RisQ: Recognizing Smoking Gestures with Inertial Sensors on a Wristband

Abhinav Parate Meng-Chieh Chiu Chaniel Chadowitz Deepak Ganesan
Evangelos Kalogerakis

University of Massachusetts, Amherst
{aparate,joechiu,cheni,dganesan,kalo}@cs.umass.edu

ABSTRACT

Smoking-induced diseases are known to be the leading cause of death in the United States. In this work, we design *RisQ*, a mobile solution that leverages a wristband containing a 9-axis inertial measurement unit to capture changes in the orientation of a person's arm, and a machine learning pipeline that processes this data to accurately detect smoking gestures and sessions in real-time. Our key innovations are fourfold: a) an arm trajectory-based method that extracts candidate hand-to-mouth gestures, b) a set of trajectory-based features to distinguish smoking gestures from confounding gestures including eating and drinking, c) a probabilistic model that analyzes sequences of hand-to-mouth gestures and infers which gestures are part of individual smoking sessions, and d) a method that leverages multiple IMUs placed on a person's body together with 3D animation of a person's arm to reduce burden of self-reports for labeled data collection. Our experiments show that our gesture recognition algorithm can detect smoking gestures with high accuracy (95.7%), precision (91%) and recall (81%). We also report a user study that demonstrates that we can accurately detect the number of smoking sessions with very few false positives over the period of a day, and that we can reliably extract the beginning and end of smoking session periods.

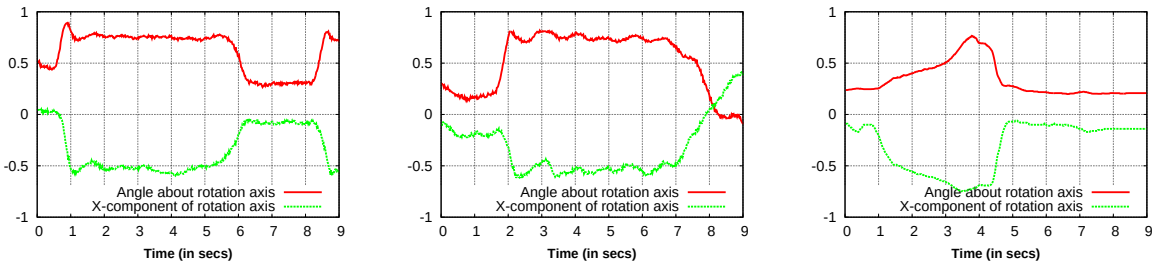
1. INTRODUCTION

Tobacco use remains the single largest preventable cause of death and disease in the United States and worldwide. According to CDC estimates, cigarette smoking kills more than 440,000 Americans each year, either through cancer, heart disease, stroke, or lung diseases. It also increases the chances of other serious illnesses, such as diabetes [2]. In addition, smoking-related illness in the United States costs $96 billion in medical costs and $97 billion in lost productivity each year. The numbers are more alarming worldwide, where tobacco use is increasing rapidly in low- and middle-income countries. In fact, it is estimated that there are about a billion smokers worldwide, with more than 80% in the low and middle-income countries. Of these, about 6 million die through smoking-related causes each year.

At the heart of addressing this scourge is early detection and timely treatment. Several smoking cessation programs have been developed that show that intervening at opportune moments can help a person quit smoking. In theory, continuous sensing using the mobile phone and wearables has the potential to enable such informed and timely interventions both by observing the evolution of a patient's smoking pattern over time, as well as measuring contextual factors that influence smoking (environment, social interactions, stress, etc). However, the first step towards such methods is the ability to detect smoking events in real-world settings, a goal that has proven elusive.

We argue that there is a dire need for a simple and reliable smoking detector that has high sensitivity and specificity, and is easy to wear on a day-to-day basis. Existing methods fall short on one or more of these axes. A sensitive detector for smoking is a tomography meter (e.g. CReSS monitor [1]). However, the user needs to attach the tomography device to the cigarette prior to smoking. This is both burdensome and can change the underlying smoking behavior. A recent alternative is to use the deep respiration cycles associated with inhalation and exhalation of smoke, which can be detected by a respiration chest band (mPuff [4] and Sazonov et al [16]). Unfortunately chest bands are cumbersome to wear for long periods and therefore have limited appeal beyond clinical trials. Other wearables that have been previously proposed include RF-based proximity sensors worn on the collar and wrist to detect when the hand is in the vicinity of the mouth, but this is not robust to confounders [17]. One promising approach is to embed a sensor in a cigarette lighter which detects whenever the lighter is lit [18]. Unfortunately this does not provide information about individual puffs, which is particularly useful for determining the degree of nicotine intake across subjects [11].

Our work relies on a wristband embedded with a single, low-power 9-axis inertial measurement unit (IMU) that fuses information from an accelerometer, gyroscope, and compass to provide 3D orientation of the wrist. IMUs are easy to integrate with wrist-worn wearables, many of which already have accelerometers embedded for calorie or activity tracking. However, there are many challenges in robustly recognizing smoking gestures from orientation data.



(a) Smoking when standing still (b) Smoking when walking (c) Eating with a spoon

Figure 1: Illustration of IMU signals for various hand-to-mouth gestures. (a) & (b) Smoking gestures are characterized by a quick change in orientation when taking a cigarette towards the mouth and a long dwell time while taking a "smoking puff". (c) In contrast, eating gestures are characterized by a slow change when taking a spoonful of food towards the mouth and a very short dwell time.

1.1 Challenges

The central challenge that we face is that we need to detect and recognize a smoking gesture from a plethora of other gestures that a user performs each day. While recognition of intentional gestures from inertial sensors is commonplace in gaming devices (e.g. Nintendo Wii), there are several challenges in achieving our goal in a natural setting.

Signal variation due to orientation changes: The first challenge is that the signal from the inertial sensor varies significantly depending on the user's body orientation. Thus, if the user changes his/her body orientation while smoking, the orientation output of the sensor will change. An example is shown in Figure 2, where a 180° change in the user's body orientation completely changes the signal. However, we still need to detect the beginning and end of a smoking gesture despite the unknown user's body orientation. Note that this problem is not present in gesture recognition systems designed for user interaction, such as the Nintendo Wii. This is because a user typically faces a fixed direction during the interaction. In addition, the user can consciously adjust the gesture so that it is recognized by the system.

Detecting smoking gestures: A second challenge is that a smoking gesture needs to be detected without any explicit information given by the user regarding the beginning and end of a smoking session. Devices such as the Nintendo Wii address this problem by having the user press a button, thereby explicitly providing information about the beginning and end of the gesture. Other gesture recognition systems such as the Bite-Counter [8] use a similar method: they assume that the user presses a button prior to an eating session. In contrast, we wish to avoid any user interaction altogether, and instead design a passive gesture tracking system. Thus, we aim at designing methods that distinguish a smoking gesture from a huge number of different gestures that a user can perform using his/her hands.

Concurrent activity while smoking: A third challenge is that the user might smoke while performing a concurrent activity. Concurrent activities can modify the characteristic patterns of smoking gestures and further complicate our problem. For example, Figure 1 shows how a smoking gesture looks when the user is stationary (Figure 1a) against when the user is walking (Figure 1b). We can see that smoking gestures have a few bumps when the user walks, in addition to the distortions at the beginning and end of the gesture where the user brings the cigarette down and continues walking by swinging his/her hand. Similarly, the user might be conversing with a friend while smoking, or might perform other activities while sitting or standing. All such concurrent activities also complicate the smoking gesture recognition problem.

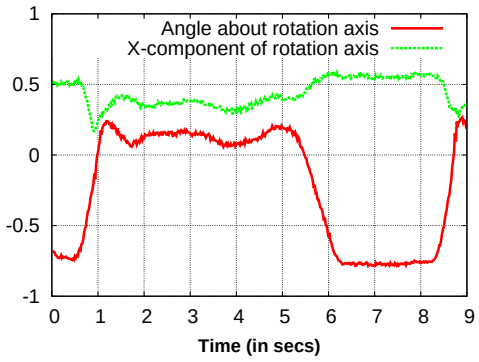


Figure 2: IMU signals for the smoking gesture in Figure 1(a) when the user changes facing direction by 180°. The characteristic patterns observed in IMU signals change with the user's orientation.

