March 2019

# MACHINE LEARNING METHODS FOR PERSONALIZED HEALTH MONITORING USING WEARABLE SENSORS

Annamalai Natarajan
*University of Massachusetts Amherst*

# MACHINE LEARNING METHODS FOR PERSONALIZED HEALTH MONITORING USING WEARABLE SENSORS

A Dissertation Presented

by

ANNAMALAI NATARAJAN

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2019

College of Information and Computer Sciences

# MACHINE LEARNING METHODS FOR PERSONALIZED HEALTH MONITORING USING WEARABLE SENSORS

A Dissertation Presented

by

ANNAMALAI NATARAJAN

Approved as to style and content by:

_____

Deepak Ganesan, Co-chair

_____

Benjamin M. Marlin, Co-chair

_____

John Staudenmayer, Member

_____

Justin Domke, Member

_____

Andrew Campbell, Member

_____

James Allan, Chair of the Department
College of Information and Computer Sciences

# DEDICATION

*To my parents.*

*I have reached this point because of your love, support, sacrifices and constant encouragement.*

# ACKNOWLEDGMENTS

I have many people to thank who have helped me reach this point.

I would like to start by thanking my two advisors: Deepak Ganesan and Ben Marlin. Deepak has been instrumental in helping me think about interesting problems and novel solutions. In the past six years, he has been very approachable and has offered advice both in and outside of graduate school. I would like to thank Ben Marlin for being patient with me while honing my research skills and for taking the time to painstakingly review every document I have churned out in graduate school. I consider myself to be lucky to have two advisors who provided me a great graduate school experience.

I would like to thank professors Justin Domke, John Staudenmayer and Andrew Campbell for graciously agreeing to serve to on my dissertation committee and providing valuable feedback. This whole dissertation came about with the novel cocaine use detection study which would not have been possible without the team at Yale. I would like to especially thank Robert Malison, Gustavo Angarita and Edward Gaiser. Many fruitful discussions with our collaborators at Yale led to a better understanding of cocaine addition and its effects on human physiology. I would like to thank Abhinav Parate for his help with setting up infrastructure to make data collection possible and all the staff at the Connecticut Mental Health Center who helped with the cocaine study.

I would also like to thank fellow graduate students in both the Machine Learning for Data Science and the Sensors labs. I would like to mention a few by name: Steve Li, Roy Adams, Tao Sun, Garrett Bernstein, Kevin Winner, Conrad Holtsclaw, Aaron Schein, Jeffrey Varghese, Juston Moore, Jeremy Gummeson, Abhinav Parate, Addison Mayberry, Ali Kiaghadi, Mohammad Rostami, Soha Rostaminia, Erik Risinger, Ari Kobren, Amee

Trevedi, Luis Pineda, Kyle Wray and Sandya Saisubramaniyan. I wanted to extend special thanks to Leeanne Leclerc, Michele Roberts, Karren Sacco, Laurie Connors and Eileen Hammel for taking care of administrative stuff. I would also like to thank the staff at CSCF and front office for their help with myriad tasks.

Many thanks to funding agencies for providing various forms of assistanships throughout graduate school. I would like to thank the National Science Foundation (CNS-0910900, CNS-0855128, IIS-1722792, IIS-1350522), the National Institute on Drug Abuse (K24 DA017899, R01 DA033733, P20 DA027844), the Yale Center for Clinical Investigation and the National Center for Advancing Translational Science (UL1 TR000142), the National Institutes of Health (1U54EB020404-01), CTSA (UL1 RR024139) from the National Center for Research Resources and the National Center for Advancing Translational Science, the Mobile Health to Knowledge (MD2K) grant and, the President's Science and Technology Fund, University of Massachusetts, Amherst.

I would like to thank my support system who helped me to sustain graduate school. I would like to thank my parents for accommodating my frequent calls and infrequent visits throughout graduate school. I would like to thank my uncle, older sister and grandparents for their patience and understanding. I would like to thank my wife's parents for selflessly spending two brutal New England winters to care for our infant son. I would like to thank Jim and Gerry for providing emotional and childcare support during our stay in the Amherst area. Big thanks to other friends and families in the CS department, North village, Morning star preschool and the Common school. Lastly, I relied on two people to confide, lament and, celebrate every little event in graduate school. Thank you Selvi and Karthik for your love and many sacrifices.

# ABSTRACT

## MACHINE LEARNING METHODS FOR PERSONALIZED HEALTH MONITORING USING WEARABLE SENSORS

FEBRUARY 2019

ANNAMALAI NATARAJAN

B.Tech., UNIVERSITY OF MADRAS

M.Sc., COLORADO STATE UNIVERSITY, FORT COLLINS

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Deepak Ganesan and Professor Benjamin M. Marlin

Mobile health is an emerging field that allows for real-time monitoring of individuals between routine clinical visits. Among others it makes it possible to remotely gather health signals, track disease progression and provide just-in-time interventions. Consumer grade wearable sensors can remotely gather health signals and other time series data. While wearable sensors can be readily deployed on individuals, there are significant challenges in converting raw sensor data into actionable insights. In this dissertation, we develop machine learning methods and models for personalized health monitoring using wearables. Specifically, we address three challenges that arise in these settings. First, data gathered from wearable sensors is noisy making it challenging to extract relevant but nuanced features. We develop probabilistic graphical models to effectively encode domain knowledge when extracting features from noisy wearable sensor data. Second, prediction models developed on one population in lab settings may not generalize to other populations in field

settings. We develop domain adaptation techniques to improve lab-to-field generalizability. Third, collecting ground truth labels for health monitoring applications is expensive and burdensome. We develop active learning methods to minimize the effort involved in collecting ground truth labels. We evaluate these methods and models on two case studies: cocaine use detection and human activity recognition.

# TABLE OF CONTENTS

# LIST OF TABLES

**Table** **Page**

# LIST OF FIGURES

**Figure** **Page**

# CHAPTER 1

# INTRODUCTION

## 1.1 Overview

According to the centers for disease control and prevention, key risk factors including high blood pressure, tobacco use, alcohol use, inadequate physical inactivity, unhealthy diets and abnormal sleep patterns play a key role in many chronic diseases[1]. Hence, there is a need to continuously monitor at risk individuals for their health status and activities over extended periods of time in their natural settings with the goal of improving their health and well being.

Wearable devices make it possible to continuously and remotely monitor individuals in their natural settings. Wearable devices are devices worn on, in or around the body. With reduced form factors, longer battery life and enhanced networking capabilities, wearable devices make it possible to monitor individuals over extended periods of time. Additionally, most consumer grade wearable devices come equipped with an accelerometer, gyroscope and/or magnetometer, making it possible to detect motion, orientation and direction. When wearable sensors are coupled with smartphones and cloud computing, large volumes of data can be remotely analyzed to find interesting patterns, detect abnormalities and detect target activities from continuous streams of sensor data. The resulting insights may benefits both the individual on whom the sensors are deployed as well as the individual's healthcare providers. Among other applications, wearable sensors make possible proactive healthcare monitoring (*e.g., tracking diseases before symptoms otherwise manifest*),

[1]CDC's National Center for Chronic Disease Prevention and Health Promotion

personalized interventions (*e.g., just-in-time intervention*) and tracking disease progression (*e.g., diabetes*).

We have seen great success in the ability of commercial, off-the-shelf wearable sensors to count steps [27], estimate heart rate [7], detect sleep [77] and recognize physical activity [27, 77]. Beyond simple activities, we have seen limited commercial success in the usability of wearable sensors. A handful of research studies have demonstrated the feasibility of wearable sensors to detect human emotional states (mood[111], stress[95]), detect activities of daily living (eating[105], smoking[5], drinking[6]), and detect activities specific to certain populations (drug addiction[43], autism[26], epilepsy seizures[89]). We attribute this limited success to the many challenges that arise in complex activity detection. To illustrate these challenges, we present a generic activity detection framework used to detect complex human activities.

The generic activity detection framework consists of three components: (1) data sensing and logging, which includes an appropriate choice of sensing modality and data logging frequency. (2) feature extraction, in which raw sensor data is analyzed to extract useful information in order to detect target activities. Here the challenge is to extract features relevant to the task from sensor data that may be corrupted by many artifacts. (3) classification, where a machine learning model is used to detect or predict complex activities of interest. Additionally, for many applications, we develop this activity detection framework in one environment and would like to deploy it in another environment and/or another user cohort. Among the many challenges that arise in this framework, we focus on three challenges in this dissertation, which we detail in the nest section.

## 1.2 Challenges and Contributions

In this section, we present three challenges that arise in the generic activity detection framework and our contributions to addressing each of them.

**1. Challenges in extracting nuanced features from wearable sensor data**

Most wearable devices are not approved for medical use but are rather consumer grade devices with limited functionalities. Typically, off-the-shelf wearable sensors are non-adhesive and placement of sensors is not specific to any one location. These traits which make wearable sensors easy to use also directly affect the quality of data due to devices shifting, occasionally dropping contact with body, and introducing noise in data streams. These problems are even more exacerbated when performing complex activities, (*e.g., drug use*), which systematically give rise to windows of poor quality of sensor data thereby rendering them unusable. All these factors affect the quality and volume of sensor data available for further analysis.

While there are a diverse set of wearable sensors available to choose from, there are only a limited number of physiological signals that can be measured using wearable sensors. The vast majority of prior work relies on extracting a limited number of features from physiological signals for use in downstream tasks. Common examples include heart rate from electrocardiogram (ECG) and photoplethysmogram (PPG) signals, breathing rate from respiratory inductance plethysmography, electrodermal activity from galvanic skin response and core body temperature. While these features are adequate for detecting simple activities, they can be inadequate for detecting complex activities. For example, arrhythmia detection relies on precise location and shapes of ECG waves[46], special populations with autism and epilepsy seizures require access to more nuanced features from galvanic skin response[38, 71], and stress markers rely on heart rate variability [45], which is relatively more challenging to estimate when compared to heart rate. Despite having access to raw sensor data, it is much more challenging to extract nuanced features due to the artifacts introduced by wearable sensors, as well as the inherent between user variance in physiological signals.

To address this challenge, we develop probabilistic graphical models to encode domain knowledge when extracting features and learning models from streams of sensor data. We demonstrate the usefulness of our approach in the electrocardiogram (ECG) signal domain.

The ECG signal consists of repetitive patterns, which we encode as domain knowledge in probabilistic graphical models to perform structured prediction. This work is published in $5^{th}$ ACM conference on Bioinformatics, Computational Biology, and Health Informatics, September, 2014 [73]. To the best of our knowledge, this is one of the first lines of work to demonstrate the use of structured prediction models to effectively and principally encode domain knowledge in mobile health settings. This work has directly or indirectly inspired several other works on the use of structured prediction models in mobile health applications [3, 8, 75, 15, 85].

**2. Challenges in deploying the activity detection framework in real world settings**

A common study design to many recent mobile health (mHealth) studies is a two-stage study design [105, 43]. The first stage is executed in controlled settings in order to obtain clean, isolated physiological responses within specific target activities or conditions. In the second stage, activity detection models are deployed in real world settings. By designing experiments in controlled settings, we can exert control over the duration and sequence of activities of interest and limit the occurrence of confounding activities. As a consequence, data gathered in controlled settings can have low ecological validity with limited generalization performance. This is a serious limitation when deploying the activity detection framework in real world settings. One other advantage to designing experiments in controlled settings is that it allows for gathering reliable ground truth labels at fine granularity (*e.g., start and end times of target activity and activity types*). This is often not the case in real world settings in which ground truth labels are unavailable or unreliable. In some applications, reliable labels are only available at coarse granularity (*e.g., number of cigarettes smoked in one hour time periods*). This leads to a mismatch in labels collected in controlled settings and real world settings, making it challenging to evaluate lab-based models in field settings.

Lastly, as other prior work has noted, there is significant variability between users when performing the same activity in different environments [9, 121]. This variability is even more pronounced for complex tasks when compared to tasks with repetitive patterns.

To address this challenge, we develop domain adaptation techniques to improve lab-to-field generalizability. Specifically, our use case is a novel drug use detection study using wearable sensors. In the first stage, we develop a drug use detection framework in lab settings to demonstrate the feasability of using wearable sensors. In the second stage, we deploy this framework to detect drug use in real-world settings. The framework, as hypothesized, exhibited poor generalization performance due to the change in the environments. We identified three shifts in datasets gathered in the lab and field settings. We develop domain adaptation techniques to handle all three dataset shifts. When handling all three dataset shifts, we show that we can achieve good generalization performance, better accuracy than self-report, and comparable accuracy to existing gold standards in drug testing. We published this research in the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing [74] and the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing [72].

**3. Challenges in the availability of ground truth labels in real world settings**

While wearable sensors can be readily deployed leading to collection of large volumes of unlabeled sensor data there are significant challenges when collecting associated ground truth labels. We alluded to some of the problems in collecting ground truth labels in the previous challenge as well, but in this challenge we focus on the scarcity of ground truth labels in real world settings. We require ground truth labels to train prediction models in the first place and to personalize prediction models to each user. Availability of ground truth labels from real world settings is essential to deployment of the activity detection framework in the same environment.

The most popular and practical approach is to request that users proactively supply labels, but the manual effort involved is prohibitive [91, 110]. Another approach is to

query users for labels selectively via prompts. Here again, it is challenging to determine when users are willing to respond and are most likely to supply correct labels. [39, 18]. In yet another approach, experimenters follow study participants to note their activities, which are then used as labels [17]. This last approach is simply impractical and will not scale to large user studies.

To handle this challenge, we develop a hierarchical active learning framework to minimize the number of labeled examples required per user. At the core of this framework are active learning methods to address label scarcity in real world settings. Our framework also allows for sharing of labeled examples between users that are very similar, further minimizing the number of labeled examples required. We show that we can achieve comparable performance to fully personalized models, but with a significant reduction in labeling effort. This work is a proof of concept that active learning methods can reduce the manual labeling effort in real world health monitoring and mobile health applications.

## 1.3 Case Studies

Below, we provide a brief description of the two case studies used in this dissertation.

**1. Cocaine use detection using wearable electrocardiogram sensors**

The first case study is a novel cocaine use detection study using a wearable chest band sensor. The long term goals of this work were to provide personalized treatment plans to cocaine addicts and to improve our understanding of addiction related triggers. The short term goal of this work is to reliably detect cocaine use with wearable sensors. We collected data from habituated cocaine users in both the lab and field settings. Our choice of sensing modality was wearable ECG sensors since cocaine is believed to cause robust and predictable changes in ECG (discussed in more detail in Chapter 2). In this dataset, we have access to about $900$ hours of ECG data from $15$ participants. More details on the lab and field study protocols are given in Sections 4.1 and 5.1 respectively.

