Fig. 7b) in which a state can transition to itself or to the next state only. The skipping of states is not allowed in this topology.

One important parameter in a HMM is the number of hidden states. In general, the number of states depends upon the complexity of the gesture being modeled and is empirically determined. For instance, Amft et al. [2] varied the number of states between 3–10 for drinking and eating gestures. They observed that the recognition performance increased marginally with more than five states. Similarly Junker et al. [7] trained a Left-right-banded model with 4–10 states for various daily activity gestures where the optimal number of states varied with the type of gestures.

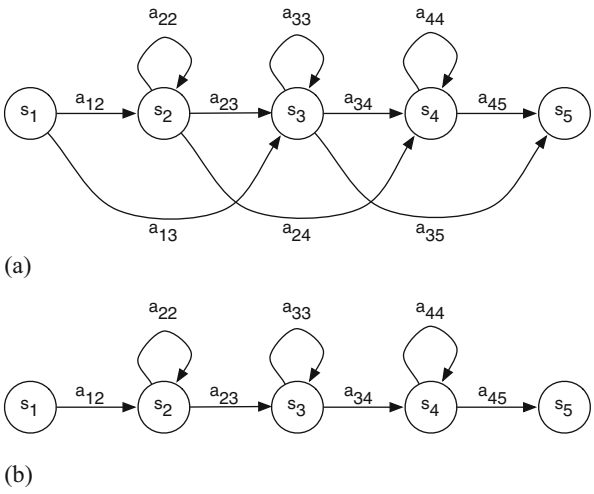


Fig. 7 Left to right HMM models are a popular choice for gesture recognition due to the temporal ordering of sub-gestures observed in a hand gesture. a_{ij} gives the probability of state transition from s_i to s_j. (**a**) Left-right model. (**b**) Left-right banded model

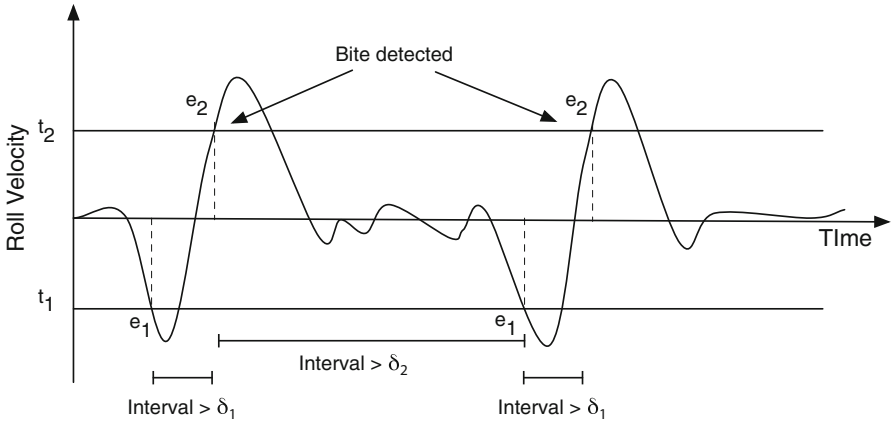


Fig. 8 BiteCounter [6] observes a sequential pattern of threshold-crossing events in the roll-velocity of a wrist to detect a food-intake gesture. The thresholds $t_1, t_2, \delta_1, \delta_2$ are learnt empirically

We note that unlike previously mentioned classification models like Random Forests and SVM where a single feature vector is extracted to represent a segment, HMM operates on a time-series of feature signals for a gesture segment. For example, the time series of instantaneous *pitch* and *roll* were used as the feature signals by Amft et al. [2, 3, 7].

Heuristic Methods In addition to classification-based methods, other heuristic-based approaches have also been proposed for gesture recognition. BiteCounter [6] is based on recognizing a sequential event pattern in a stream of roll-velocity signals obtained using a gyroscope (refer Fig. 8). Recall that the roll velocity is given by the

angular velocity measured around the wrist as its axis. The event pattern is based on the following observation while taking a food bite.