Confounding gestures: A fourth challenge is that many other gestures have arm movement patterns similar to smoking. For example, a user can perform several isolated gestures each day that involve raising the hand towards the mouth and bringing it back down similarly to smoking gestures. In particular, eating and drinking activities not only have similar hand-to-mouth gestures, but also involve repeated sequences of these gestures, as in smoking. Figure 1 shows an example of an eating gesture. Visually, there are differences in the pattern of eating compared to smoking. However,

designing an algorithm that robustly distinguishes smoking from other confounding gestures across different users is particularly challenging.

Labeling smoking puffs: In order to recognize smoking gestures robustly, we design a method based on *supervised* classification. However, to train such a classifier, we need training data in the form of fine-grained labels for the beginning and end of each gesture. Many existing approaches for obtaining such labeled data are not feasible or have drawbacks: a) requesting self-reports from the users is impractical, since users cannot possibly label each smoking puff manually while smoking: such an approach would force the users to change their gestures from smoking to interacting with a device to create these self-reports, b) video capture of the smoking session for post-facto annotation is restrictive, since it requires the user to always face the camera, and c) having an observer annotate each smoking puff is cumbersome and not scalable. Thus, one particular challenge is how to enable fine-grained labeled data collection while limiting the burden on participants.

1.2 Contributions

The key innovation of our work is the ability to recognize sequences of hand gestures that correspond to smoking sessions "in the wild". This involves a sensing pipeline with several important aspects: a) we detect candidate hand-to-mouth gestures by continuously tracking and segmenting the 3D trajectory of a user's hand, b) we extract discriminative trajectory-based features that can distinguish a smoking gesture from a variety of other confounding gestures like eating and drinking, c) we design a probabilistic model based on a random forest classifier and a Conditional Random Field that analyze sequences of hand-to-mouth gestures based on their extracted features, and accurately outputs the beginning and end of smoking sessions. Finally, (d) in order to train the classifier and Conditional Random Field, we also propose a simple method to gather training labeled data "in the wild": we ask subjects to wear two IMUs, one on the elbow and one on the wrist, which allows us to build 3D animations of arm movements that can be easily labeled by a third party without compromising the subject's privacy. The only limited burden on the subject is to specify coarse time windows where smoking occurred to verify that we detect gestures within the correct periods of time. Each module in this whole pipeline addresses challenges specific to our problem domain, and we present several new ideas and algorithms.

Our results show that we can detect smoking gestures with 95.7% accuracy and 91% precision. In addition, we can detect smoking session time boundaries reliably: the error in the estimated duration of a smoking session is less than a minute. Finally, we demonstrate with a user study that we can accurately detect the number of users' smoking sessions in the period of a day with only few false positives (less than two in our study). In all, we think that RisQ is a very promising approach for use in smoking cessation and intervention.

2. BACKGROUND

Inertial Measurement Unit: The Inertial Measurement Unit (IMU) is an electronic device consisting of 3-axis accelerometer, 3-axis gyroscope and 3-axis magnetometer. IMU has an on-board processor that fuses the output of these three sensors to compute the orientation of the device in a 3D world (absolute) coordinate system defined by y-axis pointing towards the magnetic north, z-axis pointing perpendicular to the ground in the direction opposite to the earth's gravity and x-axis pointing to the geographical east. We use the Invensense MPU-9150 IMU for our experimental study. This sensor outputs the 3D orientation of a device in the form of a *quaternion*.

Quaternions: Quaternions are convenient mathematical entities for representing orientations and rotations of objects in three dimensions. Quaternions have found a great applicability in robotics, computer graphics, and computer vision communities due to the ease in calculations for performing rotations of objects in 3D space.

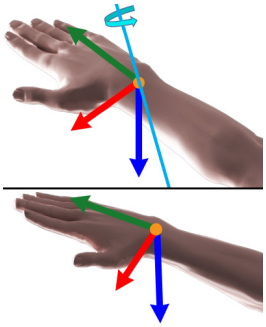


Figure 3: Top: 3D arm and its initial reference frame. A quaternion rotates the arm about a rotation axis (shown in cyan). Bottom: rotated arm and its updated reference frame.

Formally, a *quaternion* is defined using a scalar component q_s and a 3D vector (q_x, q_y, q_z). We write a quaternion $\mathbf{q}$ as follows:

$$\mathbf{q} = q_s + q_x\hat{i} + q_y\hat{j} + q_z\hat{k}$$

where $\hat{i}, \hat{j}$ and $\hat{k}$ are imaginary basis elements, each of which squares to -1. A quaternion $\mathbf{q}$ is said to be a *unit quaternion* if its magnitude $|\mathbf{q}|$ given by $\sqrt{(q_s^2 + q_x^2 + q_y^2 + q_z^2)}$ is equal to one. 3D rotations can be compactly represented with unit quaternions. Given a 3D rotation defined through a unit vector $\langle x, y, z\rangle$ representing the axis of rotation, and an angle θ representing the amount of rotation about the axis, the corresponding quaternion representing this rotation is defined as:

$$\mathbf{q} = \cos\tfrac{\theta}{2} + x\sin\tfrac{\theta}{2}\hat{i} + y\sin\tfrac{\theta}{2}\hat{j} + z\sin\tfrac{\theta}{2}\hat{k}$$

It can be shown that a point $\mathbf{p}$ in 3D space with coordinates $\{p_x, p_y, p_z\}$ can be rotated using a quaternion with the following formula:

$$\mathbf{p}' = \mathbf{q}\cdot\mathbf{p}\cdot\mathbf{q}^{-1}$$

where $\mathbf{p}$ is written as $p_x\hat{i} + p_y\hat{j} + p_z\hat{k}$, and $\mathbf{q}^{-1}$ is the so-called conjugate quaternion of $\mathbf{q}$, which is expressed as:

$$\mathbf{q} = \cos\tfrac{\theta}{2} - x\sin\tfrac{\theta}{2}\hat{i} - y\sin\tfrac{\theta}{2}\hat{j} - z\sin\tfrac{\theta}{2}\hat{k}$$

It is straightforward to see that a quaternion not only represents a 3D rotation, but can also represent the 3D orientation of an object. Given an initial orientation of a 3D object de-

fined by its initial frame of reference, the quaternion q rotates the object yielding a new orientation (see Figure 3).

3. SYSTEM OVERVIEW

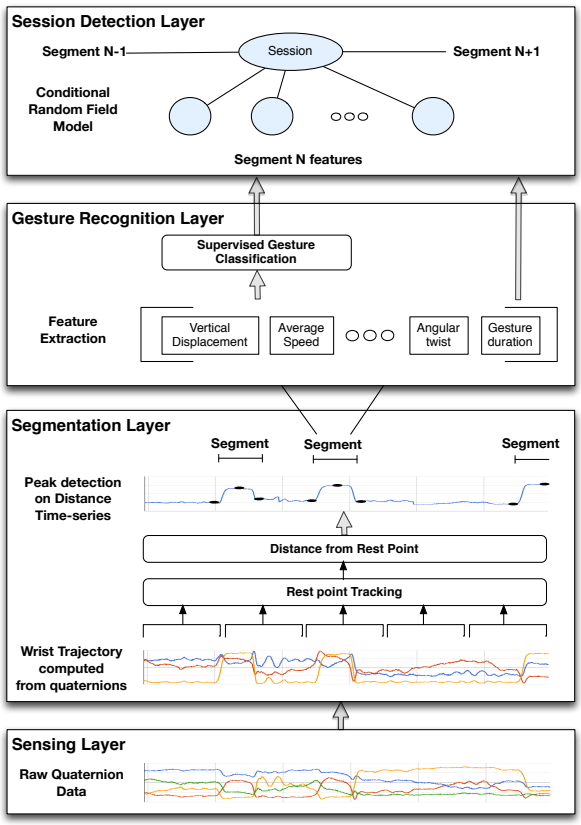


Figure 4: RisQ: Quaternion data processing pipeline

In this section, we provide an overview of the RisQ computational pipeline that recognizes smoking gestures and sessions. We also overview the training data collection methodology we used to obtain fine-grained labeled data for the purpose of training our supervised classification method.