**2. Human activity recognition using wearables**

The second case study focuses on human activity recognition. The goal of human activity recognition is to segment and label various activities of interest given continuous streams of sensor data. We use a publicly available dataset that has data from 60 participants in real world settings. Users have supplied labels for about 116 activities. Subsequently, the experimenters cleaned the user supplied labels when there were label inconsistencies. In this study, users wear a Pebble wrist watch paired to a study smartphone. More details on this dataset can be found in Section 6.1.1.

While cocaine use detection applies to a specific population of individuals, activity recognition has widespread applications from chronic diseases to fitness monitoring. Despite their differences, both applications exhibit significant between user variability and hence personalization maybe useful to improve accuracy.

At this point, it is worth discussing the need for two diverse datasets in the dissertation. Two of the three challenges discussed above pertain to the cocaine use study for which we develop and evaluate machine learning methods. However, due to lack of reliable ground truth labels in the cocaine field study, we are unable to evaluate our techniques to collect ground truth labels in real world settings. One way to get around this problem is to use the lab cocaine dataset (for which we have reliable ground truth labels) to simulate the dynamics of users supplying labels. But this requires us to create transitions among scripted activities which are abrupt and artificial. We adopted the human activity recognition task due to the availability of a long term, labeled dataset gathered in real world settings. Due to the simple nature of the task, most users provide reliable ground truth labels that can also be verified from the corresponding sensor data. This also serves to illustrate a practical problem in developing machine learning methods for wearable sensor data analysis – the scarcity of long term, annotated datasets collected in environments with high ecological validity.

## 1.4 Dissertation Outline

In this dissertation, we propose machine learning methods and models to address the three challenges described above. We develop and evaluate these methods on two case studies: cocaine use detection and human activity recognition.

In Chapter 2, we review machine learning models that we use in cocaine use detection and human activity recognition problems. In both case studies, we treat target activity detection as a classification problem in machine learning. We provide details on the classification models as well as how we perform domain adaptation and transfer learning in these models. We also provide relevant background on cocaine, a brief introduction to electrocardiogram and electrocardiography, and cocaine-induced morphological changes in ECG.

In Chapter 3, we present machine learning methods to extract morphology from noisy ECG sensor data. Our processing pipeline consists of two components for ECG morphology extraction. The first is a sparse coding model that learns sparse underlying basis representations of ECG waves that effectively handles the variance in shapes of the ECG waves across time and across users. The second is a conditional random field model that effectively encodes domain knowledge in the ordering and shapes of ECG waves to extract morphology from ECG cycles. We evaluate both components on ECG data gathered from wearable chest band sensors.

In Chapter 4, we present a framework to detect cocaine use in controlled settings. We present a framework that encompasses data sensing, data logging, ECG morphology extraction, feature aggregation and classification. We use the Zephyr BioHarness [117] wearable chest band sensor to gather ECG data. We extract ECG morphology as outlined in Chapter 3. We perform feature aggregation over temporal windows which are then classified as cocaine use or non-cocaine events. We evaluate this framework on a novel cocaine use study on habituated cocaine users in controlled clinical settings.

In Chapter 5, we extend cocaine use detection to real world settings. Our approach is to the deploy lab-based cocaine use detection framework (as outlined in Chapter 4) to detect cocaine use in real world settings. This is challenging due to the systematic differences in ECG feature distributions and label proportions between lab and field datasets respectively. As a result, directly deploying lab-based models results in poor generalization performance. We develop techniques to quantify and handle dataset shifts, which allows for lab-based models to be deployed more effectively in real world settings. In addition to the above dataset shifts, we propose methods to handle label granularity shift – a mismatch in label granularity between lab and field datasets. Handling this novel form of shift makes it possible to evaluate cocaine use detection in field settings. We evaluate these domain adaptation techniques on data gathered from habituated cocaine users in real world settings.

In Chapter 6, we focus on the problem of collecting ground truth labels in wearable sensing applications. This work is largely inspired by the challenges and lessons learned with data collection in the cocaine use field study. We observed that subjects had low compliance with supplying labels and on many occasions supplied incorrect labels. Among the many challenges pertaining to collecting ground truth labels, we specifically focus on minimizing the number of labels required to learn personalized prediction models. We present a transfer active learning framework that learns personalized prediction models while minimizing the number of labeled examples per user. The core of this framework is active learning, which determines what examples to label during learning. This is complemented by transfer learning, which leverages similarities between users to further reduce the number of examples required by the active learner. Both techniques work in tandem with the goal of improving performance. We evaluate the feasability of these techniques on the human activity recognition dataset.

# CHAPTER 2

# BACKGROUND

## 2.1 Machine Learning Methods and Models

In this section, we provide background on machine learning methods that we use in the cocaine use detection and human activity recognition tasks.

### 2.1.1 Logistic Regression

We use a standard linear logistic regression classifier for binary classification [28] since it directly outputs probabilities, which are often more desirable in health settings such as ours. We denote random variables in upper case (*e.g.,* $X$) and the values these variables take in lower case (*e.g.,* $x$). Given a feature vector $X \in \mathbb{R}^D$ consisting of $D$ features, the binary logistic regression classifier returns the probability of that feature vector belonging to the positive class:

$$P(Y = +1|X = x) = \frac{1}{1 + \exp\left(-(b + W^\top x)\right)} \tag{2.1}$$

where, $W$ is a length $D$ vector of feature weights, $b$ is the bias term and $Y \in \{-1, +1\}$ represents the label for the instance $X$. An equivalent representation to compute class probabilities is,

$$P(Y = y|X = x) = \frac{1}{1 + \exp\left(-y(b + W^T x)\right)} \tag{2.2}$$

This prediction model has a linear decision boundary specified by the weights $W$. The default classification rule when using logistic regression is to predict that the data case belongs to the positive class if $\frac{P(Y=+1|X=x)}{P(Y=-1|X=x)} > 1.0$.

Learning the weights of the logistic regression classifier is accomplished by maximizing the log likelihood of the training data using numerical optimization [28]. Given a dataset $\mathcal{D} = \{(y_n, x_n)\}_{n=1:N}$ of $N$ labeled examples, the objective function is defined as,.

$$\underset{b,W}{\operatorname{argmin}} \sum_{n=1}^{N} \log\left(1 + \exp(-y_n(b + W^T x_n))\right) + \lambda \|W\|_2^2 \tag{2.3}$$

The first term is the log likelihood of $N$ data examples and the second term provides regularization of the norm of the weight vector to minimize overfitting. $\lambda$ determines the relative contribution of the two terms to the objective function. Minimizing this objective function is equivalent to minimizing the *logistic loss*. It is very similar to the hinge-loss function used in support vector machines [28]. It is a continuous, convex optimization problem with no constraints. It can be solved using any gradient-based optimizer. In this work, we use the limited memory Broyden Fletcher Goldfarb Shanno (BFGS) algorithm [78]. This binary model can be extended to multiple classes using multinomial logistic regression [36].

### 2.1.2 Transfer Learning in Logistic Regression

In the standard logistic regression model's objective function, the regularization term penalizes the square of the $l_2$ norm of the weight vector $W$, as in Equation 2.3. This has an equivalent interpretation as incorporating a zero-mean Gaussian prior with covariance $\frac{1}{2\lambda} I$ on the weights. One can also incorporate prior knowledge into the prediction model by penalizing the deviation of the model parameters, $W$, from a prior set of model parameters, $W_p$, as shown below,

$$\underset{W,b}{\operatorname{argmin}} \sum_{n=1}^{N} \log\left(1 + \exp(-y_n(b + W^\top x_n))\right) + \lambda \|W - W_p\|_2^2 \tag{2.4}$$

Setting $W_p = 0$ yields a standard penalized $l_2$ model. Prior model parameters $W_p$ can also be set to model parameter estimates derived from a source domain, effecting a simple, but powerful form of transfer learning [16].

### 2.1.3 Domain Adaptation in Logistic Regression

The generalization performance of prediction models is affected when the test data distribution is shifted away from the training data distribution. This shift is referred to as a dataset shift problem in machine learning. Examples of dataset shifts in wearable sensing applications include training a prediction model on data gathered in controlled clinical settings and testing on data from real world settings. It has been demonstrated that making the prediction model aware of dataset shift leads to improved performance [102, 32]. One approach to making prediction models aware of dataset shifts is to assign importance weights to the training distribution to reweight the training distribution to match that of the test distribution. This approach to domain adaptation – reweighting the dataset in the source domain to help with prediction in the target domain – is called importance weighting. When using importance weights, the prediction model parameters are tuned to the reweighted dataset and the model often performs better on the test set.

We incorporate this reweighting directly into the objective function of the prediction model. For example in logistic regression the objective function has two terms: the log likelihood and the regularizer. To accommodate the reweighting of data examples to mitigate dataset shifts, we augment the standard conditional log likelihood with a per data case importance weight, $\delta(y, x)$, that can depend on the features and the label of the data case, as seen below.

$$\underset{W,b}{\operatorname{argmin}} \sum_{n=1}^{N} \delta_n(y_n, x_n) \log(1 + \exp(-y_n(b + W^T y_n))) + \lambda \|W\|_2^2 \tag{2.5}$$

### 2.1.4 Conditional Random Fields

Conditional random fields (CRFs) are a sub-class of probabilistic graphical models [54] that generalize independent probabilistic classifiers like logistic regression [42] to the case of structured prediction. CRF models contain feature variables and label variables connected in a graph that captures problem-specific probabilistic dependencies between the label variables. In this dissertation, we use a linear chain CRF model like the one

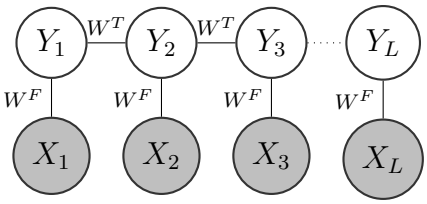


Figure 2.1: Linear Chain CRF

shown in Figure 2.1. Here $L$ corresponds to the length of the input sequence. The shaded nodes $X_1$ to $X_L$ represent the feature variables, and the unshaded nodes $Y_1$ to $Y_L$ are the corresponding label variables. We assume the label variables take values in the set $\mathcal{V}$. The feature variables $X_i \in \mathbb{R}^D$ represent a $D$-dimensional vector. Each $(X_i, Y_i)$ pair is associated with a feature potential $W^F$ that captures the dependence between the features and the associated labels. Each pair of adjacent labels, $Y_i, Y_{i+1}$, are associated with a transition potential $W^T$ to capture the first order Markov dependence between pairs of label values.

In a CRF model, the probability of a sequence of observed labels $y = [y_1, ..., y_L]$ conditioned on the observed feature values $x = [x_1, ..., x_L]$ is given by,

$$P_{\mathbf{W}}(Y = y|X = x) = \frac{\exp(E_W(y, x))}{Z_W(x)} \tag{2.6}$$

where $E_W$ is the energy function of the model and $Z_W(x)$ is the partition function. The feature and transition potentials that define a CRF model are collectively represented by $W = [W^F, W^T]$. The energy function is given by,

$$\begin{aligned} E_{\mathbf{W}}(y, x) = \big( &\sum_{i=1}^{L} \sum_{d=1}^{D} \sum_{v \in \mathcal{V}} W^F_{dv}[y_i = v] x_i \\ &+ \sum_{i=1}^{L-1} \sum_{v \in \mathcal{V}} \sum_{v' \in \mathcal{V}} W^T_{vv'}[y_i = v][y_{i+1} = v'] \big) \end{aligned} \tag{2.7}$$

The partition function is given by,

$$Z_{\mathbf{W}}(x) = \sum_{y \in \mathcal{V}^L} \exp(E_{\mathbf{W}}(y, x)) \tag{2.8}$$

The dimensions of $W^F$ and $W^T$ matrices are $d \times |\mathcal{V}|$ and $|\mathcal{V}| \times |\mathcal{V}|$ respectively, where $|\mathcal{V}|$ is the cardinality of the label set $\mathcal{V}$. The unknown parameters $W = [W^F, W^T]$ must be learned from training data before using the model for inference. Given a dataset $\mathcal{D} = \{(y_n, x_n)\}_{n=1:N}$ of fully labeled training sequences, the parameters can be estimated by maximizing the objective function as shown below,

$$\underset{W}{\operatorname{argmax}} \sum_{n=1}^{N} \log P_{\mathbf{W}}(y_n|x_n) - \lambda \|W\|_2^2 \tag{2.9}$$

The first term is the conditional log likelihood and the second term provides regularization of the weight matrices to avoid overfitting. Transfer learning can also be incorporated into the CRF model, like in logistic regression, by penalizing the deviation of the weight matrices $W$ from a prior set of model parameters, $W_p$:

$$\underset{W}{\operatorname{argmax}} \sum_{n=1}^{N} \log P_{\mathbf{W}}(y_n|x_n) - \lambda \|W - W_p\|_2^2 \tag{2.10}$$

In either the $l_2$ penalty or transfer case, this objective function is strongly convex, so gradient-based methods are guaranteed to find the unique optimal solution. Computing the gradients requires all single label variable marginal probabilities as well as pairwise marginal probabilities for all pairs of adjacent label variables [54]. All of these marginal distributions can be found in time linear in the length of the chain (as can the partition function) using the well-known sum-product belief propagation algorithm [54].

### 2.1.5 Sparse Coding

Sparse coding is an unsupervised dimensionality reduction technique. It reconstructs input vectors as sparse linear combinations of a set of $K$ basis vectors $\beta_k$ and coefficients

$\alpha_k$ [80]. The objective function is as follows,

$$\underset{\alpha,\beta}{\operatorname{argmin}} \sum_{n=1}^{N} \left|\left| x_n - \sum_{k=1}^{K} \alpha_k \beta_k \right|\right|_2^2 + \lambda \|\alpha\|_1^1 \tag{2.11}$$
$$s.t. \|\beta_k\|_2^2 \leq 1 \forall_k \in 1, ..., K$$

The first term is the reconstruction error between input data examples and linear combinations of coefficients and basis elements. The second term is the $l_1$ norm of the coefficients, which induces sparsity. Given a dataset $\mathcal{D} = \{x_n\}_{n=1:N}$ of labeled examples, the basis itself is learned to minimize the sum of the errors between each data case and its reconstruction under the constraint of $l_2$ regularized basis vectors, as seen above. The typical approach to solving this problem is an alternating minimization strategy since the objective function is not jointly convex in both $\alpha$ and $\beta$, but is convex in one variable when fixing the other. We used the SPAMS toolbox to perform sparse coding [68].

The advantage of sparse coding over methods like principal components analysis (PCA) is that it produces sparse feature vectors, which can help to reduce over-fitting when these features are used for classification. Unlike PCA, sparse coding can also be used to learn an over-complete basis ($K > D$). This can help to make classification problems easier by making the feature vectors more linearly separable than the original data in the higher-dimensional feature space.

### 2.1.6 Active Learning

Active learning methods query an oracle for labels to improve the performance of a prediction model [63]. Traditional approaches to active learning have focused on the pool-based setting where all unlabeled examples are available to query and the goal is to pick and choose examples with high utility, which generally leads to improvement in the performance of the prediction model. We first describe various components of the active learning framework, followed by a description of the basic active learning algorithm.