- The angular velocity about the wrist increases in magnitude and crosses a threshold (t_1) when the hand takes the food towards the mouth. The time of crossing the threshold marks the event e_1.
- After taking the food bite, the hand falls back and retraces its steps in the opposite direction i.e. the angular velocity about the wrist increases in magnitude but in the opposite direction and crosses a threshold (t_2). The time of crossing this threshold marks the event e_2.
- The time elapsed between events e_1 and e_2 during the food-intake gesture is greater than some threshold (δ_1). This threshold represents the minimum time needed to take a food bite.
- The time elapsed between events e_2 for a food-intake gesture and the event e_1 for the subsequent intake gesture is greater than some threshold (δ_2). This threshold is chosen because humans cannot have very rapid bites in succession as the chewing and swallowing of food takes time.

A food-intake gesture is detected upon observing the above mentioned pattern, and can be tracked without the need to buffer the sensor signals. Using the Android wear's frame of reference, the velocity around the wrist is given by the angular velocity measured around the x axis by the gyroscope. For event e_1, the velocity will become increasingly negative whereas for event e_2, the velocity will increase beyond a positive threshold. The values for various thresholds in this pattern is selected empirically from the training data such that it maximizes a score given as $\frac{4}{7}\times$ precision $+\frac{3}{7}\times$ recall. Note that similar characteristics can be observed for other hand-to-mouth gestures such as the puffing gesture while smoking a cigarette (See Fig. 3). However, the values $t_1, t_2, \delta_1, \delta_2$ characterizing a smoking gesture will differ from that of an eating gesture.

Activity Recognition

As we explained in the previous section, gesture classification is done for each segment independently and can yield noisy gesture label predictions. Thus, recognizing a high-level activity from the noisy labels can lead us to falsely detect an activity and give us an incorrect estimation of the activity duration and its boundaries. In this section, we give an overview of the approaches to perform joint classification of all the segments to recognize the activity. The activity recognized in this step can provide a feedback to the gesture classification module and correct some of the noisy gesture labels predicted earlier. In the following, we describe the key intuition behind some of the activity recognition approaches and give an overview of some of the state of the art approaches.

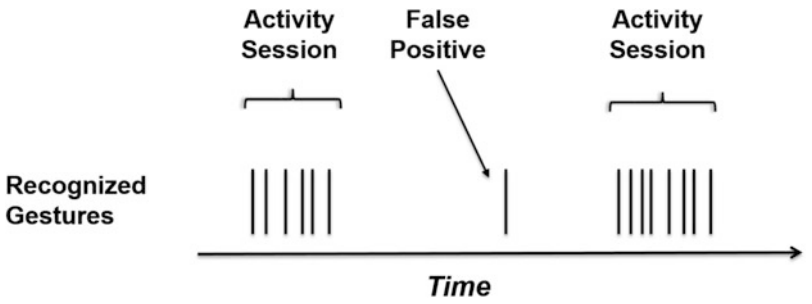


Fig. 9 In a typical session of gesture-driven activity like smoking, the characteristic gestures (e.g. taking a cigarette puff) form a temporal cluster i.e. the gestures are repeated at least a few times, and are found relatively close to each other in time. From the time-series of recognized gestures, we can extract these temporal clusters to identify an activity session. Any isolated recognized gesture that is not a member of any cluster can be discarded as a false positive

Temporal Cluster of Gestures in Activity Sessions

Gesture-driven activities like eating and smoking typically involve repeated gestures in each session (See Fig. 9). Thus, gestures that appear isolated in the time-series are unlikely to be a part of a true activity session. On the other hand, a gesture that is preceded by at least a few gestures of the same type in the vicinity is likely to be a part of an on-going activity. This observation forms the basis of approaches in the literature that cluster a group of detected gestures to identify an activity session and its boundaries.