Computational pipeline: Figure 4 gives an overview of the computational pipeline that we use for detecting smoking gestures and sessions. At the lowest layer of the pipeline is the extraction of quaternion data from the single wrist-worn 9-axis IMU.

The second layer in the pipeline is the segmentation layer that extracts segments containing candidate gestures from the raw sensor data and filters out extraneous data. The intuition behind the segmentation process is that, while performing a hand gesture, humans start from "a rest position" in which the arm is relaxed, then move their arm, and finally the arm falls back to another, possibly different rest position. Thus, the gestures tend to lie in segments between these resting positions. The segmentation layer accomplishes the segment extraction by computing the spatio-temporal trajectory taken by the wrist using quaternion data and tracking rest positions. Based on the computed trajectory, at each time step it computes the distance of the wrist from rest positions, and identifies segments using a peak-valley detection algorithm. The output of this layer are the extracted segments containing candidate hand gestures.

In the third layer of our pipeline, our method computes a feature vector for each segment, consisting of features that can discriminate hand-to-mouth gestures corresponding to smoking from a large number of other hand-to-mouth gesture candidates. We then use a supervised classifier, called Random Forests [5, 7], that outputs the probability for the type of gesture ("smoking", "eating" or "other" gesture) based on the extracted features for each segment individually.

The top-most layer in the processing pipeline is a session detection layer that identifies whole sessions ("smoking", "eating", or "other") based on the Random Forest classifier outputs and the segment features. The key intuition behind this layer is that each smoking session involves a continuous sequence of smoking gestures, and is unlikely to contain gestures from other activities. We use a probabilistic model based on a Conditional Random Field (CRF) [13] that captures probabilistic dependencies between the type of gestures contained in consecutive segments. In particular, it has the advantage of filtering out spurious classification outputs of the lower layer and robustly detecting the beginning and end of smoking sessions.

While the focus of this paper is on recognizing smoking gestures and sessions, our approach can be tailored to other instances of repetitive gestures. In §6.3, we use eating sessions to train our method, and we find that it performs considerably better than state-of-art eating detectors that use inertial sensors.

Labeled data collection: We use a novel data collection methodology to obtain fine-grained gesture labeled data from participants for training our method. Our data collection framework consists of two tools: (i) a logging application and (ii) a 3D visualization and labeling tool. The logging application collects data from on-body IMU sensors placed on the user's wrist and upper arm. Since our goal is to reduce the burden of labeling for the user and avoid interfering with his gestures, our logging application provides a simple user-interface to log events on his mobile phone. This helps us identify time windows that require fine-grained labeling of the smoking or eating gestures. The 3D visualization and labeling tool produces 3D hand animations from the logged sensor data and provides an interface for fine-grained labeling of the gestures, such as "smoking a puff" or "taking a bite of food". The type of gestures can be easily labeled by other users in these animated sequences. Unlike video recording, our visualization tool allows users to see and label gestures from any viewing angle they prefer.

4. GESTURE DETECTION & RECOGNITION

In this section, we describe how we detect and label hand-to-mouth gestures when the user wears the single IMU on his wristband. The first step in our data processing pipeline

is to partition the quaternion time-series obtained from the IMU into segments such that each segment contains a gesture (§4.1). The second step is to extract some features that can be used to discriminate different types of gestures (§4.2). The last step is to recognize the detected gestures and extract smoking sessions (§4.3).

4.1 Gesture detection

In order to detect a gesture, it is essential that we segment the time series of quaternions into segments, such that each segment contains a complete gesture. The problem of correctly segmenting sequences of quaternions presents a number of challenges. First, the segmentation should extract segments representing the entire gesture duration. Otherwise characteristic features that are useful in classifying a gesture will be lost. Moreover, the extracted segments should not be much longer than the gesture, otherwise the computed features related to the gesture will be inaccurate. Another fundamental challenge is that the characteristic patterns observed in a time series of quaternions are dependent on the orientation of the person's body performing the gesture. For example, the pattern observed for a gesture when the person is lying on a bed is very different from the pattern observed when he is standing. The problem gets further complicated due to the large number of variations in the gestures observed for the same activity, within and across users, making the variability of characteristic patterns extremely high. An additional challenge comes from the huge number of all possible gestures and possibly confounding gestures performed by humans throughout their daily life.

The intuition behind our segmentation approach is the observation that the humans tend to perform a gesture starting from an initial rest body pose, and then ending the gesture in another, possibly different, rest pose. This observation holds true for the hand-to-mouth gestures we intend to identify. For example, a person may start a smoking gesture by having his hand lie on an armrest of a chair, then move his hands towards his mouth, then back to the armrest of the chair or the top of a table. Alternatively, the arm might end up being straight, hanging from the body. In general, the beginning and end rest poses are not necessarily stationary and are not necessarily identical. In the following sections, we describe our segmentation approach based on continuous tracking of the rest positions of the fore-arm and computing the spatio-temporal trajectory of the wrist in 3D space.

Relative trajectory computation: Our first step towards identifying hand-to-mouth gestures is computing the spatio-temporal trajectory of the hand from the stream of quaternion data. The main challenge here is that trajectory is affected by body motion and orientation. For example, a smoking gesture when a person is walking takes a much longer path in 3D space compared to the same gesture performed when the person stands at one place. Similarly, a smoking gesture could be affected by any changes to the pose of the user's body while smoking.

The key insight in our approach is that we care less about the absolute trajectory of the hand in 3D space. We care more about the trajectory of the hand relative to a body joint, such as the shoulder joint. Such a relative trajectory would not depend on the orientation of the person's body, and every smoking gesture should result in roughly isomorphic trajectories with similar shape characteristics.

While such a relative trajectory would be easy to compute if the individual also had an IMU close to the shoulder, the fact that we only have a single wrist-worn IMU complicates the problem. To address this, we make a simplifying assumption that the position of the elbow remains stationary (the shoulder joint does not rotate) while the gesture is being performed. Specifically we compute the trajectory of the wrist relative to a stationary elbow, assuming that the wrist is in a fixed, unit distance away from it. We find that the above assumptions have little influence in our trajectory computation, since hand-to-mouth gestures primarily involve rotating the elbow and not moving the upper arm.

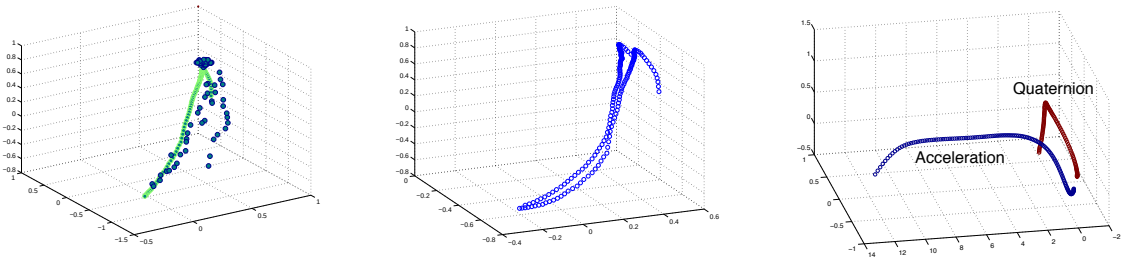
More formally, given a time sequence of quaternions ($\mathbf{q}_1$, $\mathbf{q}_2$, $\ldots$, $\mathbf{q}_n$), we can compute the trajectory taken by the wrist by computing the position of the wrist for each time step t ($t = 1...n$). The position of the wrist $\mathbf{w}$ at time t in the elbow's frame of reference $\mathcal{F}$ can be computed as follows:

$$\mathbf{w} = \mathbf{q}_t \cdot \mathbf{w}_0 \cdot \mathbf{q}_t^{-1} \tag{1}$$

where $\mathbf{w}_0 = 0\hat{i} + 1\hat{j} + 0\hat{k}$ (i.e., has coordinates $\begin{bmatrix}0 & 1 & 0\end{bmatrix}$) represents the position of the wrist in the device's local coordinates based on the assumption that the length of the fore-arm of person has unit length. This assumption also works in our favor as the computed trajectories are independent of the particular variations of the arm length across the population.

Due to this approximating nature of the computed trajectory and the continuous shifts in the elbow position, the wrist trajectory does not always appear smooth. Thus, we smooth the trajectory by fitting a curve using cubic B-splines. Figure 5(a) shows a sample trajectory for a smoking puff gesture before and after the smoothing operation.