#### 2.1.6.1 Prediction Model

At the core any active learning algorithm is a prediction model. The goal of active learning is to improve the performance of this prediction model by picking unlabeled examples to label. Most active learning algorithms also use the prediction model to determine the utility of unlabeled examples. At the very first iteration, when no labeled examples are available, the prediction model randomly guesses the utility of unlabeled examples. In subsequent iterations, when the prediction model has access to a sufficient number of labeled examples, this generally leads to better estimates of utility. Typically prediction models are retrained after each query to accurately represent what the model is certain and uncertain about. Classification models such as neural networks [34], support vector machines [107] and multinomial logistic regression [97] have been investigated as prediction models.

#### 2.1.6.2 Querying Strategy

When evaluating a pool of unlabeled examples, the querying strategy is used to compute utilities. The higher the utility of an unlabeled example, the more likely it is to be queried for a label. Querying strategies are largely organized into optimizing decision theoretic or information theoretic criteria. The former queries for examples with the objective of minimizing error on the test dataset, while the latter queries examples with the objective of shrinking the hypothesis space [44]. A vast majority of querying strategies minimize decision theoretic criteria and rely on prediction models to determine the utility of unlabeled examples. Some of the most popular querying strategies include uncertainty sampling, query-by-committee, expected error reduction, variance reduction, model change and their hybrids [101].

#### 2.1.6.3 Oracle

Oracles provide labels for a chosen example. Additionally, active learning makes the assumption that the oracle is always responsive to queries and always provides the correct label. In practice these assumptions may not hold since oracles (*e.g., human labelers*)

may or may not respond to a query and oracles may inadvertently supply incorrect labels. Relaxing these assumptions leads to proactive learning [25].

#### 2.1.6.4 Label Cost and Budget

Active learning typically assumes that the cost of obtaining a label is uniform irrespective of label type or perceived difficulty in supplying a label. In practice this assumption can be utilized to train the active learning algorithms to issue queries only for windows of sensor data where oracles are likely to provide labels by making certain windows very expensive to query (*e.g., when the user is driving*). Budget constraints enforce limits on the number of queries that can be issued. Active learning terminates when the budget is exhausted or there are no more unlabeled examples in the pool.

#### 2.1.6.5 Algorithm

In this section we describe the basic active learning algorithm for classification. To do so, we first introduce some notation. Assume we have access to a dataset of $U$ examples of the form $\{(x_1, y_1), ..., (x_U, y_U)\}$ where, each $x_i \in \mathbb{R}^D$ be data samples and $y_i \in \mathcal{V}$ be labels. The goal of active learning is learn a prediction model: $f : \mathbb{R}^D \rightarrow \mathcal{V}$. In order to simulate active learning, we assume the $U$ data samples are available in a sample pool as unlabeled examples, and the corresponding labels are only accessible through an oracle.

The general setup is illustrated illustrated in Figure 2.2. Active learning proceeds by first randomly drawing an unlabeled example, $X$, from the sample pool (Figure 2.2 step 1) and querying the oracle for a label (Figure 2.2 step 2). The labeled example, $(x, y)$, is added to the labeled set $\mathcal{D}$ and the prediction model $f$ is retrained (Figure 2.2 step 3). This updated prediction model is used in querying strategies to compute utilities for all available unlabeled examples in the sample pool (Figure 2.2 step 4). Following this step, the example with the highest utility is selected to be queried by the oracle (Figure 2.2 steps 1 and 2). These steps are repeated for a predetermined budget, $B$, or until the sample pool is exhausted.

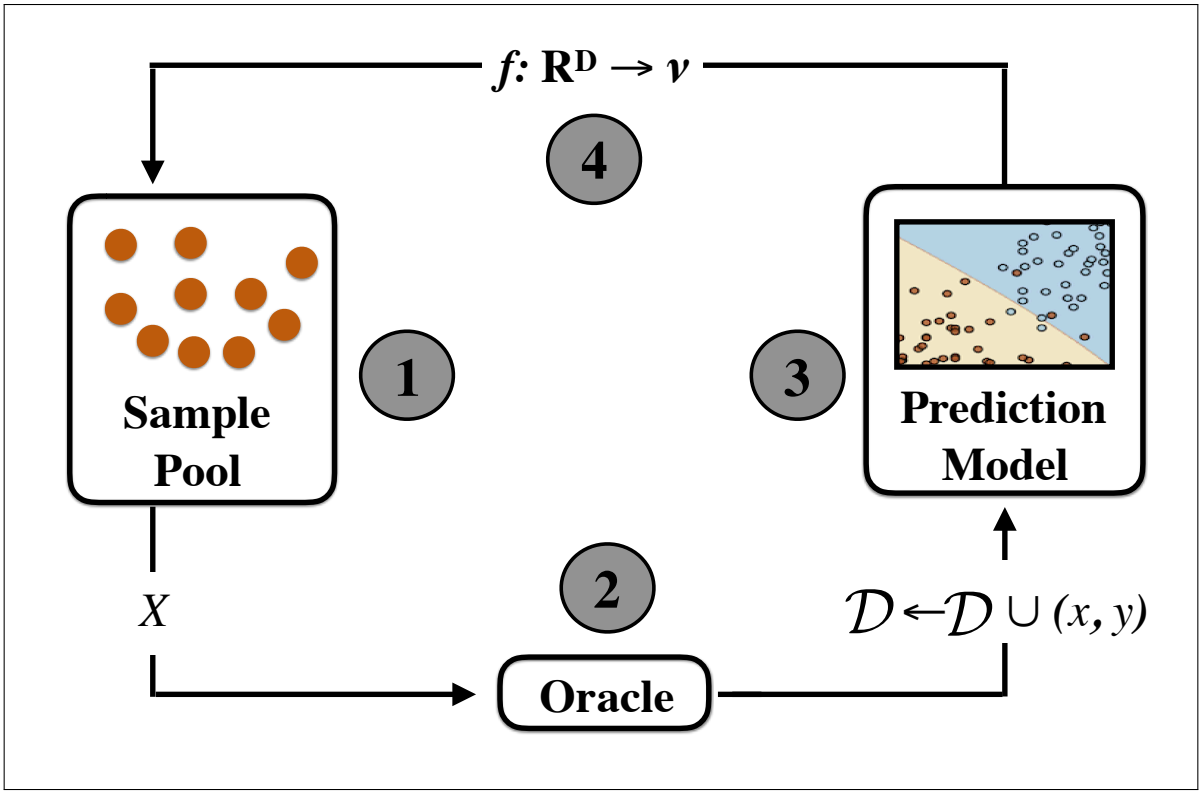



Figure 2.2: Active learning algorithm

The original active learning problem can be viewed as optimally selecting $N$ data examples from a sample pool of $U$ data examples. This problem is intractable in general [37], but in practice, it has been shown that a myopic (greedy) approach to active learning leads to good solutions [101].

### 2.1.7 Hierarchical Agglomerative Clustering

Hierarchical agglomerative clustering is a type of unsupervised clustering algorithm that recursively merges clusters pairwise based on a linkage distance criterion [92]. The algorithm begins by having access to $M$ clusters, in each iteration it merges a pair of clusters until all $M$ clusters are merged into a single cluster.

The advantage of this clustering approach is that we do not need to specify the number of clusters ahead of time. However, hierarchical agglomerative clustering requires defining a notion of similarity or distance to merge clusters. Typical examples include distance-

based measures (*e.g., euclidean distance*). In personalized health monitoring applications, we perform clustering on users versus individual data examples hence the distance metrics will need to be computed using all data examples from each user. A default approach is to compute the mean of feature vectors for each user and then compute the similarity using distance-based measures as described above. Many alternates exist that leverage summary statistics of each user to compute similarity [10, 48, 60, 116]. The results of hierarchical agglomerative clustering are often organized and presented via a dendrogram. The leaf nodes in the dendrogram correspond to the original $M$ clusters and the non-leaf nodes are a result of the recursive merges.

We used the implementation of hierarchical agglomerative clustering in Python's sklearn module [87]. We supplied a precomputed similarity matrix between all pairs of users. We provide more details on how we compute the similarity matrix in Section 6.4.6. In each iteration of the hierarchical agglomerative clustering, two users with the smallest distance in the smilarity matrix are merged. All other settings were set to default.

## 2.2 Cocaine Use and Electrocardiogram

In this section we provide relevant background on cocaine use, electrocardiography and cocaine-induced morphological changes in electrocardiography signals.

### 2.2.1 Psychological and Physiological Effects of Cocaine Use

Cocaine is a powerful, addictive stimulant drug made from coca plants native to South America. In 2014, global cocaine use was reported to be close to $18$ million users[1]. Cocaine is typically consumed in one of three forms: as hydrochloride salt, freebase cocaine, or crack cocaine [88]. Once consumed, cocaine acts as a stimulant on the central nervous system, creating a feeling of euphoria and high energy. Cocaine taps into the reward pathways in the brain that usually respond to other rewarding and pleasurable behaviors such

[1]United Nations World Drug Report, 2016

as eating and sex. This stimulus-reward response serves as a positive reinforcement to the brain leading to an addictive, compulsive behavior [88].

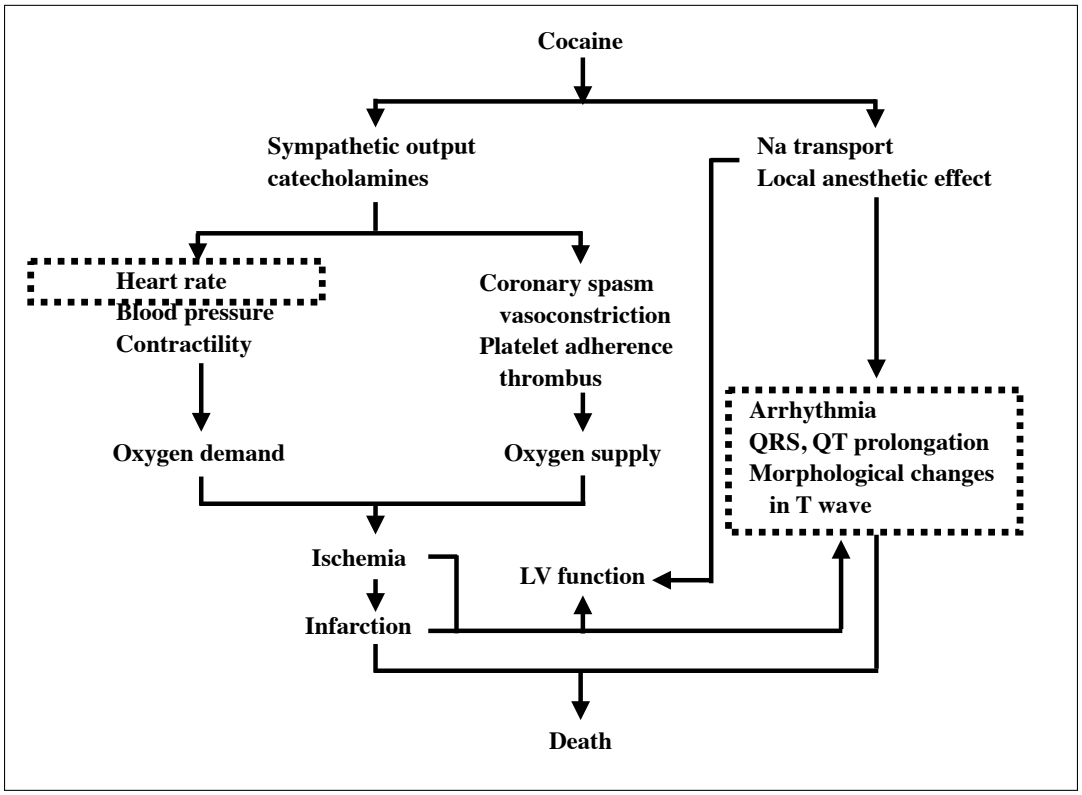



Figure 2.3: Psychological and Physiological effects of cocaine use in humans. Figure recreated from [99]. Boxes are drawn around physiological effects that we are interested in using to detect cocaine use with wearable sensors.

Cocaine addiction is associated with predictable and highly characteristic physiological, behavioral, and subjective effects [79]. Such effects derive directly from cocaine's well-established pharmacological mechanism of action: it is an indirect agonist/monoamine reuptake inhibitor. By virtue of its peripheral actions on the sympathetic nervous system, cocaine produces changes in primary indices of cardiovascular and neurological function (increases in heart rate, systolic, and diastolic blood pressure and pupillary diameter) and tremors and muscle twitches [81]. As a psychostimulant, cocaine also produces a characteristic profile of centrally-mediated, behavioral effects including increased restlessness, irritability, panic attacks, paranoia and psychosis [82]. In Figure 2.3 we show a flowchart of the psychological and physiological effects of cocaine use in humans.

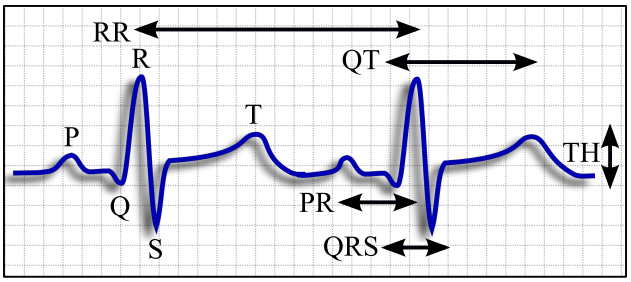



Figure 2.4: This figure illustrates two ECG cycles. The P, Q, R, S and T waves are labeled on the left cycle. ECG morphological features such as RR, QT, PR, QRS intervals and T wave height are labeled on the right cycle

### 2.2.2 Electrocardiogram and Electrocardiography

An Electrocardiogram (ECG) is a graphical recording of the heart's electrical activity as a function of time. The source of the electrical impulse is the sinoatrial node, impulese then pass through the arteries and finally through the ventricles. The depolarization and repolarization of these muscles causes the heart to pump blood. A healthy heart has an orderly progression of electrical impulse through the heart's muscles which translates to a sequence of waves in the ECG signal.

Figure 2.4 illustrates two cardiac cycles of an ECG signal. We can see that each cycle is characterized by a series of five deflections away from the baseline referred to as the P, Q, R, S and T waves. These five deflections are collectively known as the PQRST complex. The P wave corresponds to the atrial depolarization, the QRS wave corresponds to the ventricular depolarization and the T wave corresponds to the ventricular repolarization. Typically, ECG is recorded by placing 12-lead electrodes on the surface of the skin.