DBSCAN Clustering Algorithm

Thomas et al. [16] use a clustering algorithm called *Density-based spatial clustering of applications with noise* (DBSCAN) for recognizing eating activity session from the noisy gesture labels. DBSCAN has three characteristics that make it useful for gesture-based activity recognition; (1) there is no need to specify the number of clusters ahead of time, (2) it is good for data that contains clusters of similar density, and (3) it is capable of identifying outliers (e.g. food intake gestures) in low-density regions. DBSCAN requires two parameters: the minimum number of points required to form a dense region (*minPts*), and a temporal distance measure given as a temporal neighborhood ε. Using these parameters, DBSCAN algorithm finds clusters that have at least *minPts* within the timespan covered by ε. Any detected gesture that does not belong to any of the clusters identified by the DBSCAN algorithm is considered to be a false positive, and ignored.

Rule-Based Activity Recognition

Saleheen et al. [13] use a rule-based approach to recognize the session boundaries for a smoking activity and to filter out isolated puffing gestures that are likely to be false positives. In their approach, a detected puff is considered an isolated puff if no other puff is within two standard deviations of the mean inter-puff duration (28 ± 18.6 s learnt from the training data across 61 subjects). After removing isolated puffs, they are left with clusters of (2 or more) puffs in the data stream. A simple rule-based method is proposed to declare a cluster of puffs as a smoking session, i.e., if it contains at least *mp* (minimum puff count) puffs. The appropriate value for *mp* is obtained by analyzing the recall rate for the true smoking sessions and the false smoking session detection rate. The best result was achieved when $mp = 4$. A similar set of rules can be learnt for other activities such as eating.

Temporal Consistency of Activity Sessions

One observation that can be used in determining activity boundary is that most activities have some temporal consistency [12] i.e. they last for a reasonably long time period. In other words, a person who is currently performing a certain activity is likely to continue with the same activity in the next time instant. This observation has been used to smooth out and correct intermittent misclassifications made at the lower classification layer. Unlike a standard activity recognition task, the modeling of temporal consistency in a gesture-driven activity is not straightforward as the output of the lower classification layer is a gesture label and not the activity label. Now, we look at two such approaches used specifically for the problem of gesture-driven activity recognition.

Conditional Random Fields

Parate et al. [11] use a graphical model-based approach that uses Conditional Random Fields (CRF) to jointly classify the sequence of segments (Fig. 10). This model introduces random variables (shown as top-level nodes in the figure) representing the type of activity for each segment obtained at the segmentation layer of the computational pipeline. The edges connecting these random variables are associated with pairwise factors that model the consistency of activity labels in the connected graph. The input to this model is a sequence of gesture labels generated at the gesture recognition layer. The gesture labels along with the classifier's confidence scores are given as input. The edge connecting the gesture label node with the activity label node is associated with a factor that models the consistency between a gesture label and the activity label. The CRF model outputs a smooth sequence of activity labels identifying the activity session and its boundaries. The activity labels are used to correct some of the false positives generated in the gesture

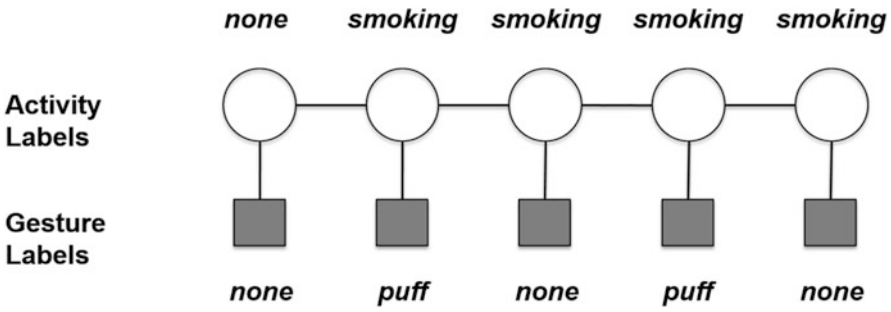


Fig. 10 Most human activities exhibit temporal consistency i.e. a person currently performing a certain activity is likely to continue with the same activity in the near future. In a gesture-driven activity, this means that the consecutive segments in a sequence are more likely to have the same rather than different activity labels while the gesture labels may change. Conditional Random Fields (CRF) is a model that takes into account this temporal consistency and outputs smooth and consistent activity labels based on the input sequence of gesture labels

classification stage. For example, if the CRF predicts that there is no smoking activity within a time-interval then any smoking gesture (i.e. taking a cigarette puff) recognized within this interval is a false positive and can be corrected.