IMU vs Accelerometer: One question that is worth asking at this point is whether we really need a 9-axis IMU that provides quaternion data (by fusing information from the accelerometer, gyroscope, and compass) to compute trajectories, or whether we can do the same with a single accelerometer. Figure 5(b) shows two smoothed sample trajectories taken for two consecutive cigarette puffs while standing still. These trajectories are computed relative to the elbow and align well as the gestures are isomorphic. In Figure 5(c), we show the accelerometer-based and quaternion-based trajectories for the smoking gesture when the user is walking. We see that the accelerometer-based trajectory in this case appears longer. This is because it is relative to the starting point in world coordinates. On the other hand, the trajectory computed using quaternions is relative to the elbow has similar characteristics to trajectories observed when standing still.



(a) Trajectory smoothing using curve-fitting

(b) Trajectories for two consecutive smoking puff gestures

(c) Quaternion and Acceleration-based trajectories for the same gesture when the user is walking

Figure 5: Figure (a) shows a noisy trajectory for a smoking puff gesture and the smooth trajectory (green curve) obtained after curve-fitting. Figure (b) shows trajectories for two consecutive puffs while standing still. These two trajectories are well-aligned for the isomorphic gestures. Figure (c) shows the trajectory for a smoking puff gesture when the user is walking, computed from acceleration data (blue curve) and quaternion data (brown curve). The acceleration-based trajectory is significantly elongated and distorted when the user is walking, while holding the cigarette in his mouth. The quaternion-based trajectory instead is computed relatively to the elbow and is free of such motion artifacts.

Segment Extraction: Our segmentation procedure is based on the observation that humans tend to keep their hands in rest positions and any gesture starts from one rest position and terminates at another rest position. The key challenge is determining these rest positions in a continuous stream of wrist movement. The rest positions need not be the same across gestures (e.g. rest position while sitting and smoking would be different from standing and smoking). Even the beginning rest position for a single smoking puff might be different from the final rest position, since there can be many positions where the human arm is "relaxed".

Our segment extraction method involves four stages:

- First, our segment extraction method continuously tracks the rest position from which a current hand gesture started. To detect the rest position, the trajectory is traced according to short sliding time windows (e.g. 10 seconds). Within each window, positions where the wrist velocity is very low are computed. If one or more candidates exist, then we compute the centroid of these low velocity points as the "rest point". Thus, we can now annotate each point in each trajectory trace with the most recent point corresponding to the rest position for which the hand gesture started.
- Second, we compute the spatial distance of each point in the trajectory from the most recent rest point. For any hand-to-mouth gesture, we expect that the distance from the rest point should increase rapidly, plateau for a short period, and then return to the next rest point.
- Third, we use a peak detector to detect peaks and troughs in the distance time series obtained from the previous stage, and look for a tell-tale trough-peak-trough pattern that is typical of a hand-to-mouth gesture. To filter false positives from either tiny hand movements or very long gestures, we use a relatively loose threshold for distance separation between trough and peak, as well as the expected duration of a typical smoking puff. At this point, the segment extraction pipeline provides the trough-peak-trough segment to the next level of the classification pipeline.

4.2 Segment Feature Extraction

In order to recognize the type of gesture in each segment, we need to compute features that can discriminate different types of gestures. The feature extraction procedure is based on the observation that a hand-to-mouth gesture can be divided into three stages: i) an "ascending" stage that corresponds to the trough to peak movement when the hand goes towards the mouth, ii) a "stationary" stage that corresponds to the period where the hand stays near the peak (close to mouth), and iii) a "descending" stage where the hand goes from peak to trough i.e. back to the rest position. These three sub-segments are useful for extracting features as we describe below.

Table 1 shows the set of features computed for each segment. The following paragraphs describe the features we compute from the smoothed spatio-temporal trajectory and the quaternion information:

- **Duration features.** These features measure the time duration of the segment containing the gesture and the durations of the three stages within the segment: ascending, descending and stationary.
- **Velocity features.** Second, we use the computed positions of the wrist at each time step to compute the instantaneous velocity of the wrist for the ascending and the descending stages. We compute the average speed, the maximum speed and the variance in speed for both ascending and the descending stages giving us six features.
- **Displacement features.** These features measure the displacement of the wrist during the ascending and the

Feature Set		
Duration Features		
Duration	4	Total gesture duration, and duration for the ascending, descending, and the stationary stage.
Velocity Features		
Speed	6	Mean, max and variance of the wrist speed. We compute these for the ascending and the descending stages.
Displacement Features		
distZ	2	Vertical displacement of the wrist during the ascending and descending stage.
distXY	2	Horizontal displacement of the wrist during the ascending and descending stage. This is computed on a plane parallel to the ground.
dist	2	Net displacement between the rest position and the peak position. This is computed for the ascending and the descending stage.
Angle Features		
roll velocity	4	Median and maximum angular velocity about the axis of the arm. We compute 2 features each for the ascending and the descending stage.
roll	2	Net angular change about the axis of the arm during the ascending and the descending stage.
pitch	12	Angle between the arm and the direction of the gravity (pitch angle). We compute the pitch angle at the peak during the ascending and descending stage. Also, we obtain 9 decile features from the pitch pitch distribution for the ascending stage. We compute the median pitch angle for the descending stage.

Table 1: Feature set extracted for a segment containing a gesture. The table describes each feature type and gives the count of features used for each type.

descending stages of the wrist motion. We measure the vertical displacement, displacement on the plane parallel to the ground and the net displacement at the end of each stage as features.

- **Angle features.** From the orientation information, we can compute the *roll* component of the rotation about the axis of the arm. This is computed by converting the quaternions into Tait-Bryan angles. We call the rate of change in the angle of the rotation along the axis of the arm "roll velocity". We compute the median and the maximum of instantaneous roll velocities and use them as features. Also, we compute the net roll angle as the feature. We extract these 3 features for the ascending and the descending stage. Next, we compute "pitch" that measures the angle between the vector along the arm and the direction of the gravity. We use pitch angle at the peak point of the ascending and the descending stages as features. Next, using the distribution of pitch angles observed at each time step for the ascending stage, we compute 9 values that give the $10^{th}, 20^{th}, ..., 100^{th}$ percentile values as the features. For the descending stage, we use median pitch as a feature.

4.3 Gesture recognition

After extracting segments that contain individual gestures, the next step in our pipeline is to recognize each gesture contained in each segment. The input to this step is a set of segments with their features and the output is a set of labels representing the type of gesture (e.g. "smoking", "eating", or "other" depending on available labels).

Labeling segments poses a number of challenges. First, there is no simple mapping between the duration, velocity, displacement and angular features and the labels. For example, if the duration of the gesture is more than a particular threshold, we cannot infer that the gesture is a "smoking puff" with total certainty. It might be the case that the user instead drinks slowly from a cup. Therefore, we have to also consider other features: for example, smoking is likely to also be strongly correlated with certain ranges of arm twisting angles, wrist velocities, horizontal and vertical displacements and so on. Manually constructing a mathematical formula that detects all smoking gestures based on features is very hard. Noise in the trajectory features make things worse, thus we need to adopt a probabilistic method that outputs the probability of assigning each label to each segment. In the following paragraph, we describe a probabilistic classifier for recognizing each individual gesture. Unfortunately, classifying each gesture independently from all the others in a sequence of segments does not seem to work very well. For example, a sequence of gestures might be labeled as $\{smoking, smoking, eating, smoking, smoking\}$. However, it is very unlikely that the user interrupts and intermixes gestures in a small period of time. Such noisy labelings can easily occur due to feature noise and large deviation of wrist trajectories from the ones in our training data. Thus, we employ a probabilistic model that takes as input the probabilistic output of the classifier for each individual segment, and additionally examines how likely is for successive segments to have the same or different label. Based on this probabilistic model, we estimate the most likely joint labeling of all segments together. We describe this probabilistic model later in this section.

Individual classification of gestures: To build a mapping from segment features to segment labels, we observe that the type of gestures is likely to be correlated with certain value ranges of segment features extracted in the previous section. For this reason, we use a classifier that has a form of a decision tree. A decision tree contains a set of nodes and at each node, the value of an individual feature is compared to a threshold. Then depending on whether the feature of the segment has a smaller or larger value than the threshold, the

classifier examines the left or right child node, which in turn examines the value of another feature. When a leaf node is reached, the decision tree outputs a probability based on how many segments in our training data had the same set of splits while traversing the tree.