### 2.2.3 Effects of Cocaine on Electrocardiogram

There is substantial evidence from human and animal studies that cocaine use causes changes in cardiovascular function that are observable in ECG signals. Cocaine use has a robust effect on heart rate, causing it to increase significantly [99]. An increased heart rate manifests as a reduced RR interval, as shown in Figure 2.4. Cocaine has also been reported to have an effect on the QT interval [99]. Some research has also made use of a

corrected QT interval, QTc, meant to partially normalize out the effect of heart rate on QT interval. QTc is typically computed as the length of the QT interval divided by the square root of the length of the RR interval (Bazett's correction) [113]. Two studies have reported QTc prolongation in the presence of cocaine [67, 62]. Magnano et al. have also reported changes in the height and shape of T waves in the presence of cocaine. Animal studies have pointed to additional effects of cocaine on the PR and QRS intervals [33, 98].

### 2.2.4 Summary

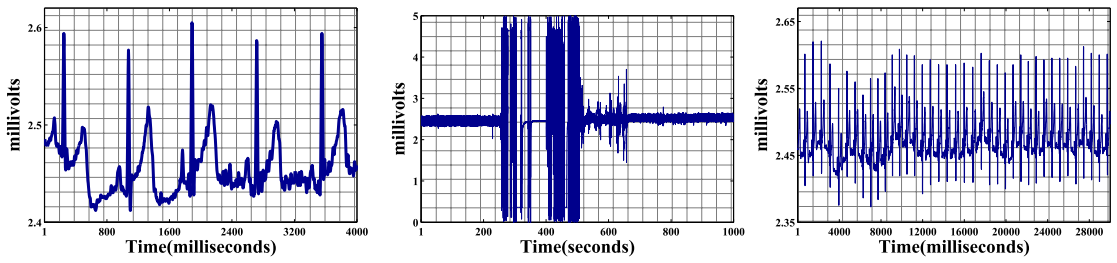
In this chapter, we presented background information on machine learning models and techniques that we use in this dissertation. We use the conditional random field model in Chapter 3 to extract morphology from ECG signals. The background material on ECG and cocaine-induced morphological changes will be useful when extracting features from ECG morphology to be used in cocaine-use detection. We use the penalized logistic regression model in detecting cocaine use as well as specific activities in the problem of human activity recognition. We utilize domain adaptation techniques in Chapter 5 to adapt a lab-based cocaine use detection model to detect cocaine use in field settings. Lastly, we utilize the active learning framework in Chapter 6 to collect ground truth labels in human activity recognition problem. The prediction model used in active learning is penalized logistic regression with transfer learning.

# CHAPTER 3

# PROBABILISTIC GRAPHICAL MODELS TO ENCODE DOMAIN KNOWLEDGE IN ECG FEATURE EXTRACTION

Many physiological signals exhibit repetitive patterns. Examples include respiratory (inhalation-exhalation cycles), photoplethysmogram and elecrocardiogram signals. In order to detect complex target activities (*e.g., smoking, drug use*) we would like to extract features from these signals, particularly information on the constituent peaks within each repetitive pattern. One approach is to segment the peaks and label each segment independently using existing machine learning models (*e.g., SVM, multinomial logistic regression*). The disadvantage of this approach is that it does not leverage the temporal ordering of peaks within each cycle. An alternate is to segment and perform joint labeling of the sequence of peaks within each cycle. This approach leverages the temporal ordering of peaks. In this chapter, we illustrate the utility of sequential labeling using structured prediction models applied to one type of physiological signal: the electrocardiogram.

A substantial body of work has explored the use of wearable ECG sensors for applications in personalized health monitoring [40, 31, 86, 111]. Nearly all of these studies used instantaneous heart rate and heart rate variability as features in conjunction with features from other sensing modalities. For more complex tasks such as cocaine use detection or arrithymia detection, we would like access to more nuanced ECG features, specifically ECG morphology. For example, one of the symptoms of myocardial infarction is elevation or depression of the ECG segment between the S and T peaks. Hence we require access to locations of both the S and T peaks to detect ST elevation or depression which is useful for mycardial infarction.




(a) (b) (c)

Figure 3.1: This figure depicts some of the issues that occur when using a wearable ECG device. (a) The data are inherently noisy compared to ICU-quality ECG. (b) Various forms of signal dropout occur in our data, including cases that manifest as extreme noise. (c) The data are also subject to baseline drift even over short time scales.

Extracting morphology from ECG data is a challenging problem due to inherent variability in locations and shapes of ECG waves coupled with underlying target activity and/or medical conditions that dynamically change ECG morphology. These problems are exacerabated when using consumer grade wearable sensors with a small number of non-adhesive electrodes. We present several examples of raw data obtained from wearable ECG sensors in Figure 3.1. This figure illustrates various difficulties with the use of a wearable sensor like the Zephyr BioHarness [117] chest band where the electrodes are not adhesive.

**1. Data quality**: Figure 3.1a gives an indication of how noisy the raw data is in the best case. We also often see ECG periods that have larger-scale distortions where the R wave may still be evident while the other waves are not discernible. Such distorted periods would not pose a difficulty for features based on the RR interval only (heart rate), but they do pose challenges when attempting to extract the complete PQRST complex. Fortunately, these distorted periods appear to be transients and don't frequently occur in long runs.

We also observe that there is significant variance in shapes of ECG peaks even over short time intervals. For example, in Figure 3.1a, there is substantial variability in shapes of the T waves especially at the start and end points of these waveforms. This makes it challenging to directly use the shapes to detect ECG peaks.

**2. Sensor dropout**: Figure 3.1b shows an example of signal dropout resulting in extended intervals of extreme noise. They typically result from large-scale disturbances to the sensor like completely removing or readjusting the chest band. These intervals are easy to identify because their characteristics differ widely when compared to normal signal. They contain no useful information and no features can be extracted from them. Ideally, we would want our feature extraction techniques to elegantly ignore such windows without manual interventions.

**3. Baseline shifts**: Figure 3.1c shows the degree to which the signal baseline drifts over short time spans. The baseline is also observed to drift over longer time spans. The long-run drift is likely due to slippage of the sensor over time. It is unclear what causes the short-run drift, but it is likely a hardware issue with the sensor itself. Again, the drift is a minor issue when extracting features based on the RR interval, but needs to be accounted for when extracting morphological features.

Despite these challenges, we would like to be able to accurately extract morphology from wearable ECG signals collected in real world settings. We observe that there is known structure in the ordering of valid ECG peaks within each cardiac cycle. Additionally, the ordering is preserved across time as well as as across users. The research question we address in this chapter is how to encode this domain knowledge to extract morphology in wearable ECG?

The primary contributions of this chapter are, we encode domain knowledge about ECG morphology via structured prediction models, specifically the linear chain CRF model. We demonstrate the usability of sparse coding – an unsupervised dimensionality reduction technique, to learn the underlying basis representations from ECG peaks that exhibit substantial variance in shapes. We evaluate the performance of both structured prediction models and sparse coding on real world ECG data gathered from wearable chest band sensors worn by users when consuming cocaine and performing other activities.

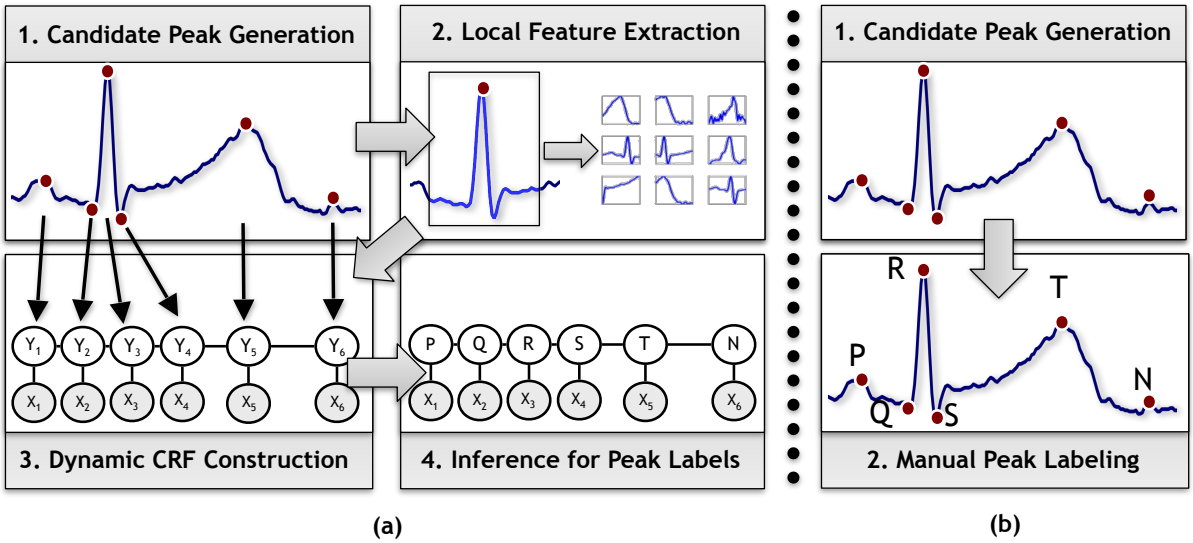



Figure 3.2: Illustrates (a) the ECG morphology extraction pipeline and (b) the ground truth data labeling pipeline

The rest of this chapter is organized as follows. We first describe our machine learning pipeline to extract ECG morphology (Section 3.1). We then describe the dataset that we used in our experiments (Section 3.2) and our empirical protocols (Section 3.3). We next present results to demonstrate the performance of ECG morphology extraction (Section 3.4). Finally, we review related work (Section 3.5) and present conclusions in Section 3.6.

## 3.1 ECG Peak Labeling Pipeline

Our approach to ECG peak labeling is based on exact probabilistic inference in chain-structured conditional random fields [58]. We label ECG peaks by following four primary steps: candidate peak generation, feature extraction, dynamic CRF graph generation and CRF inference. These steps are illustrated in Figure 3.2.

Before performing candidate peak generation, we perform a small amount of pre-processing on the raw ECG data. Raw ECG data is measured in millivolts and is typically recorded at hundreds of samples per second. Over extended time periods typically encountered in mHealth settings, ECG data from wireless on-body sensors exhibits signif-

icant baseline drift. We apply a standard low-pass Gaussian filter with a standard deviation of 600ms to estimate the baseline drift. We subtract the estimated drift from the raw data to yield baseline corrected data. All of our subsequent processing is based on baseline corrected ECG.

### 3.1.1 Candidate Peak Generation

The core of our approach is based on the idea of over-generating a set of candidate peak locations that will subsequently be labeled. Our aim is for this set to include the locations of all valid P, Q, R, S and T waves, as well as a minimal number of additional peaks caused by noise and other artifacts in the ECG data. Candidate peak generation is illustrated in Step 1 of Figure 3.2a. In this work, we apply Billauer's PeakDet method as we have found it be simple, fast and robust to noise [12].

### 3.1.2 Candidate Peak Feature Extraction

Given a set of candidate peak locations, we next extract features from the ECG data in the local neighborhood of each candidate peak. Specifically, we define a window of width $w$ samples centered at each candidate peak location and extract features from the ECG data contained in that window. In this work, we use sparse coding [80], as outlined in Section 2.1.5, to learn an over-complete basis from ECG data in a fully unsupervised manner. Sparse coding is an attractive choice for this application as it aims to describe each candidate peak as resulting from a sparse linear combination of basis vectors. The sparse coefficient vectors of these linear combinations are the sparse coding feature vectors. Sparse coding feature extraction is illustrated in Step 2 of Figure 3.2a. We combine the sparse coding feature vectors with additional features representing the height of each candidate peak.

### 3.1.3 Dynamic Conditional Random Fields

Given a set of candidate peak locations and their corresponding features, we construct a dynamic CRF model. We instantiate one label variable $y_i$ and one feature variable $x_i$ for each candidate peak location $i$. Importantly, we augment the label set with an additional label $N$ to indicate candidate peaks that do not correspond to any of the valid ECG waves. We set the feature vector $x_i$ to the sparse coding feature vector extracted for candidate peak $i$ in the previous step. Finally, we connect adjacent label variables to form a chain-structured graph. This process is illustrated in Step 3 of Figure 3.2a.

### 3.1.4 Learning and Inference

We perform maximum likelihood learning as outlined in Section 2.1.4. Once a CRF has been dynamically instantiated given the candidate peak locations, standard probabilistic inference methods can be used to infer the most likely values for the labels of the candidate peaks. The restriction to a chain-structured graph permits the application of linear-time exact inference methods [54]. Compared to an independent classification model like logistic regression, the CRF model is able to leverage the high degree of regularity in the ECG peak label transitions to aid in determining labels in regions of high noise. The CRF model has the advantage that it determines all peak labels jointly. This makes it more robust in cases where the local evidence for identifying *e.g.,* QRS waves is weak due to transient noise, but other waves like P or T are clearly discernible. Inference for an ECG trace with six peaks is illustrated in Step 4 of Figure 3.2a.

## 3.2 Dataset

Wearable ECG data was collected from six habituated cocaine users in a NIDA-approved clinical study in lab settings. The subjects wore the Zephyr BioHarness single-channel ECG chest band sensor [117]. The wireless sensor on these chest bands samples ECG data at 250Hz and transmits the data to a smartphone via bluetooth. Data were collected from

| Subject | Session length | # Samples | # Candidate peaks | # Labeled peaks | # Clusters |
|---|---|---|---|---|---|
| 1 | 6h36m | 5,624,954 | 217,941 | 3145 | 175 |
| 2 | 7h01m | 5,649,203 | 214,563 | 4558 | 462 |
| 3 | 7h42m | 6,537,902 | 301,317 | 3231 | 141 |
| 4 | 11h01m | 9,492,152 | 333,165 | 4104 | 219 |
| 5 | 11h55m | 6,736,003 | 245,995 | 2341 | 135 |
| 6 | 15h45m | 13,565,502 | 450,256 | 3966 | 332 |
| Total | 60h | 47,605,716 | 1,763,237 | 21,345 | 1464 |

Table 3.1: Dataset details including the total dataset sizes and the number of labeled peaks per subject.

subjects both in the presence and absence of cocaine use. More details on the protocol used to collect data in the lab settings are presented in Section 4.1.

## 3.3 Empirical Protocols

In this section, we describe the details of our training and evaluation protocols, features extracted around candidate peaks, generating ground truth ECG peak labels, baseline methods and evaluation metrics.

### 3.3.1 Manually Labeling ECG Peaks

We manually labeled over 20,000 candidate peak locations across six subjects. An advantage of our approach is that it is not necessary to fully label the raw ECG data to indicate which wave each individual sample belongs to. Instead, we first run the peak detection method to generate a set of candidate peak locations and then manually specify labels for the candidate peak locations only. This makes the entry of label information much faster. This approach is illustrated in Steps 1 and 2 in Figure 3.2b.

We also note that it is not necessary to fully label each sequence of candidate peak locations. For a chain-structured CRF, the learning algorithm only needs access to labels for pairs of adjacent label variables to estimate the transition parameters. For each available ECG trace, we labeled all candidate peaks in multiple short segments consisting of one to

three cardiac cycles. We refer to these segments as *clusters*. We designed a simple GUI to implement this labeling approach. Each labeled cluster serves as an instance that is used to learn the CRF parameters. The details of the dataset are listed in Table 3.1. Importantly, the use of the candidate peak generation step reduces the number of locations considered by the CRF during inference by more than $27$ times relative to making predictions for all time points.