Weighted Scores for Variable Length Windows

Tang et al. [15] propose a heuristic-based approach that predicts the state of smoking at the current time point, using the most recent history of three specific lengths: 1, 4 and 8 min. The longest history length is set to represent the average smoking session duration observed in the training data. For each history window, the puff frequency is calculated by counting the number of puffs detected. Then the score of smoking for the current window is estimated using the Gamma distribution of puff frequency. Lastly a weighted average of the scores of smoking for the three different window lengths is computed as the final score of smoking at the current time point. This approach models continuity by using a larger weight for the most recent 1 min period in time. The detector uses a threshold on the smoking score to output the current state.

HMM-Based Model for Gesture Recognition

Another approach for activity recognition is to use hidden markov models (HMMs). The intuition behind this method is that a gesture is always followed by one or more non-relevant gestures before the relevant gesture is observed again. Thus, we can construct a gesture spotting HMM by combining HMM models that recognizes relevant gestures, with another HMM called *garbage model* that recognizes all the

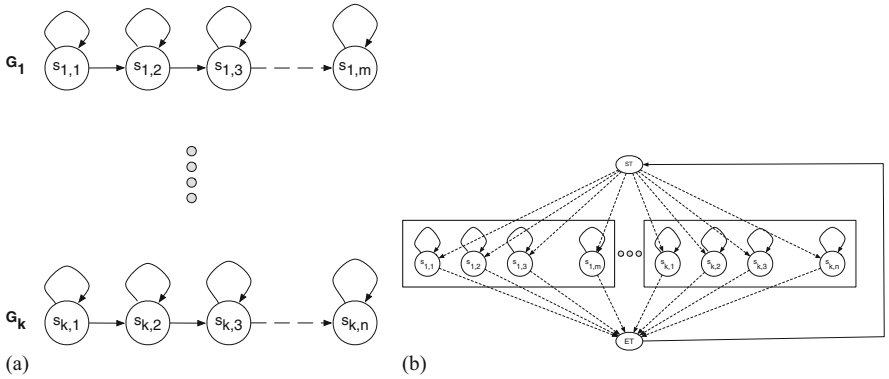


Fig. 11 A non-gesture model constructed using the gesture models. *ST* and *ET* are the start and the end dummy states respectively. (**a**) Left-right banded models for gestures G_1 to G_k. (**b**) General non-gesture model

possible non-relevant gestures, in a cyclical manner. However, training a garbage model is a difficult job since there are infinite number of meaningless or non-relevant gestures. Lee et al. [9] addresses this problem by introducing a new model called *threshold model* that consists of the state copies of all the trained gesture-specific models.

Let us understand this model using an example. A typical gesture model trained to recognize a specific type of gesture is usually modeled as a left-to-right model. In another words, a state in this model can transition next to itself or to the following states in the model. This is true for most gestures as there exist a temporal order in the sub-gestures within a gesture. Lee et al construct a non-gesture model (called *threshold model*) by collecting all the states of all the gesture-specific models. This model is constructed such that it is possible to transition from any state to any other state. Figure 11 shows an example of a non-gesture model constructed from all the gesture models. This model is a weak model for all the trained gestures and represents every possible pattern. The likelihood of a true gesture computed using this model is always smaller than the dedicated model for the given gesture.