The thresholds and feature choices at each node of the tree are automatically constructed from the training data during an offline stage. The best choice of features and thresholds are selected to maximize the confidence of predictions [7].

Unfortunately, fitting a single decision tree in the training data results in poor performance. The reason is that a single set of splits yields predictions that are suited only for segments whose features are very similar to the ones of the training segments. A popular alternative is to build an ensemble of decision trees: 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 [7]. This ensemble of decision trees is called random forest classifier. The output of the random classifier is the probability of assigning a label to each segment, computed as the average of the probabilities outputted by the individual decision trees.

Joint classification of gestures: As we explained in the beginning of the section, classifying each segment independently from all the others in a sequence of segments yields noisy label predictions. Instead, we classify all segments jointly by using the output of the random forest and also considering that consecutive segments in a sequence are more likely to have the same rather than different label.

Given a sequence of $k = 1...K$ segments, we introduce a random variable C_k for each segment representing the type of gesture contained in the segment. We express the joint probability of assigning a sequence of labels to all the random variables $\mathbf{C} = \{C_1, C_2, ..., C_K\}$ of the segments given their features $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K\}$ as follows:

$$P(\mathbf{C}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_k \phi(C_k, \mathbf{x}_k) \prod_{k,k+1} \psi(C_k, C_{k+1})$$

The unary factors $\phi(C_k, X_k)$ assess the consistency of each individual segment to each label based on the random forest classifier. The pairwise factors $\psi(C_k, C_{k+1})$ assess the consistency of each pair of consecutive segments to their labels. The denominator $Z(\mathbf{x})$ serves as a normalization constant for each input sequence to ensure that the sum of probabilities over all possible label assignments is equal to 1. The type of this model is known as Conditional Random Field [13] in the literature of machine learning.

In our implementation, the unary factors have the following form:

$$\phi(C_k = l, \mathbf{x}_k) = \exp(\theta_l f_l(\mathbf{x}_k) + \theta_{l_0})$$

where $f_l(\mathbf{x}_k)$ are the probabilistic outputs of the random forest classifier for each label $l = \{smoking, eating, other\}$. The parameters $\{\theta_l, \theta_{l,0}\}$ re-scale the probabilities of the random forest classifier, and are learned from the training data. The $\exp$ function ensures that the final result will lie in the interval $[0, 1]$ (i.e., will also be a probability) after applying the normalization constant, as typically done in logistic regression and log-linear CRF models [19].

The pairwise factors evaluate how likely is to assign pairs of labels in consecutive segments and are expressed as:

$$\psi(C_k = l, C_{k+1} = l') = \exp(\theta_{l,l'})$$

The parameters $\theta_{l,l'}$ are also learned from the training data. The learned parameters turn out to favor the same label for consecutive segments, as expected.

Parameter learning: It is extremely hard to hand-tune the parameters $\boldsymbol{\theta} = \{\theta_l, \theta_{l,0}, \theta_{l,l'}\}$ in the above probabilistic model. Instead, the model parameters are learned such that they maximize the likelihood of the training sequences. Specifically, the parameters are estimated to maximize the following objective function [12]:

$$L(\{\lambda_k\}) = \sum_{n=1}^{N} \log(P(\mathbf{C}[n] \mid \mathbf{x}[n])) - \mu||\boldsymbol{\theta}||^2$$

where the first term expresses the log likelihood of the training sequences given their features. The second term is a regularization term that penalizes large values in the parameters. Such large values would favor over-fitting of the parameters to the training dataset, and are less likely to generalize to new data. The parameter μ is selected through cross-validation.

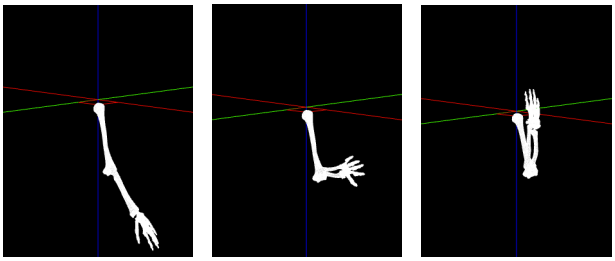
The objective function is maximized using the L-BFGS method [14] which is a popular optimization method for solving unconstrained optimization problems. The objective function is convex function, thus, a unique maximum exists [12].

Inference: Given a sequence of segments, we estimate the most likely joint assignment of labels to the segments based on the above model. From the assigned labels, we can simply extract the different gesture sessions by tracking when the label changes in the sequence of segments.

Inference is performed with the technique known as max-product belief propagation [12], which yields the most likely sequence labeling in this type of model. The complexity of the evaluating the unary and pairwise terms, as well as executing belief propagation is linear in the number of segments, thus the labelings can be predicted very efficiently.

5. LABELED DATA COLLECTION

In this section, we describe the framework used to capture ground truth data in uncontrolled environment settings with a low labeling overhead for the user, and the dataset that we obtained through this approach. The main idea is to have the subject wear two inertial sensors, one at the wrist, and one at the elbow, and segment the data post-facto through a 3D visualization tool. We note that the use of two sensors is used only for gathering training data. During actual usage of RisQ (i.e., during testing), the user always wear a single IMU on a wristband, which is more practical, user-friendly, and less costly.



(a) (b) (c)

Figure 6: Smoking gesture visualization using sensor data obtained from the two IMUs placed on the upper arm and the wrist.

5.1 Labeling Framework

Sensors: We use Invensense MPU-9150 IMU to obtain orientation data sampled at 50Hz. We fit a user with a wristband and an armband, each containing an IMU device and use a mobile app to log the sensor data on the phone. Our goal is to reconstruct the hand gestures performed by a user using the logged data.

Constructing hand gesture: A human arm consists of an upper arm and a forearm. The upper arm is connected to the shoulder, and is connected with the forearm at the elbow. The forearm extends from the elbow to the wrist. For constructing the hand motion, we assume that the orientation is uniform for the entire forearm and the entire upper arm. This assumption is simplistic as the orientation of the forearm close to the elbow can be different from the orientation of the wrist but this assumption enables us to reconstruct a fairly accurate hand motion with fewer sensors.

In order to reconstruct the hand motion, we use a digital model of a 3D human skeleton in a polygon mesh format (Figure 6). The 3D model consists of skeletal limbs that that can be oriented independently by applying the orientation information in the form of quaternions to the shoulder and elbow joints of the skeleton. The orientations given by the quaternions are applied to the skeleton hierarchically as in skeleton-based animation methods commonly used in computer graphics applications. Given that we log the sensor data at 50Hz, we render the 3D model at 50 frames per seconds where we draw a new pose of the 3D model every frame to enable real-time playback. Rendering is implemented in Processing 2, a popular library used in computer graphics.

Labeling: Using the 3D animation and visualization method described above, we can now mark the start and the end of a gesture like a cigarette puff. However, manually visualizing and marking the gesture for hours of data can be a time-consuming exercise, and in the absence of any additional information, it is difficult to distinguish smoking or eating from any confounding gestures that may look similar. To address this, we ask the participating users in the data collection to mark the beginning and the end of the smoking and eating periods using a mobile app. For fine-grained gesture labeling, we limited our efforts to the marked periods.

5.2 Gesture Dataset

Our dataset consists of IMU data collected for over 32 hours from 15 volunteers for over 20 smoking sessions, 15 eating sessions, 6 drinking sessions, and a variety of other gestures. Users typically wear sensors for several hours, therefore, our dataset contains many potential confounding gestures. Of this data, we had to discard a few smoking and eating sessions due to interruptions and extreme noise in the data. This left us with 28 hours of data, containing 17 smoking sessions and 10 eating sessions. Overall these sessions include 369 smoking puffs and 252 food bites. The data included users performing several concurrent activities including smoking while standing alone, smoking in a group while having a conversation, smoking using a hand-rolled cigarette and smoking while walking around.

We used our 3D visualization and labeling tool to hand-label the start and end of each smoking puff within the smoking periods reported by the participants. There are a few instances when defining a precise start or a precise end of a gesture is not possible. Since our focus is on detecting a smoking puff, any interval that begins at least 1 second prior to the puff and ends after at least 1 second of the puff is valid for training our algorithm.