### 3.3.2 Train, Validation and Test Splits

We randomly partition the available data for each subject into a training set consisting of $10\%$ of labeled clusters, a validation set consisting of $45\%$ of labeled clusters and a test set consisting of $45\%$ of labeled clusters, up to a total of 135 clusters, which is the minimum number across all subjects. These splits remain fixed for each subject throughout all experiments. The training sets are used to train the CRF model. The validation sets are used to select the CRF regularization parameter as well as to select between different feature sets (outlined in Section 3.3.4). The test sets are used to evaluate model performance.

### 3.3.3 Evaluation Protocols

Our evaluation uses three different learning protocols: within-subjects, between-subjects, and transfer learning. In the within-subjects protocol, we use the training and validation set for each subject $s$ to learn a subject-specific model and evaluate the model on the test data for subject $s$. In the between-subjects evaluation, for a given subject $s$, we pool the training set and the validation set for the subjects other than $s$ and use this pooled data to learn a model. We evaluate this model on the data for subject $s$. In the transfer learning evaluation, for a given subject $s$, we begin by learning the between-subjects model. We then use the learned weights from the between-subjects model to define a data-dependent regularizer when learning the within-subjects model for subject $s$. We present more details on this regularization towards the end of Section 2.1.4.

### 3.3.4 Extracting Features from Candidate Peak Windows

We set the size of the sparse coding basis to $K = 100$ and the sparsity parameter to $\lambda = 0.01$. The basis vectors were learned on ECG data extracted from a window of size $51$ samples ($204$ms) centered at each candidate peak location. These values were found to yield good performance in preliminary testing. For within-subjects training, we learn a separate set of sparse coding basis vectors from all of the data windows available for each subject $s$. In between-subjects training and transfer learning, we learn the sparse coding basis for subject $s$ using all of the available data windows for each subject other than subject $s$. We also make the height and the height squared of each candidate peak location available as additional features. We consider three different feature sets when learning a model: sparse coding only ($SC$), sparse coding with peak height ($SCH$), and sparse coding with height and height squared ($SCHH^2$).

We also consider several different ways of normalizing the data within each window prior to extracting the features. We consider subtractive normalization ($SN$) where we shift the data to have zero mean within each window; subtractive and divisive normalization where we shift the data to have zero mean within each window and re-scale it to have unit standard deviation within each local window ($SDN_L$); and subtractive and divisive normalization where we shift the data to have zero mean within each window and jointly re-scale all of the windows to have unit standard deviation globally ($SDN_G$).

In each of our experiments, we consider nine possible feature extraction pipelines given by the cross product of a choice of feature set from $\{SC, SCH, SCHH^2\}$ and a choice of data normalization framework from $\{SN, SDN_L, SDN_G\}$. For each model, we select one of the nine possible feature extraction pipelines using the validation set in each experiment.

### 3.3.5 Baseline Methods

In each of our experiments, we consider three different methods for extracting ECG peak locations and labels including our dynamic CRF approach, an independent multino-

mial logistic regression model (MLR) and the open-source ECGPUWave toolbox [83, 59]. Multinomial logistic regression is a special case of the CRF model that makes independent predictions for each candidate peak by not taking the transitions between adjacent peaks into account.

The ECGPUWave toolbox follows a traditional two-stage approach based on first identifying QRS complex locations and then performing a local search to identify the peak locations within each cardiac cycle. The ECGPUWave toolbox can operate in conjunction with a number of different QRS complex detectors. The classical detector used with ECGPUWave is the Pan-Tompkins detector [84]. We found that the more recent open-source WQRS detector of Zong et al. performed significantly better on our data. The WQRS detector is based on the curve length transform and has been shown to be very robust, achieving a QRS sensitivity of 99.65% and a gross QRS positive predictive accuracy of 99.77% on the MIT-BIH Arrhythmia Database [120].

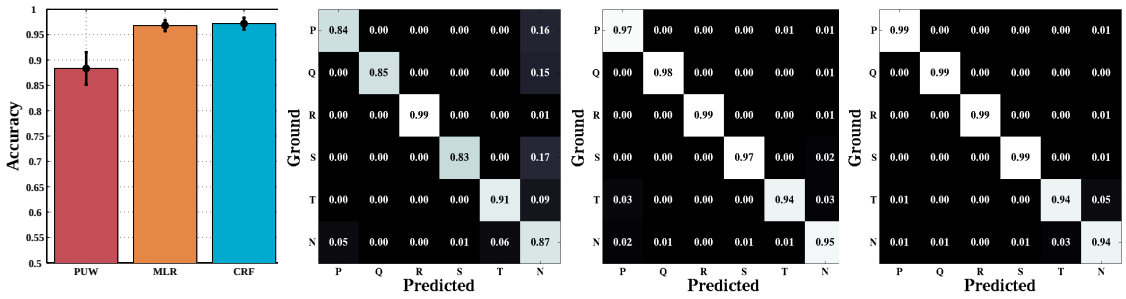
Since our data is labeled in terms of candidate peak locations and the CRF and MLR models are restricted to making predictions only at these locations, it is straightforward to assess their prediction performance. ECGPUWave can predict peaks at arbitrary locations so evaluating its accuracy requires some care. We apply a minimum weighted bipartite matching algorithm to the ground truth and ECGPUWave label locations to establish a correspondence between the true and predicted labels based on the distance between their time points [55]. We allow the ECGPUWave predictions to match ground truth labels within a window of plus or minus four samples (16ms). We define an ECGPUWave prediction as being correct if it is matched to a ground-truth label of the correct type. As a result of the matching window constraint, all correct peak labels must be within plus or minus four samples of a ground truth label of the correct type. Also due to the matching window constraint, some ECGPUWave predictions may not match any ground truth label locations. These predictions are considered as matching a ground truth label of $N$ (not a valid peak location), which counts as a labeling error.

We performed a preliminary analyses of the effect of window size on the number of matched ECGPUWave predictions and determined that the number of matches remains nearly constant as the window size is increased to nearly the average width of a full cardiac cycle. This indicates that the lack of a match for ECGPUWave typically means it did not identify a given wave type within a cardiac cycle at all. Failure to identify a given ground truth wave is assessed as a prediction of $N$ (not a valid peak) for that ground truth label. By contrast, the CRF and MLR methods are required to match the ground truth label locations exactly for their predictions to be considered correct.

### 3.3.6 Evaluation Metrics

We evaluate the three morphology extraction methods described above using several different metrics. All of the results that we report are averaged over the test set performance of our six subjects and the standard error of the mean is also reported. The first metric we employ is average labeling accuracy over all six label types (P,Q,R,S,T,N). We also report confusion matrices where we list the fraction of each ground truth label that is predicted to be of each label type. This allows for a detailed analysis of the types of prediction errors that each method tends to make.

We are also interested in assessing the impact of morphology extraction accuracy on the computation of ECG morphological feature values. We use the distance between the Q and T waves as an example feature related to cocaine use. We assess the recall and precision of QT intervals as well as the error in the distance for recalled QT pairs. The recall is the number of complexes where the ground truth contained a QT pair and both Q and T peaks were predicted to be present, divided by the number of complexes where the ground truth contained a QT pair. The precision is the number of complexes where the ground truth contained a QT pair and both Q and T peaks were predicted to be present, divided by the number of complexes that were predicted to contain a QT pair. The error in the QT interval




(a) Labeling Accuracy (b) PUW (c) MLR (d) CRF

Figure 3.3: (a) Shows the average labeling accuracy for within-subject training. (b)-(d) show the corresponding confusion matrices for PUW, MLR and CRF.

is defined to be the absolute difference between the predicted QT interval (the distance between the predicted peaks) and the ground truth QT distance.

## 3.4 Results

In this section, we describe the results of our empirical evaluation including the within-subjects evaluation, between-subjects evaluation and transfer learning evaluation. Throughout this section, PUW refers to ECGPUWave using the WQRS detector, MLR refers to multinomial logistic regression, and CRF refers to our dynamic CRF framework.

### 3.4.1 Within-Subjects Evaluation

The results of the within-subjects evaluation as shown in Figure 3.3. Figure 3.3a shows the average prediction accuracy results for each of the three methods. We can see that the CRF and MLR methods both achieve the same average accuracy above $0.95$, while PUW performs substantially worse with an average accuracy of about $0.87$. The confusion matrices shown in Figures 3.3b-3.3d provide a more detailed look at the performance of the methods on a per-peak type basis. We can see that the prediction profiles for both the CRF and MLR models are nearly identical. We can also see that the distribution of errors for PUW is highly non-uniform. Consistent with past results for the WQRS detector, the PUW's identification of R peaks is highly accurate ($99\%$). However, performance for all

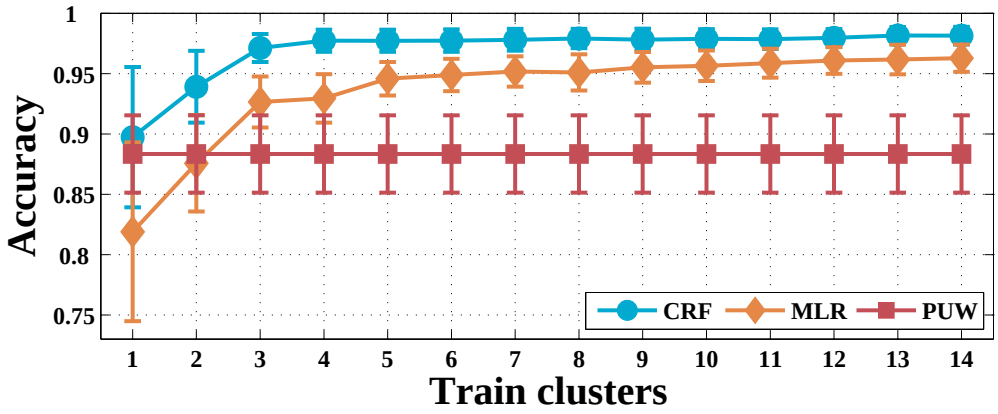
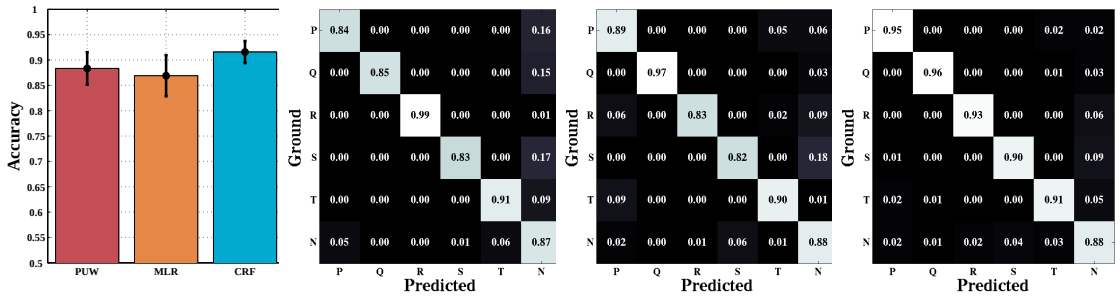



Figure 3.4: Shows average labeling accuracy as a function of number of training label clusters for within-subjects training

of the other peak types is much worse. In essentially all cases, this poor performance is caused by PUW failing to identify valid peaks, resulting in a prediction of $N$ (not a valid peak).

The fact that MLR and CRF have similar performance in the within-subjects case indicates that the feature representation provided by sparse coding is rich enough and the amount of data is large enough that there is no marginal benefit to structured prediction. However, the full within-subjects training protocol is based on hundreds of peak labels per subject. The need to label this much data for each individual subject is highly prohibitive. To assess the performance of the MLR and CRF methods given less data, we repeated the within-subjects evaluation while varying the number of labeled clusters available during training between $1$ and $14$ (each cluster contains $15$ labeled peaks on average). The results of this assessment are given in Figure 3.4. We can see that the performance of MLR and CRF are strongly differentiated in the more realistic low-data limit. With only one cluster of labels, the CRF still out-performs PUW on average, while MLR does not. We can also see that as more data become available, the CRF is able to improve its performance significantly faster than MLR.




(a) Labeling Accuracy (b) PUW (c) MLR (d) CRF

Figure 3.5: (a) Shows the average labeling accuracy for between-subject training. (b)-(d) show the corresponding confusion matrices for PUW, MLR and CRF.

### 3.4.2 Between-Subjects Evaluation

A natural alternative to learning ECG peak labeling models for each individual subject is to learn a model from an existing database of ECG peaks and apply that model to new subjects. The between-subjects evaluation assesses the performance of this approach when a model is learned using data from $5$ subjects and then evaluated on the $6^{th}$ held-out subject. We report results averaged over holding out each subject. Figure 3.5 gives the results of this assessment. We can see that both MLR and CRF suffer a decrease in performance relative to the full-data within subjects case. However, the CRF still out-performs PUW in the between-subjects setting while MLR performs worse on average. The confusion matrices show that MLR confuses a variety of similar wave types in this setting (P for T, R for P and T, T for P). The CRF makes similar types of errors, but to a reduced extent. This discrepancy can be explained by the fact that the CRF's transition parameters are able to exploit the regularity in the ordering of the waves within a cardiac cycle to compensate for feature parameters tuned for other subjects. By contrast, MLR only has access to features values. When there is a poor match between the shapes of the waves between-subjects, its performance thus degrades much more quickly.