Having constructed a non-gesture model, we now show how a model for gesture spotting is constructed. In a continuous human motion, gestures are sporadic with non-gestures in between. There is no specific order among different gestures. One way to define the alternating sequence of gestures and non-gestures is to construct a cascade connection of gesture models and a non-gesture model. A more effective structure is a circular interconnection of models: gesture models and then a non-gesture model which is then connected to the start of the gesture HMMs. An example of a construction of the gesture spotting model is shown in Fig. 12.

A potential weakness of using a threshold model is the spotting speed. The threshold model usually has a large number of states in proportion to the number of the gesture models in the system. Accordingly, the computational requirement

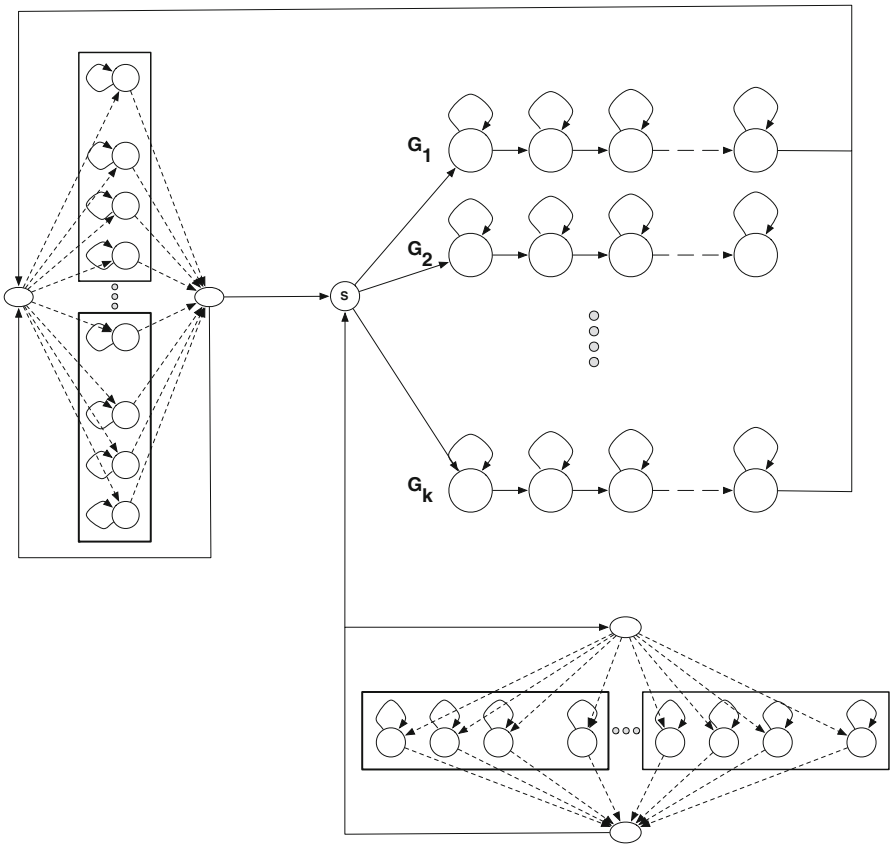


Fig. 12 A gesture spotting model

increases exponentially and the spotting speed slows down. This problem can be alleviated by reducing the number of states of the threshold model as described in [9].

Conclusions

This chapter provides an overview of a wide spectrum of approaches that have been proposed for behavior detection from wrist-worn inertial sensors. We discussed the challenges in segmenting the time-series stream from inertial sensors on the wrist to deal with the temporal dynamics in the gestural pattern, identifying key features from the inertial sensors, classifying the gestures into hand-to-mouth gestures of different types, and finally identifying repeated patterns of gestures

that are aggregated to robustly detect the behavior (e.g. eating or smoking session). We describe many recent efforts that address some of these challenges.