6. EVALUATION

Our evaluation has three parts. We start with evaluating how well we detect individual smoking gestures (taking smoking "puffs") and whole smoking sessions. Then, we compare our results against prior work that detects wrist gestures and smoking behavior. Finally, we include a user study for which we implemented the full computational pipeline on a mobile phone, and demonstrate the effectiveness of RisQ for recognizing users' smoking sessions in real-time.

6.1 Recognizing smoking sessions

The ultimate goal of our work is to detect smoking sessions accurately so that we can track: a) how many times a user smoked each day, b) the length of each smoking session, and c) how many puffs they took during each session.

Smoking session detection: First, we look at how many smoking sessions were identified correctly by RisQ. We ran a leave-one-out cross-validation experiment where we leave a single smoking session for testing, and use rest of the data for training. We repeat this procedure for each smoking session in our dataset described in §5. At each time, we evaluate the output of max-product belief propagation in our CRF.

RisQ detected 15 complete sessions out of the 17 sessions in our dataset— it missed 2 smoking sessions for two users who had gestures largely different from other users.

RisQ is also able determine the duration of smoking sessions. Table 2 shows the results of this experiment. The average ground-truth session duration in this set up was 326.2

seconds. We observe an average error of 65.7 seconds in our session duration estimate for the 17 smoking sessions in our leave-one-out cross-validation experiment. We note that when we exclude the two undetected sessions, the average error is reduced to 26.22 seconds. The errors are caused due to the approximate estimates of the beginning and end timestamps of each session.

Statistic	Avg ± Std. Dev.
Duration of smoking sessions	326.21±19.65 s
Error in estimation	65.7±30.6 s

Table 2: Errors in session duration estimates using our CRF.

6.2 Recognizing smoking gestures

We examine now how well we can recognize individual hand-to-mouth smoking gestures that correspond to each smoking puff in a session. Since the total number of smoking gestures is large in our dataset (369 "smoking puffs", 252 "food bites" and 4976 other gestures), we perform a 10-fold cross-validation experiment here: we split the gestures into 10 groups (folds), and of the 10 folds, a single fold is retained for testing, and the remaining 9 are used for training. Then we repeat for each test fold.

First, we evaluate how well individual smoking gestures can be detected by the Random Forest (RF) classifier which examines each segment independently of the others. Then we evaluate the performance of the CRF that infers the type of gesture for each segment by considering the probabilistic dependencies of the segment labels in a sequence.

Gesture detection performance: Table 3 shows the performance metrics for 10-fold cross-validation for smoking puff gesture detection using i) the RF classifier , and ii) CRF. We see that the RF classifier gives 93% accuracy in gesture recognition, and a moderate precision value of 0.72 in detecting smoking gestures. The CRF significantly improves the performance by reducing the number of false positives by 75.83%. This results in a much higher precision value of 0.91 in detecting smoking gestures, and overall 95.74% accuracy in overall gesture recognition. The CRF only causes a slight decrease in recall from 0.85 to 0.81.

Optimizing performance: We now examine how we can optimize the precision and recall for smoking gesture detection by tuning our method. While training the RF classifier, we use a cost function that adds a penalty for missing a training smoking gesture. When this penalty is increased, it increases recall for the smoking gestures, although it also increases the number of false positives. Figure 7 shows the change in precision and recall obtained using RF versus CRF as this penalty increases. We can make two observations here: first, a false positive rate as small as 0.05 results in a significant drop in precision (0.6) of the RF classifier. This is because of the large number of non-smoking hand gestures. Second, the CRF cleans up the mis-classifications of the RF classifier to improve precision. Figure 7 shows that this improvement can be as high as 0.53. However, this improvement comes at a cost of drop in recall, as CRF smooths out some of the true gestures. We can see that the best performance is achieved for a false positive rate of 0.023.

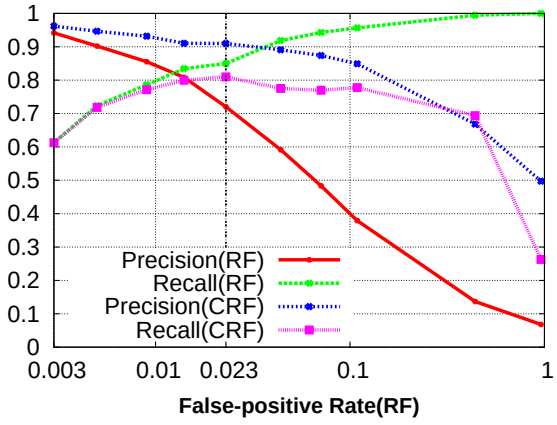


Figure 7: Precision & Recall versus False Positive rate, while adjusting the cost function of the Random Forest (RF) classifier during training. A small increase in false positive rate reduces precision of the RF classifier but the CRF improves the precision dramatically.

Generalization across subjects: We now examine whether our method can successfully detect gestures for a user by using training data only from other users. In other words, we examine how well our method *generalizes* to new users' data not included during training. For this experiment, we test the smoking gesture detection for each of 14 smokers, given training data from the other 13 in our dataset. Figure 8 shows precision and recall in this "leave-one-user-out" scenario using the Random Forest classifier and the CRF. The precision and recall of our method are high on average which indicates that our method is able to generalize to new users. In particular, the CRF improves precision, but it may occasionally filter out correctly detected puffs. This happens if the user's gestures and their temporal distribution is largely different from what is observed in the training. This suggests the it is better to acquire training data from a variety of users.

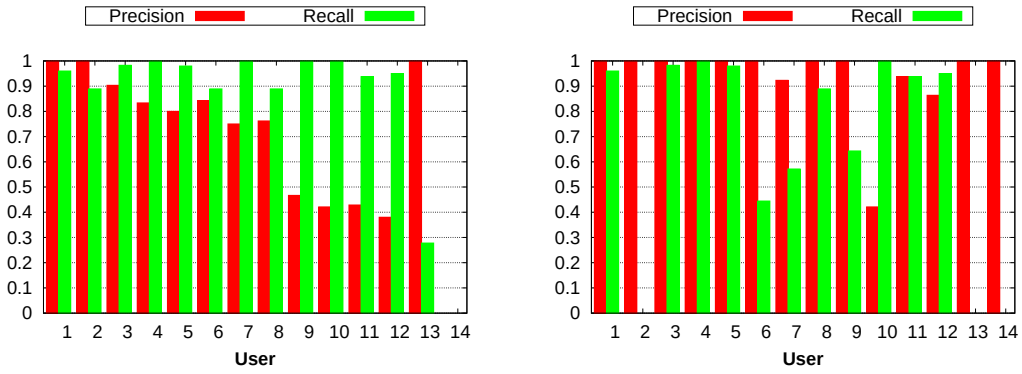
6.3 Comparison #1: Bite Counter

Our first comparison examines a different IMU-based gesture recognition technique, and the benefits of using our computational pipeline.

Bite-Counter [8] is a state-of-art method to detect eating bites while having a meal using angular velocity from a gyroscope worn on the wrist. The intuition is that the angular velocity about the axis along the wrist increases above an empirically observed threshold (T_1) while taking a bite and drops below another threshold (T_2) after taking the bite. Also, there is a minimum time-lapsed (T_3) between these two threshold-crossing events while eating food, and a thresh-

	Performance Metrics			
	Accuracy	Recall	Precision	False-positive rate
Random Forests	93.00%	0.85	0.72	0.023
CRF	95.74%	0.81	0.91	0.005

Table 3: Performance metrics for smoking puff detection obtained using 10-fold cross-validation evaluation with i) Random Forests classifier (RF), ii) Conditional Random Fields (CRF) with Random Forests. CRF reduces 75.83% of the number of false positives generated by RF, thus improving precision.



(a) Random Forests (b) CRF

Figure 8: "Leave-one-user-out" evaluation results. The CRF improves precision over the RF classifier. Occasionally the CRF smooths out puffs correctly detected by the RF classifier resulting in a slightly lower recall.