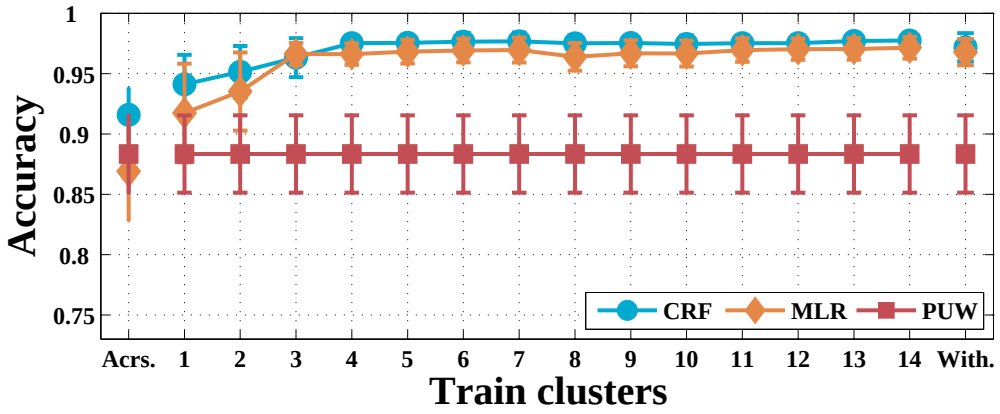


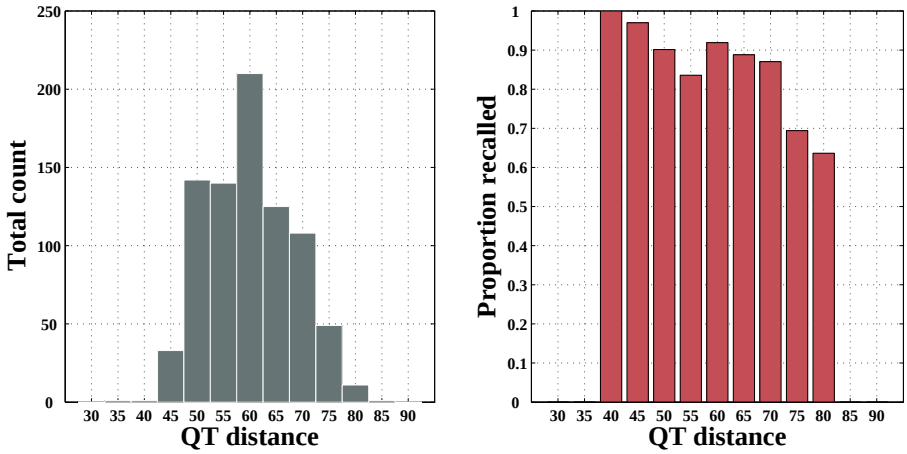

Figure 3.6: Shows average labeling accuracy as a function of number of training label clusters for transfer learning

### 3.4.3 Transfer Learning Evaluation

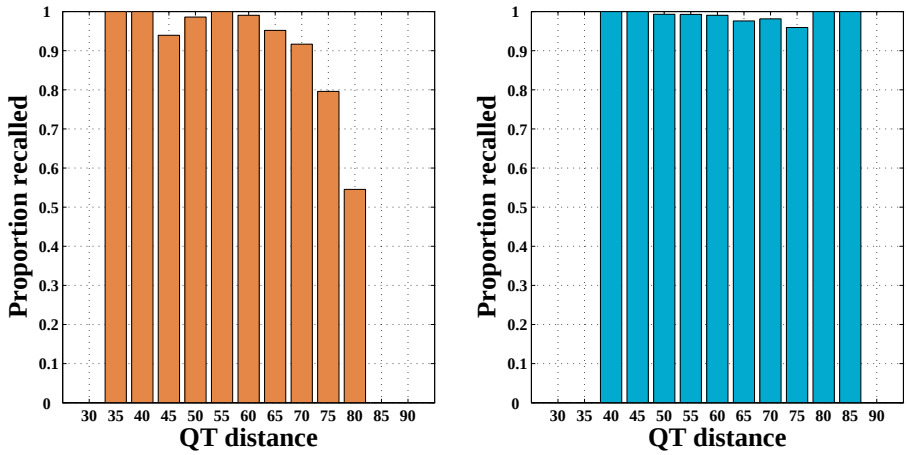
The drop in performance of MLR and CRF in the between-subjects setting motivates the evaluation of a third training protocol: transfer learning. Under the transfer learning approach we employ, (outlined in Section 3.3.3), data from other subjects is used to create a prior distribution over the model parameters. In the absence of any data for a given subject, the learned model falls back to the between-subjects model. As more data becomes available for an individual subject, transfer learning can smoothly interpolate between the between-subjects model and the within-subjects model. Figure 3.6 shows the results of this analysis. We can see that transfer learning is able to dramatically improve the performance of both MLR and the CRF in the low-data limit. With just one cluster of labels observed (approximately 16 labels), both MLR and CRF out-perform PUW and their corresponding between-subjects results.

### 3.4.4 QT Feature Extraction Evaluation

From the perspective of mHealth research, an important question is how differential accuracy in ECG peak labeling relates to the accuracy of ECG feature extraction. As a case study, we consider the problem of extracting QT distances from ECG data. The standard approach to this problem is to first identify the individual peak locations, and then compute




(a) Ground truth (b) PUW

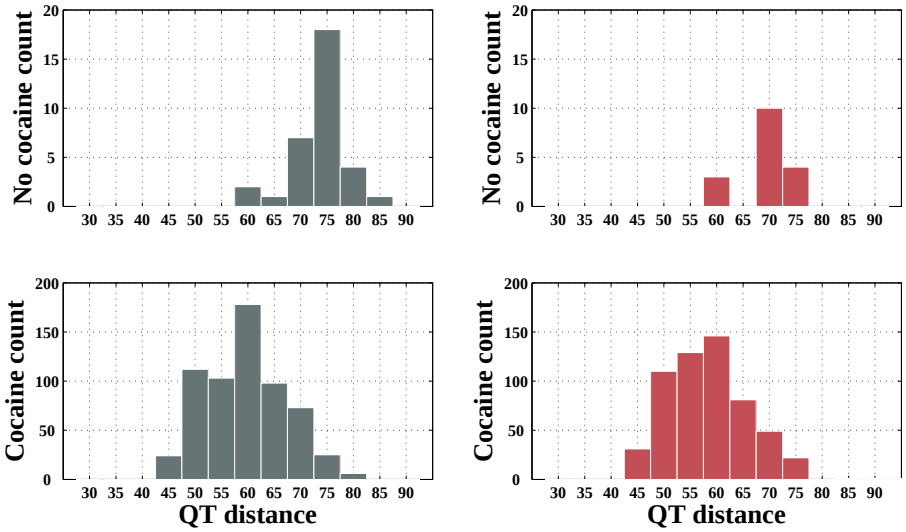



(c) MLR (d) CRF

Figure 3.7: (a) shows the ground truth distribution of QT distances over all data. (b)-(d) show recall rates as a function of ground truth QT distance for each method. These results show that PUW exhibits a strong differential recall rate as a function of the ground truth QT interval, while the CRF does not.

QT distances using the identified waves. The potential problem with this approach is that failure to predict either the Q or T peak results in the absence of a QT feature. Complexes for which feature values could not be extracted are typically discarded from subsequent analysis. However, this can lead to a systematic bias in the subsequent analysis if there is a relationship between the true value of a feature and the ability of a feature extraction

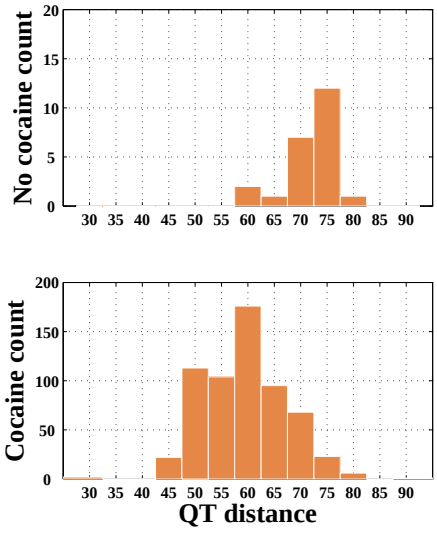
method to extract it reliably. This is essentially a non-random missing data problem in the sense of Little and Rubin [64].



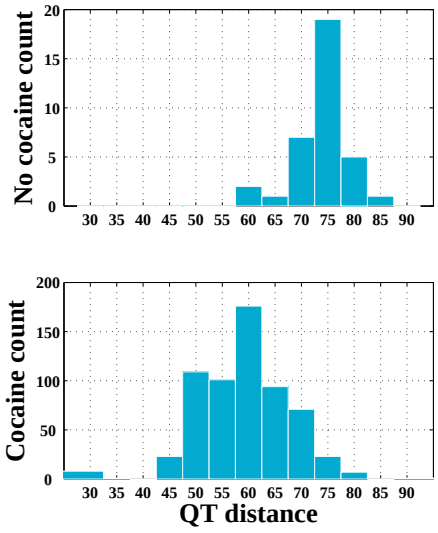

(a) Ground truth




(b) PUW




(c) MLR




(d) CRF

Figure 3.8: Distribution of QT distances for cocaine vs no cocaine. (a) shows ground truth QT distance distribution. (b)-(d) shows distributions of predicted QT intervals for PUW, MLR, and CRF.

| Model | Error | Recall | Precision |
|---|---|---|---|
| PUW | 8.5914±12.8231 | 0.8733 | 0.9689 |
| MLR | 0.8469±13.5030 | 0.9549 | 0.9912 |
| CRF | 1.9085±17.4729 | 0.9854 | 0.9830 |

Table 3.2: QT interval evaluation for PUW, MLR and CRF.

To assess the extent of this issue in our data, we used the MLR and CRF models trained using transfer learning with four clusters of labeled data to give a more realistic scenario for comparing subject-specific models to ECGPUWave. The results are summarized in Table 3.2. We can see that the lower accuracy of PUW results in significantly lower recall and precision of QT distances, as expected. We can also see that PUW has much higher mean error for the QT intervals that are retrieved than either MLR or PUW. Details of how we compute QT errors, precision and recall are explained in Section 3.3.6.

However, the interesting question is whether the recall rate for QT distances is uniform across all ground-truth QT distance values. Figure 3.7a shows the ground truth distribution of QT distances for our test data, pooled over all subjects. Figures 3.7b to 3.7d show the recall rate as a function of the ground truth QT distance (in bins of 5 samples). We can see that both PUW and MLR exhibit a strong differential recall rate as the ground truth QT distance increases. Only the CRF method achieves a nearly flat recall rate as a function of ground truth QT.

The final component of this case study looks at the distribution of QT values as a function of the study condition (cocaine vs no cocaine). Figure 3.8a presents the distribution of ground truth QT distances for both conditions pooled over all subjects. Figures 3.8b to 3.8d show the distribution of predicted QT distances for the complexes where both Q and T waves were identified. We can see that the CRF matches the ground truth distribution of QT distances quite closely for both the cocaine and no cocaine conditions as a result of its flat recall profile. On the other hand, PUW fails to identify any of the QT distances in bins $65, 80, 85$ under no cocaine and significantly skews the QT distribution in the presence of

cocaine. MLR also misses a large number of cases in bins $75, 80, 85$ under no cocaine, but performs well in the cocaine setting.

## 3.5 Related Work

The vast majority of past work on ECG morphology extraction has focused on QRS complex detection. Pan and Tompkins developed a widely used and widely cited QRS complex detection algorithm based on simple features of the ECG trace. Their approach achieves a QRS detection accuracy rate of $99.325\%$ on the MIT-BIH dataset [84]. However, systematic errors were noted in cases where the ECG signals contained stretches of noise, baseline shifts, unusual morphology and other artifacts. More recent work on QRS complex detection has focused on methods based on various transforms including the curve length transform [120] and the wavelet transform [69]. Both of these approaches give QRS complex identification precision and recall rates above $99.5\%$ on MIT-BIH dataset.

Other works on ECG morphology extraction first performs QRS detection followed by a local search procedure to identify individual waves [49, 69]. Research on atrial fibrillation has looked at extraction of specific morphological features from ECG. For example, [13, 14] uses QRS duration and PR interval to detect atrial fibrillation. A downside of these approaches is that a large number of threshold parameters are involved in the local search procedure. The method of Martinez et al. [69] for instance, depends on fifteen threshold parameters that are set by hand for an existing dataset such as MIT-BIH. More recent work has used supervised learning to select the set of scales used in wavelet decomposition [21].

The work of Hughes et al. [47] and de Lannoy et al. [23] treat morphology extraction as an ECG segmentation problem using hidden Markov models (HMMs). However, Hughes et al. specify the HMM directly over raw ECG samples and partially specify the transition structure by hand. De Lannoy et al. specify the HMM over coefficients of multiple mother wavelets and additionally make an assumption that all windows of ECG data start with a P

wave. Both approaches are forced to introduce self transition constraints into the model to counter the natural geometric distribution of self transition times inherent in an HMM.

CRFs have also been applied to ECG data previously, but for the problem of heartbeat classification [22]. In the work of de Lannoy et al. the CRF labels correspond to the beat type of each complete cardiac cycle. In fact, their work uses the method of Martinez et al. to extract morphological features [22]. We refer interested readers to [30] for a review of techniques and algorithms for ECG morphology extraction.

## 3.6 Conclusions

We started with the observation that there is domain knowledge in the cyclic patterns exhibited in many physiological signals. We hypothesized that leveraging this information will lead to improved performance on extracting low level features which in turn is used to detect high level target activities. We demonstrated the usefulness of this approach on one sample signal, ECG. We encoded domain knowledge via structured prediction models. We also demonstrated the usability of sparse coding to handle the inherent variance in shapes of ECG peaks. We evaluated the performance of these techniques on real world sensor data.

The structured prediction model resulted in a relative error reduction of $33\%$ when compared to both independent and baseline methods in a between user evaluation study. In order the minimize the manual labeling effort we also demonstrated transfer learning techniques which achieved the same performance as personalized models but with $77\%$ reduction in the number of supplied labels ($3$ vs. $13$ clusters of labels) on average. We also demonstrated that the CRF model introduces less systematic bias in regard to extracted features on downstream tasks when compared to baseline methods.

Inspired by this work, other researchers have used structured prediction models to encode domain knowledge when detecting target activities. The linear chain CRF model was used to label hand-to-mouth gestures in a smoking detection study [85] and was also used

to detect craving in smoking cessation studies [15]. In both studies, the structured prediction model performed better than independent and baseline methods. Even in the case of ECG morphology extraction, we observe improvement in performance when taking more long range dependencies into account when labeling ECG peaks. The first order Markov assumption in the linear chain CRF can restrictive when labeling ECG peaks especially, when the CRF model encounters a sequence of N's. The linear chain model looses track of the valid ECG peak preceding the N's. The context free grammar CRF (CRF-CFG) model leverages long range dependencies and obtains a $20\%$ relative error reduction when compared to the linear chain CRF model on the same ECG dataset as ours [75].

Another variant is a hierarchical CRF model that both labels and segments continuous streams of sensors data into high level activities [3]. This model further extends the idea of structured prediction to a second level of activity segmentation from streams of sensor data. More recently, the CRF-CFG model has been demonstrated to be useful in conversation detection using respiratory signals [8]. This followup research demonstrates the usefulness and generality of structured prediction models in the space of personalized health monitoring using wearables.

# CHAPTER 4

# MACHINE LEARNING PIPELINE FOR COCAINE USE DETECTION USING WEARABLE ECG SENSORS IN LAB SETTINGS

Presently, there are no FDA approved medications for cocaine addiction although research is underway for such drugs. Cognitive behavioral therapy (CBT) has been demonstrated to be effective in treating cocaine addiction [88]. CBT, among others, helps cocaine addicts to self-monitor to recognize cravings early and to identify contexts that increase chances of cocaine use. In general, drug users have a variety of reasons to not divulge details on drug use ranging from legal and social issues to self-denial and stigma surrounding drug addiction. Another major problem with self-reporting in this subject population is recall bias where a subject's retrospective recall of events differ from actual events [35]. Continuous monitoring of drug users provides critical information on user behaviour, cocaine use history, context surrounding cocaine use (*e.g., location, time, social interaction, visual cues, stress*) while minimizing the impact of recall bias. All this information is pertinent to effective CBT treatment for cocaine addiction. Hence, for continuous monitoring to be effective and useful, the first order of business is to reliably detect cocaine use in real world settings.