While existing work has answered several questions, many remain to be addressed in coming years. While much work has been done in gesture-based behavior recognition, substantial work that remains to be done in scaling these methods to the population and dealing with a wider range of confounders in the field. We also need to understand how these methods can execute efficiently on power-constrained devices such as fitness bands, to reduce the burden of frequently charging these devices. Finally, there are also many questions regarding usability to answer. One question in particular is whether the public is willing to wear fitness bands (or smartwatches) in their dominant hand to allow such gesture detection to work accurately. We expect that these questions will be answered in coming years as gesture-based behavior detection methods mature and are integrated into smartwatches and fitness bands in the same way that step detection is integrated into these devices.

References

1. Amft, O., Bannach, D., Pirkl, G., Kreil, M., Lukowicz, P.: Towards wearable sensing-based assessment of fluid intake. In: Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on, pp. 298–303. IEEE (2010)
2. Amft, O., Junker, H., Tröster, G.: Detection of eating and drinking arm gestures using inertial body-worn sensors. In: Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on, pp. 160–163. IEEE (2005)
3. Amft, O., Tröster, G.: Recognition of dietary activity events using on-body sensors. Artificial intelligence in medicine **42**(2), 121–136 (2008)
4. Burges, C.J.: A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery **2**(2), 121–167 (1998)
5. Criminisi, A., Shotton, J., Konukoglu, E.: Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. Found. Trends. Comput. Graph. Vis. **7**(2–3), 81–227 (2012). doi:10.1561/0600000035. URL http://dx.doi.org/10.1561/0600000035
6. Dong, Y., Hoover, A., Scisco, J., Muth, E.: A new method for measuring meal intake in humans via automated wrist motion tracking. Applied psychophysiology and biofeedback **37**(3), 205–215 (2012)
7. Junker, H., Amft, O., Lukowicz, P., Tröster, G.: Gesture spotting with body-worn inertial sensors to detect user activities. Pattern Recognition **41**(6), 2010–2024 (2008)
8. Keogh, E., Chu, S., Hart, D., Pazzani, M.: An online algorithm for segmenting time series. In: Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, pp. 289–296. IEEE (2001)
9. Lee, H.K., Kim, J.H.: An hmm-based threshold model approach for gesture recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on **21**(10), 961–973 (1999)
10. Madgwick, S.O., Harrison, A.J., Vaidyanathan, R.: Estimation of imu and marg orientation using a gradient descent algorithm. In: Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on, pp. 1–7. IEEE (2011)

11. Parate, A., Chiu, M.C., Chadowitz, C., Ganesan, D., Kalogerakis, E.: Risq: Recognizing smoking gestures with inertial sensors on a wristband. In: Proceedings of the 12th annual international conference on Mobile systems, applications, and services, pp. 149–161. ACM (2014)
12. Parate, A., Chiu, M.C., Ganesan, D., Marlin, B.M.: Leveraging graphical models to improve accuracy and reduce privacy risks of mobile sensing. In: Proceeding of the 11th annual international conference on Mobile systems, applications, and services, pp. 83–96. ACM (2013)
13. Saleheen, N., Ali, A.A., Hossain, S.M., Sarker, H., Chatterjee, S., Marlin, B., Ertin, E., al'Absi, M., Kumar, S.: puffmarker: a multi-sensor approach for pinpointing the timing of first lapse in smoking cessation. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 999–1010. ACM (2015)
14. Scholl, P.M., Van Laerhoven, K.: A feasibility study of wrist-worn accelerometer based detection of smoking habits. In: Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on, pp. 886–891. IEEE (2012)
15. Tang, Q., Vidrine, D.J., Crowder, E., Intille, S.S.: Automated detection of puffing and smoking with wrist accelerometers. In: Proceedings of the 8th International Conference on Pervasive Computing Technologies for Healthcare, pp. 80–87. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) (2014)
16. Thomaz, E., Essa, I., Abowd, G.D.: A practical approach for recognizing eating moments with wrist-mounted inertial sensing. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 1029–1040. ACM (2015)
17. Varkey, J.P., Pompili, D., Walls, T.A.: Human motion recognition using a wireless sensor-based wearable system. Personal and Ubiquitous Computing **16**(7), 897–910 (2012)