	Eating Sessions		All data	
	Recall	Precision	Recall	Precision
Bite-Counter	0.60	0.57	0.65	0.03
RF	0.92	0.78	0.69	0.64
CRF	N/A	N/A	0.64	0.78

Table 4: Eating gesture detection using Bite-counter [8] for cases: i) within eating sessions when it is known that user is having a meal, and ii) when session information is not available.

	In-Sessions		All data	
	Recall	Precision	Recall	Precision
Bite-Counter	0.60	0.71	0.73	0.05
RF	0.97	0.95	0.85	0.72
CRF	N/A	N/A	0.81	0.91

Table 5: Smoking gesture detection using Bite-counter [8] for cases: i) within smoking sessions when it is known that user is smoking, and ii) when session information is not available.

old on the time interval between two consecutive bites (T_4). Thus, this algorithm requires searching for the values of these four parameters: $\{T_1, T_2, T_3, T_4\}$ that have the best score. The score is given by $\frac{4}{7}\times$ precision$+\frac{3}{7}\times$recall. One caveat is that the Bite-Counter algorithm knows when an eating session is in progress since the user presses a button, whereas RisQ requires no such information.

Eating gesture detection: For a fair comparison, we first look exclusively at eating sessions in our dataset, and train RisQ to detect eating gestures using the same feature set and approach that we used for smoking. For this purpose, we labeled the bite gestures in 10 of the eating sessions in our dataset. Table 4 shows the results for two cases — one where we examine eating gesture detection within known eating sessions (case for which Bite-Counter is optimized), and one where we look across the entire data that includes all types of gestures. For evaluation limited to eating sessions, we only use the Random Forest classifier, and excluded CRF as we do not need to detect sessions. Even when only looking within the eating sessions, we see that RisQ has substantially higher precision and recall (21% improvement in precision and 32% improvement in recall). We suspect that our benefits are primarily due to the use of more robust trajectory-based features rather than just the wrist rotation, and the use of the Random Forest classifier as opposed to simple thresholds. As might be expected, when the eating sessions are not known a priori, the performance of Bite-Counter drops substantially. Bite-Counter still has good recall, but the precision is extremely low (0.03) i.e., the algorithm is generating a huge number of mis-classifications since it is picking up a substantial fraction of cases where the wrist is rotated, and detecting them as a bite.

We must note here that RisQ was not designed to detect eating behavior in the first place. Thus, it is possible that with more careful exploration of features, the detection of eating gestures can improve. However, the fact that it

performs better even without such tuning demonstrates the broader potential of our approach for gestures other than smoking.

Smoking gesture detection: Just as RisQ can be re-trained to detect eating gestures, Bite-Counter can be re-trained to detect smoking gestures by looking for wrist rotations that are representative of smoking. We use the methodology in [8] to learn appropriate parameters for Bite-Counter if it were used for detection smoking gestures. Table 5 shows the comparative results. The results clearly show that RisQ perform much better than Bite-counter.

6.4 Comparison #2: mPuff

We now turn to an alternate approach for detecting smoking gestures using a respiration chest band: mPuff [4]. Although the results are on different datasets, mPuff reports precision of 91%, true positive rate of 83% and false positive rate of 8%. Our results are similar for precision, and better for other metrics.

To directly compare these methods, we collected several hours of data from the Zephyr Bioharness Chestband for respiration data and the wrist-worn IMUs. Our dataset was captured in the wild, and has several smoking sessions in addition to other activities such as drinking, driving, walking, working on a computer, and so on.

We used the approach suggested in mPuff, which suggests a list of features and the use of an SVM classifier for training the parameters. However, there was substantial noise in the breathing dataset due to other concurrent activities (walking, driving, etc), as a result of which many breathing waveforms had to be discarded prior to training the classifier. The results of training using the clean breathing waveforms were unsuccessful, and we got low precision/recall numbers using mPuff. The results using RisQ instead were similar to those shown earlier.

While further exploration is needed for a quantitative comparison, we highlight that dealing with noise and confounders is a challenge no matter what sensor modality is used for detecting behaviors such as smoking. The fact that RisQ works well despite the confounders is one of its major strengths.

6.5 User Study in the Wild

Our final study looks at the entire computational pipeline learnt using the dataset described in §5, and executed in real time for new users.

Android-based Implementation: Our Android-based implementation for the user study comprises of a wearable wristband with an IMU and bluetooth that continuously streams quaternions at a rate of 50 Hz to an Android phone. The RisQ computational pipeline for real-time smoking session detection is executed on the phone. Our application generates notifications when it detects that a smoking session is in progress and prompts the user to confirm if the detection is correct. Apart from smoking detection, our application also acts as a logging app to record data from the IMU worn by a user and to record ground truth events like the beginning and the end of a smoking session.

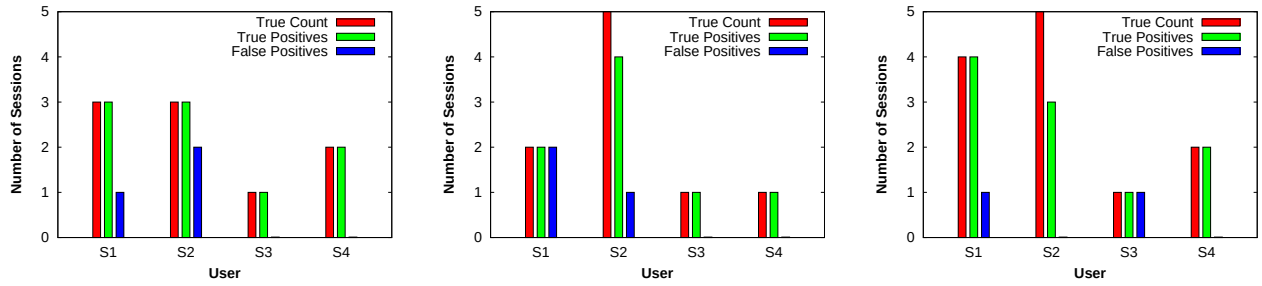
Figure 10: Smoking monitor mobile app and an IMU-equipped wristband.

The complete UI of the mobile application and an IMU equipped wristband are shown in Figure 10. The main interface of the app has a set of UI buttons that can be used by a user to report smoking events as well as confounding events like drinking and eating. A user can use this to report the session boundaries using *before, during* and *after the session* events. In addition, the UI has a tab to view the events previously reported by the user. A user can use this interface to correct the reported event times or to delete a report if there was an error in submitting reports.

The complete gesture recognition pipeline has been implemented in Java. This pipeline detects smoking gestures every 20 seconds in the the buffered sensor data. The classification is done using a Random Forest classification model, learnt using the Weka data mining software [10]. The classification outputs are then buffered for a minute and used in forming a CRF model to detect a smoking session in progress. The inference using CRF is done every minute, thus there is an average latency of 30 seconds in detecting the start of a smoking session.

User study: We recruited 4 users for this study. The users wore the IMU-fitted wristband for four hours per day for three days. We asked these users to always carry the phone with our mobile application installed. To record the ground truth information, we ask users to report the rough start and the end time for their smoking sessions using our app. Also, our app prompted these users whenever it detected smoking and the users could respond by confirming yes or no. Our goal was to check how well our algorithm would work if it were integrated into a real-world intervention algorithm that a therapist might use, such as a phone call or text message.

How well can RisQ deliver real-time interventions?: Figure 9 shows the actual number of sessions, number of sessions detected by the mobile app correctly and the number



(a) Day 1 (b) Day 2 (c) Day 3

Figure 9: User study results for 4 subjects for 3 days. Figures show the following information for each subject and each day: i) ground-truth count of the smoking sessions, ii) number of sessions detected correctly i.e. true positives, and iii) number of falsely detected sessions i.e. false positives. The results show that we rarely missed a session and had only a few false detections (0-2) per day.

of sessions falsely detected for the four users. We see that the number of sessions varied between 1 to 5 in a day for the period that the mobile app was running. Of all the 30 reported smoking sessions, we missed only 3 sessions. The number of false session detections was low and less than or equal to 2 per day. In our informal exit interviews with the users, they reported no discomfort from the small number of false alarms that was raised by RisQ. In summary, the user study provides strong evidence of the effectiveness of our technique in recognizing smoking sessions across users in real-world scenarios.