In this research, we use wearable sensors to detect cocaine use in real world settings since they are relatively easy to use, readily deployable, scalable and practical. We treat this as an instance of personalized health monitoring using wearables. However this is an extremely challenging problem since we have no prior evidence to demonstrate the feasibility of detecting cocaine use with wearable sensors. Hence we resort to first detecting cocaine use in more controlled clinical settings such as lab settings. The advantage is

two-fold: one, we isolate cocaine use activity from other confounding activities making it possible to obtain clean data; and two, we have more control over the design of experiments, which leads to high quality sensor data and associated labels. The research question we address in this chapter is can we reliably detect cocaine use in lab settings with wearable sensors?

The primary contributions of this chapter are to design and evaluate the feasibility of using wearable sensors to detect cocaine use in lab settings. We develop a cocaine use detection pipeline which includes data sensing and logging, feature extraction, feature aggregation and, lastly, cocaine use detection. We evaluate the usability of different ECG features in cocaine use detection and compare two approaches to feature aggregation over temporal windows. We evaluate the cocaine use detection pipeline on a novel cocaine use dataset gathered in the lab setting on habituated cocaine users.

The rest of this chapter is organized as follows. We first describe the experimental protocol used to gather data in the lab study (Section 4.1). We then describe our cocaine use detection pipeline (Section 4.2) and empirical protocols (Section 4.3) corresponding to our experiments (Section 4.4). We then present results (Section 4.5), review related works (Section 4.6) and present conclusions (Section 4.7).

## 4.1 Lab Study Protocol

As part of a National Institute on Drug Abuse (NIDA) approved study, we collected data from ten medically healthy, non-treatment seeking, experienced cocaine users. Subjects typically participate in the study for a two week period. All subjects reviewed and signed a consent form approved by Yale University's institutional review board. All participants were compensated monetarily for their time. This study was designed to isolate physiological responses to cocaine from other confounding activities. The study consists of multiple components that we describe below.

**1. Dry-Out Period:** When subjects are first admitted to the unit, they undergo a dry-out period to ensure that the acute influence of previous drug use does not affect the results of the study. All subjects undergo a dry-out period that lasts for several days.

**2. Cocaine Administration Session:** Subjects participate in a single 6-hour cocaine administration experiment comprised of a baseline session, three fixed-dose cocaine administration sessions and three cocaine self-administration sessions. These sessions appear in the same order for all subjects with mandatory breaks between them. The baseline session is conducted at the end of the dry-out period and immediately before cocaine administration. It provides physiological measurements in the complete absence of cocaine. The three fixed-dose sessions last 20 minutes each. At the start of each of the these three sessions, the subjects receive a single-bolus intravenous (IV) injection of cocaine. The three cocaine sessions use a fixed-order, ascending dose regimen of 8, 16, and 32 mg per 70kg respectively with a 100kg cap per adjusted dose. This procedure is based on extensive prior experience, which has shown these doses and procedures to be safe, well tolerated, valid, behaviorally relevant, and test-retest reliable [103].

The main purpose of the baseline and fixed-dose sessions is to assess subjects for participation in subsequent cocaine self-administration sessions. Physiological (ECG, respiration) and behavioral (visual analog scale) assessments are conducted at five-minute intervals throughout each session. An advanced cardiac life support certified research nurse and a basic life support certified research assistant are also present. Subjects who exhibit a heart rate greater than 160 beats per minute, diastolic blood pressure greater than 110 mmHg, systolic blood pressure greater than 180 mmHg, and/or have evidence of clinically significant cardiac ectopy, arrhythmia, or other dangerous symptoms are excluded from further self-administration sessions.

The fixed dose sessions are followed by three self-administration sessions which give subjects some control over the amount of cocaine they can receive. Each self-administration session uses one dosage level (8mg, 16mg or 32mg). The order of the dosage levels is ran-

domized and double blinded. The subject can click a button to receive an IV cocaine infusion at the given dosage level within each self-administration session. There is a minimum period of $5$ minutes enforced between subsequent infusions. All cocaine self-administration sessions take place at the Yale center for clinical investigation hospital research unit (YCCI-HRU). A saline lock, or peripheral intravenous device, is used for infusions of cocaine. Saline locks are maintained by trained research personnel in accordance with local, institutional policies and procedures.

**3. Physical Exercise Session:** In order to match the high heart rates experienced in the cocaine session, subjects were put through one (for some subjects two) physical exercise sessions. Eight of the ten subjects ran on the treadmill for twenty minutes with no or little resistance. Care was taken to ensure that median heart rates in the exercise session overlapped with the median heart rate in cocaine session for each subject. Two subjects went through a ping-pong session for the same duration.

**4. Smoking Session:**The goal here is to detect cocaine from yet another known confounder. It has been identified that nicotine causes acute changes in heart rate along with simulation to sympathetic nerve activity. ECG data collected from nicotine sessions followed a relaxed protocol were subject exit and re-entry time into clinical units were noted along with the number of cigarettes smoked. Only seven of the ten subjects participated in smoking sessions.

**5. Routine Activities:** In order to assess the subject's resting heart rate and physiological data in non-experimental settings, we gathered sensor data when subjects were performing day-to-day activities like watching television, sitting quietly, conversation, eating, etc.

## 4.2 Cocaine Detection Pipeline

In this section we describe our cocaine use detection pipeline on data gathered from ten subjects in the lab study. Our pipeline encompasses two levels of inference to analyze

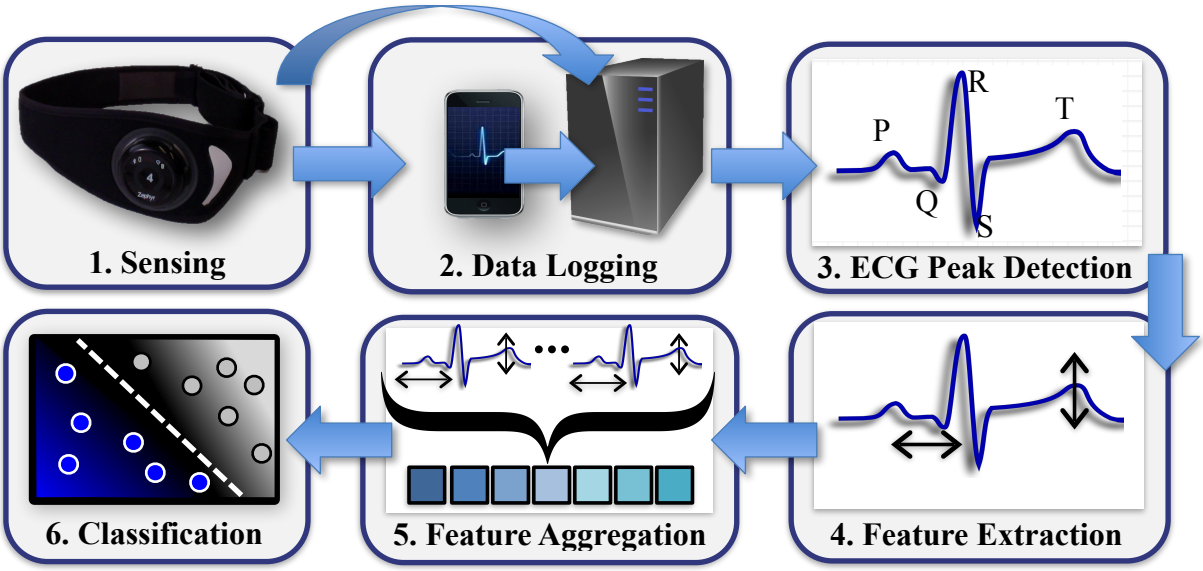


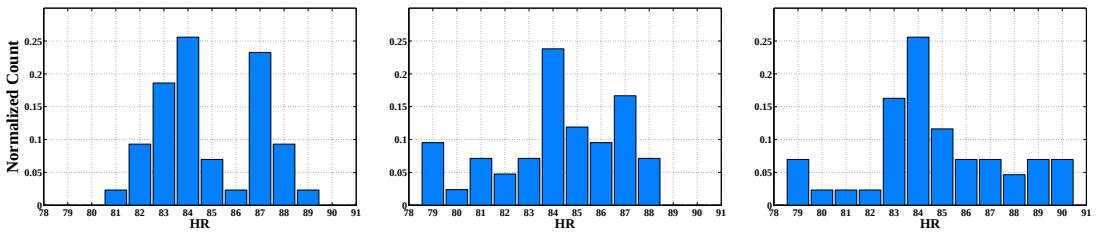

Figure 4.1: Data acquisition, processing and cocaine use detection in lab settings

raw ECG data to detect cocaine use. At the first level, we extract ECG morphology from raw sensor data using techniques described in Chapter 3. At the second level, we perform feature aggregation to explicitly take into account noise in the ECG morphology extraction process. We acknowledge that this two step process may be sub-optimal as domain knowledge can be incorporated to create a single framework to perform multilevel inference simultaneously as in [3]. We leave this to future work. We first describe the on-body sensor system, followed by feature extraction and finally the cocaine use detection model.

### 4.2.1 Sensing and Data Logging

During the lab protocol, the subjects wore a Zephyr Bioharness 3 chest band [117] which provides raw ECG data, chest band diameter, accelerometer and derived data such as heart rate and respiratory rate. These chest bands are designed to be comfortable and less intrusive to wear than Holter monitors. This sensor samples ECG data at 250 Hz and has sufficient memory and battery life for 24 hours.

Our system encompasses two levels of data logging. The first level is on the sensor itself. The second is on a Samsung Galaxy smartphone that is paired to the chest band sensor via bluetooth. The data on the sensor is downloaded at the end of each day and




(a) (b) (c)

Figure 4.2: Distribution of heart rates in three 30 second windows. All three windows have an average heart rate of 85bpm with heart rate variability of 2beats

uploaded to a secure server. The sensor also transmits summary packets to the phone, which are then periodically transmitted wirelessly to the secure server. The on-body sensor system is illustrated in steps 1 and 2 in Figure 4.1,

### 4.2.2 ECG Peak Detection

We follow the pre-processing steps and ECG peak detection approach as described in Chapter 3. ECG peak detection is illustrated in Step 3 in Figure 4.1. Since there is substantial variability in size and shapes of ECG peaks between subjects, we build the sparse coding dictionary and CRF model per subject in the lab dataset.

### 4.2.3 ECG Feature Extraction

We extract ECG features from ECG peak locations. ECG feature extraction is illustrated in Step 4 in Figure 4.1. Ideally, we would like to extract ECG features, for instance QT interval, within each ECG cardiac cycle. To do so, we pair P, Q, S and T peaks to the associated R peak thus grouping ECG peaks into individual cardiac cycles. Motivated by prior studies on effects of cocaine on ECG (Section 2.2.3) we extract six features per cardiac cycle: RR interval, QT interval, QTc[1] (corrected QT), QS interval, PR interval, and T-wave height.

[1]Bazett's correction

### 4.2.4 ECG Feature Aggregation

Raw ECG data is sampled at $250$Hz, but changes in ECG morphology as induced by cocaine happens at much lower frequency (it typically varies from $45$ minutes to $3$ hours depending on quantity and form of intake, metabolism and habituation). Hence, there is a temporal mismatch in the rate of arrival of sensor data and the rate at which we would like to make decisions on cocaine use. Additionally, features extracted from ECG morphology are susceptible to noise in the morphology extraction process which in turn affects down stream task of cocaine use detection. For example, typical cardiac cycles are made of P, Q, R, S and T waves, but due to noise or other artifacts a spurious peak may be mislabeled as a valid ECG peak or a valid ECG peak may not be labeled at all. Hence features computed using ECG morphology need to explicitly take into account noise in ECG morphology extraction process.

We perform feature aggregation to mitigate the effect of potentially noisy ECG features as well as to detect cocaine use over a given decision making window. Typical approaches to feature aggregation are computing the mean and standard deviation, which captures the location and shape of the underlying distribution. It is well known that both these statistics are sensitive to outliers and perform poorly when the underlying distribution is non-Gaussian or multi modal. As an illustration, we plot the distributions of heart rates in three $30$ second decision making windows in Figure 4.2. The three windows are chosen such that the mean heart rate is $85$ bpm and the heart rate variability is $2$ beats. Despite the fact that these three windows have identical mean and standard deviation, the underlying distributions are starkly different.

Our approach to feature aggregation is to build $1$D histograms of extracted features over decision making windows. Building histograms is illustrated in Step $5$ in Figure 4.1. The use of histogram-based features for ECG is inspired by success in using these features in computer vision. Histogram of Oriented Gradient (HoG) features have been demonstrated to be successful in many computer vision tasks like face detection [24] and pedestrian

detection [20]. While HoG features are described in the spatial domain, our 1D histogram of ECG features are described over sliding time windows of ECG data. This approach is akin to a non-linear transformation of features. These 1D histograms capture properties of the feature distribution such as multiple modes by distributing its mass over multiple bins. The 1D histograms also naturally handle outliers by placing them in the extreme histogram bins (since theses features are farther away from the mean) while averaging explicitly takes outliers into account.

### 4.2.5 Classification

The final stage of our pipeline is detection of cocaine use, as illustrated in Step 6 in Figure 4.1. Given features aggregated from a sliding windows, we view the problem of constructing a detector for cocaine use as a standard binary classifier learning problem. We treat data from the self-administration session as positive instances of cocaine use and all other activities as negative instances. Each data case consists of a feature vector, $x \in \mathbb{R}^D$, of aggregated features and a corresponding class label, $y \in \{-1, +1\}$, indicating which of the two classes the data case belongs to. We utilize penalized logistic regression as discussed in Section 2.1.1.

## 4.3 Empirical Protocols

In this section, we describe the empirical protocols including how we partitioned the data and evaluated cocaine use detection in the lab setting.

### 4.3.1 Cocaine and Non-cocaine Activities

For these experiments, we used ECG data from all activities in the lab protocol. ECG data from breaks in the self-administration session were also considered instances of cocaine use since cocaine has a half life of roughly $\sim$ 45 minutes. For the seven subjects that participated in the smoking activity, we retained ECG data from the middle one-third

of each smoking session only as the first and last one-third of ECG data included walking to/from the smoking lounge.

### 4.3.2 ECG Morphological Features and Feature Aggregation

We experimented with six ECG features RR, QT, QTc, QS, PR and TH. We also experimented with two feature groupings, ALL – all six features combined together and ALL-RR –all features combined together expect RR. In total, we experimented with eight feature sets. We experimented with two types of feature aggregation techniques: the proposed histogram-based feature aggregation and, standard summary statistics such as mean and standard deviation. The purpose of two different feature aggregations is to compare and contrast traditional methods with our histogram approach. Both feature aggregations were performed on the one minute sliding windows. We experimented with different sliding window lengths ranging from $30$ seconds to $7$ minutes. We observed that the trends were roughly similar with no significant difference between different window sizes for different feature groupings. We present this analysis in the results section. For the purposes of providing enough data samples in both positive and negative classes we present results from one minute sliding windows with zero overlap.