Statistic	Value
Time for segmentation	92.34±6.85 ms (52-134ms)
Time for feature extraction	79.88±5.30 ms (9-227ms)
Time for CRF Inference	5.89±2.50 ms (1-23ms)
Memory	12-20 MB
Binary Size	1.7 MB

Table 6: System overhead measured on a Galaxy Nexus phone

Operating Voltage	2.4-3.6 V
Operating Current	4.25 mA
Streaming Data rate	1150 bytes/s

Table 7: Specifications for Invensense MPU-9150 IMU

RisQ Implementation benchmarks: We finalized our evaluation with a brief benchmark of our real-time implementation of RisQ. We first demonstrate that our mobile app has a low system overhead. Table 6 shows the summary of the execution overhead of the smoking detection pipeline measured on Samsung Galaxy Nexus device running Android 4.3 OS. The execution of the three stages in the pipeline: segment extraction, feature vector extraction and session detection using the CRF take an average time of 92.34 ms, 79.88 ms and 6 ms respectively. These overheads are incurred every 20 seconds for segmentation and feature extraction and once in a minute for the CRF inference. The memory requirement to execute our app is modest and varies between 12-20 MB. Users running our application for the user study did not find our app to impact the performance of their phones in their daily usage.

Table 7 gives the performance metrics for the operation of Invensense MPU-9150 IMU that we used in our user study.

7. RELATED WORK

In this section, we describe the two areas of related work — gesture recognition in computer vision as well as gesture recognition and motion capture using wearable inertial sensors.

Computer Vision: This line of research deal with the problem of gesture detection and recognition using a camera. The camera is used to track a person and the motion of the arm. This gives the coordinates of various points on the body that can be used as features for gesture recognition. Yang et al. [23] used Hidden Markov Models (HMM) for segmenting and identifying the set of gestures for human-robot interaction. Elmezain et al. [9] proposed a method to use Conditional Random Fields to separate gestures from non-gestures without needing to train for non-gestures. Wang et al. [22] combined HMMs with CRFs in a Hidden Conditional Random Field that can be used as a single gesture class detector or a multi-way gesture classifier. All these methods employ similar machine learning formulations, however, recognizing arms accurately in images or video is prone to errors due to occlusions, clothing, illumination changes and other errors in pose tracking from 2D images. These computer vision methods also assume that the user is always expected to turn towards the camera. Yin et al. [24] use HMMs for gesture recognition but it requires a use of special glove to enable the tracking of the arm in a cluttered background.

Wearable sensors: The advances in wearable sensors technology has generated a lot of interest in gesture recognition and other novel uses of these sensors. Vlasic et al. [21] pro-

posed a full body motion capture system using inertial sensors. Such systems are useful to capture a person's full body motion, but require several such sensors, acoustic sensors, careful calibration and initial body pose estimates. Agrawal et al. [3] proposed *PhonePoint Pen* to enable users to use a mobile phone as a pen to write in the air using an accelerometer in the phone. This work identifies six basic strokes in the English characters that are essentially straight lines and two half-circles. These strokes are identified using correlation with the ideal strokes. *u*Wave [15] uses a dynamic time warping (DTW) technique to recognize gestures from the accelerometer present in Nintendo Wii. DTW is used to match a temporal sequence with a given template, when the number of templates is small. Chen et al. [6] use a 6-axis inertial sensor to track interactive gestures and rely on *push-to-gesture* mechanism to spot the gesture segment. All these approaches detect "control" gestures that have little variations across users. Varkey et al. [20] train two SVMs: first to detect a high-level activity in a window of inertial sensor data and another to detect micro-activities within a window. They evaluated their approach for smoking detection using a synthetic data provided by non-smokers who imitated the smoking gestures. It has not been evaluated for data in the wild. Dong et al. [8] proposed Bite Counter to detect food bites in an eating session using a gyroscope. We compared this with our approach and found it to be less useful for smoking detection.

8. CONCLUSION

Mobile phones can play a major role in detecting and preventing harmful user behavior, such as smoking. In this paper, we tackle the problem of recognizing smoking behavior using a wristband equipped with a single 9-axis inertial sensor and a mobile phone application. Our results demonstrate that we can accurately detect smoking gestures in the wild. While we focus on smoking gesture recognition in this paper, we also have preliminary experiments that indicate that our pipeline can be used for detecting other behaviors such as eating. This could be also useful for preventing harmful eating patterns that can lead to obesity. In the future, we plan to improve our trajectory extraction procedure, enrich our method with more trajectory features, and investigate more complex probabilistic models for recognizing different types of users' gestures and behavioral patterns.

Acknowledgements

This paper has benefitted from extensive discussions with several people. In particular, we would like to thank Annamalai Natarajan who helped us implement an efficient CRF training algorithm. We would also like to thank Rui Wang who has been instrumental in building the prototype of 3D visualization tool.

9. REFERENCES

[1] Cress. http://borgwaldt.hauni.com/en/instruments/smoking-machines/smoking-topography-devices/cress-pocket.html.
[2] Cdc fact sheet on smoking. http://www.cdc.gov/tobacco/data_statistics/fact_sheets/fast_facts/.
[3] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter. Using mobile phones to write in air. In *MobiSys*, 2011.
[4] A. A. Ali, S. M. Hossain, K. Hovsepian, M. M. Rahman, K. Plarre, and S. Kumar. mpuff: Automated detection of cigarette smoking puffs from respiration measurements. In *IPSN*, 2012.
[5] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
[6] M. Chen, G. AlRegib, and B.-H. Juang. Feature processing and modeling for 6d motion gesture recognition. *Multimedia, IEEE Trans. on*, 15(3):561–571, 2013.
[7] A. Criminisi, J. Shotton, and E. Konukoglu. 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, Feb. 2012.
[8] Y. Dong, A. Hoover, J. Scisco, and E. Muth. A new method for measuring meal intake in humans via automated wrist motion tracking. *Applied psychophysiology and biofeedback*, 37(3):205–215, 2012.
[9] M. Elmezain, A. Al-Hamadi, S. Sadek, and B. Michaelis. Robust methods for hand gesture spotting and recognition using hidden markov models and conditional random fields. In *IEEE ISSPIT*, 2010.
[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. In *SIGKDD Explorations*, pages 10–18, 2009.
[11] D. Hammond, G. T. Fong, K. M. Cummings, and A. Hyland. Smoking topography, brand switching, and nicotine delivery: results from an in vivo study. *Cancer Epidemiology Biomarkers & Prevention*, 14(6):1370–1375, 2005.
[12] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.
[13] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.
[14] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Math. Program.*, 45(3):503–528, Dec. 1989.
[15] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. In *IEEE PerCom 2009*, 2009.
[16] P. Lopez-Meyer, S. Tiffany, and E. Sazonov. Identification of cigarette smoke inhalations from wearable sensor data using a support vector machine classifier. In *IEEE Conf. on EMBC*, 2012.
[17] E. Sazonov, K. Metcalfe, P. Lopez-Meyer, and S. Tiffany. Rf hand gesture sensor for monitoring of cigarette smoking. In *Sensing Technology (ICST), 2011 Fifth International Conference on*, 2011.
[18] P. M. Scholl, N. Kücükyildiz, and K. V. Laerhoven. When do you light a fire?: Capturing tobacco use with situated, wearable sensors. In *UbiComp Adjunct*, 2013.
[19] C. Sutton and A. McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012.
[20] J. P. Varkey, D. Pompili, and T. A. Walls. Human motion recognition using a wireless sensor-based wearable system. *Personal Ubiquitous Comput.*, 16(7):897–910, Oct. 2012.
[21] D. Vlasic, R. Adelsberger, G. Vannucci, J. Barnwell, M. Gross, W. Matusik, and J. Popović. Practical motion capture in everyday surroundings. *ACM Trans. Graph.*, 26(3), July 2007.
[22] S. B. Wang, A. Quattoni, L.-P. Morency, and D. Demirdjian. Hidden conditional random fields for gesture recognition. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, pages 1521–1527, 2006.
[23] H.-D. Yang, A.-Y. Park, and S.-W. Lee. Gesture spotting and recognition for human-robot interaction. *Robotics, IEEE Transactions on*, 23(2):256–270, 2007.
[24] Y. Yin and R. Davis. Toward natural interaction in the real world: Real-time gesture recognition. In *ICMI-MLMI*, 2010.