In order to build histogram-based features, we also require the number of histogram bins (or alternately the bin boundaries). In our experiments, we observed computing histogram over four bins to be robust to noise typically found in ECG sensor data. Hence for each subject we choose bin boundaries: $\big\{\big[$minimum value-$33^{rd}$ percentile$\big]$, $\big[34^{th} - 50^{th}$ percentile$\big]$, $\big[51^{st} - 66^{th}$ percentile$\big]$, $\big[67^{th}$ percentile-maximum value$\big]\big\}$. These boundaries were computed per ECG feature (RR, QT interval, etc) on data collapsed from all activities (cocaine, physical exercise, etc) within each subject. These boundaries result in four bins per ECG feature with a total of $24$ features per sliding window. To avoid absolute counts from influencing downstream tasks we normalize histogram counts over bins per sliding window such that they sum to one.

### 4.3.3 Evaluation protocols

We perform both within-subjects and between-subjects evaluation. In the within-subject experiments, we partition the available data from both cocaine and non-cocaine activities into two temporally preserved halves. The training data corresponds to the first half of each session (cocaine, physical exercise, etc) and the test data corresponds to second half. This same partition was preserved within the fixed and self-administration cocaine sessions as well. We resorted to this partition to simulate real-world scenarios and to retain time correlated samples in the train/test respectively. For the between-subjects evaluation, we train the cocaine use detection model on $M - 1$ subjects and test on the held out $M^{th}$ subject (*i.e. a leave-one-user-out protocol*). We repeat the same protocol for all ten subjects.

### 4.3.4 Cocaine Detection Model

In both the within and between-subjects case, we train and test one cocaine use detection model per subject. We use penalized logistic regression as described in Section 2.1.1. For the within-subjects case, we perform hyperparameter selection by performing a $5$-fold cross validation on the train set. For the between-subjects case we perform another leave-one-user-out cross validation on $M - 1$ subjects to choose hyperparameter.

### 4.3.5 Evaluation Metrics

For both within and between-subjects analyses, we report the mean area under ROC curve (AUROC) along with standard error bars over ten subjects.

## 4.4 Dataset

In Table 4.1, we report the number of data cases in each activity following feature aggregation. Each data case corresponds to a one minute sliding window with no overlap. We have only considered sliding windows in which all six ECG features could be reliably extracted. We observe there are twice the number of data cases in the cocaine activity when compared to all other activities put together. This imbalance in sample count is the

consequence of experiment design which tends to focus on rare, target activities such as cocaine use to build reliable, robust detectors.

| Subject | Age | Sex | Cocaine session | Baseline session | Physical exercise | Routine activities | Smoking session |
|---|---|---|---|---|---|---|---|
| 1 | 49 | F | 355 | 34 | 40 | 30 | 122 |
| 2 | 46 | M | 336 | 31 | 19 | 30 | 74 |
| 3 | 44 | M | 247 | 36 | 38 | 28 | 121 |
| 4 | 46 | M | 350 | 24 | 40 | 30 | 78 |
| 5 | 42 | M | 355 | 35 | 40 | 30 | 107 |
| 6 | 36 | M | 175 | 29 | 20 | 30 | 78 |
| 7 | 49 | M | 258 | 23 | 20 | 30 | 43 |
| 8 | 30 | F | 94 | 44 | 22 | 30 | – |
| 9 | 46 | M | 333 | 20 | 15 | 29 | – |
| 10 | 49 | M | 440 | 34 | 20 | 30 | – |
| Total | – | – | 2943 | 310 | 274 | 297 | 623 |

Table 4.1: Number of data cases (one minute windows) per subject for cocaine, baseline, physical exercise, routine activities and smoking activities

## 4.5 Results

In this section we present results of both the within and between-subjects evaluation for all eight feature sets and two feature aggregation techniques.

### 4.5.1 Within-subject Cocaine Detection

While training a classifier for each individual user is clearly not practical, studying within-subject classification sheds light on which features work best if we ignore between-subject variability induced by habituation and cardiac response to cocaine. We report the mean AUROC as well as the standard error of the mean in Figure 4.3 for each feature set and feature aggregation technique.

On the x-axis are different feature sets and on the y-axis is AUROC. The first observation is that all feature sets perform with AUROC $>$0.5, which is above chance. We observe that using all features performs the best with an AUROC of $0.86$ when compared to using any one feature in isolation. This is followed by AUROC's of PR interval, QTc and RR

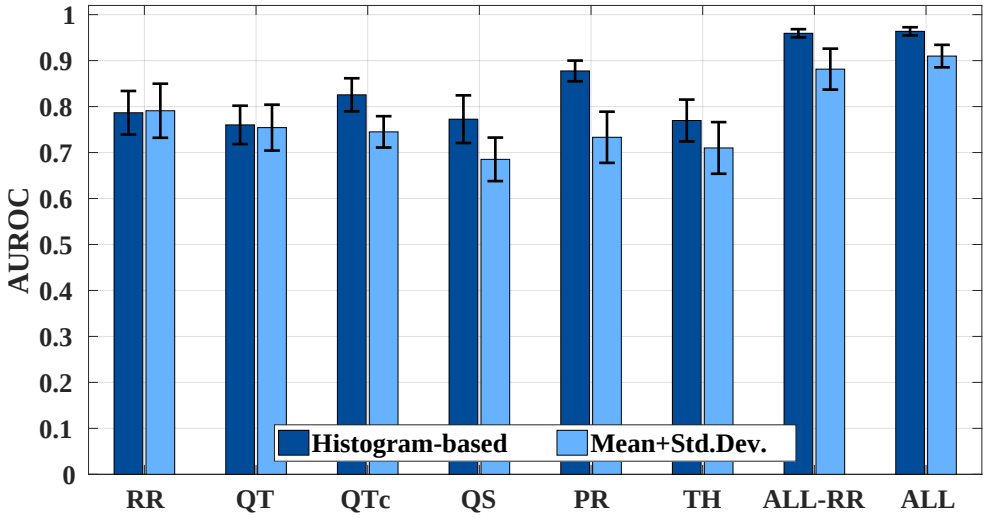



Figure 4.3: Mean within-subject AUROC over ten subjects along with standard error bars for seven features and two feature aggregation techniques

interval at $0.83$, $0.78$ and $0.76$ respectively. The differences in AUROC's between the top three features are not statistically significant as evidenced by the overlapping error bars.

In terms of feature aggregation techniques, we observe that six of the eight features using histogram-based aggregation performed as well as (or better) than summary statistics-based feature aggregation, but this difference is not statistically significant. It is also worth noting that the two feature sets for which the histogram-based feature aggregation performed worse than summery statistics are both heart rate influenced features (R and QT). It is well known that cocaine causes an increase in heart rate leading to good separability between cocaine and non-cocaine data cases. Additionally, it is relatively to easy to identify and extract feature related to RR interval when compared to extracting features associated to morphological changes in ECG.

Before moving on to the between-subjects case, we pause to consider the usefulness of RR interval as a basis for cocaine use detection outside of the clinical setting. While the RR interval has reasonable performance in the clinical setting, it is obviously confounded by any other activity that results in an increase of heart rate. The fact that other ECG features, such as QTc, yield better performance while completely removing the effect of

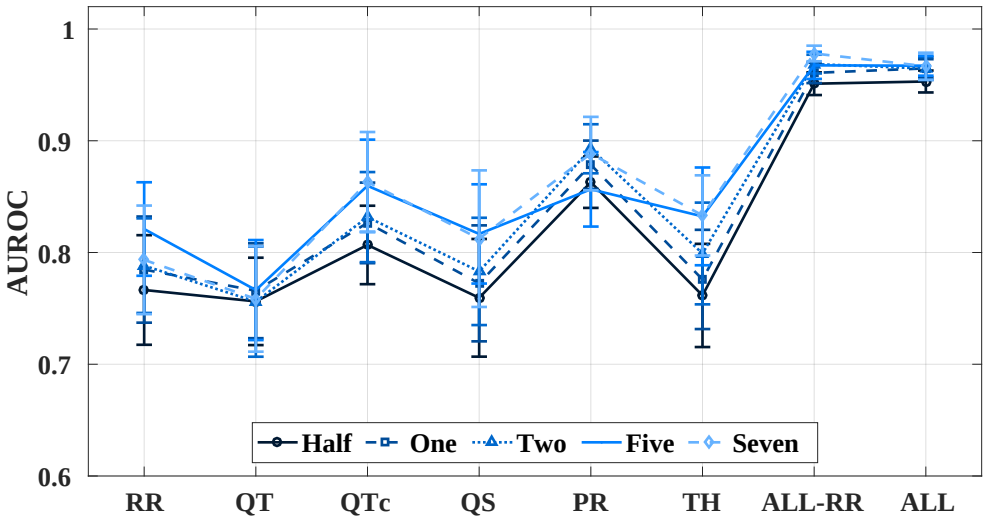



Figure 4.4: Effect of different sliding windows for feature aggregation

heart rate implies that there are significant morphological changes in ECG in the presence and absence of cocaine.

Lastly, we analyze the effect of different sliding window sizes. In Figure 4.4 we plot the within user performance using histogram based feature aggregation technique for different sliding windows. We experimented with windows ranging from 30 seconds to seven minutes. We observe that the trends for different window sizes are similar to the within user performance but exhibit strong overlap as evidenced by overlapping error bars. The performance is almost identical for ALL and ALL-RR features when compared to individual features. This suggests that when concatenating all features, the signal to noise ratio roughly remains the same for different sliding windows. We also point out that as window size grows, there are fewer data examples to train and test the classifier. This reduction in sample size is reflected in the plot as performance corresponding to windows of size seven performs better, on average, than windows of size 30 seconds.

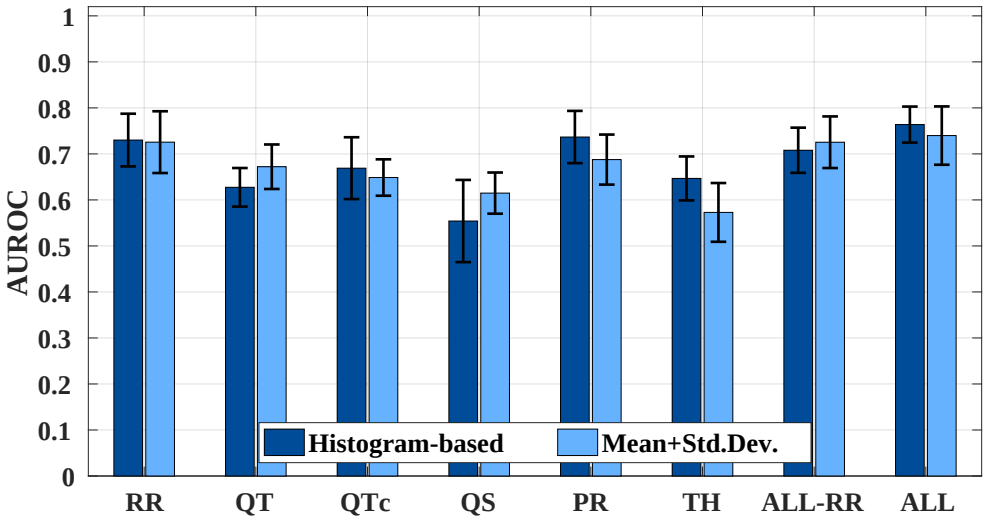



Figure 4.5: Mean between-subject AUROC over ten subjects along with standard error bars for seven features and two feature aggregation techniques

### 4.5.2 Between-subject Cocaine Detection

We now turn to between-subjects cocaine use detection. We report the mean AUROC as well as the standard error of the mean in Figure 4.5 for each feature set and both feature aggregation techniques.

Ignoring the RR interval's performance since its use outside of clinical settings is limited, we observe similar trends to that of the within-subject's case, but there is an overall reduction in AUROC across all features and aggregation techniques. This reduction is expected given the between-subject variability in the relationship between ECG and cocaine use. We observe that the best AUROC is at $0.76$ when using histogram-based ALL features. We observe that histogram-based features do not perform very well compared to summary statistic-based feature aggregation since the histogram bin boundaries, which are computed per subject, do not align very well. This causes some features in some subjects to concentrate in some bins, while it causes the same features in other subjects to concentrate in other bins which directly affects generalizability.

## 4.6 Related Work

Following our work, wearable ECG has been used to detect cocaine use in both lab and field settings [43]. Hossain et al., used heart rate and accelerometer data as features to isolate cocaine use events from other confounding activities. Central to their approach is the dynamics of the autonomic nervous system (ANS) to detect cocaine use events. Specifically, an increase in heart rate is associated to the activation of the sympathetic branch of the ANS. However increase in heart rate can be caused by several confounding activities such as physical exercise, fear, stress, etc. This paper makes the crucial observation that the parasympathetic branch handles heart rate recovery differently for cocaine and non-cocaine events. The authors train a prediction model to label candidate windows as either belonging to cocaine-free physical activity, activity-free cocaine use, or neither of these classes.

In order to minimize false positives, they perform classification only on candidate windows that are likely to have cocaine use events. Their criterion for selection of candidate windows includes a combination of heuristics, change point detection algorithms on instantaneous heart rates, urine tests and accelerometer data to screen out physical activity. Model parameters are tuned on a lab dataset and evaluation is performed on field data. On a field dataset of 27 cocaine use events spread over 25 days their model has a true positive rate of $100\%$ and a false positive of $1.13$/day.

This study differs from ours in four important ways,

1. Hossain et al., treat the subjects' self-reported drug intake event timestamps in field study as ground truth despite the fact that they are of unknown quality

2. This study uses heart rate and accelerometer data as features to isolate cocaine use events from other confounding activities while we use ECG morphology only

3. Their prediction model localizes in time cocaine use events (referred to as fine-grained predictions) and in our approach we extract features from these fine-grained predictions to predict urine test outcome (referred to as coarse-grained predictions)

4. The authors do not report any systematic differences between their lab and field datasets (we observed systematic differences in our datasets which is discussed in Chapter 5) obviating the need to perform any domain adaptation when compared to datasets gathered in our study.

## 4.7 Conclusions

In this chapter, we developed a machine learning pipeline to detect cocaine use from non-cocaine activities in the lab setting. Our pipeline consisted of data sensing and logging, feature extraction and aggregation, and cocaine use detection. We compared multiple ECG feature sets gathered from data in a novel cocaine use detection study on ten habituated cocaine users. In both within and between user evaluation protocols, all ECG feature sets perform above AUROC $0.5$, which is better than chance guessing. We observed that concatenating all features performs much better than any feature separately with a best AUROC of $0.95$ (within-user) and $0.75$ (between-user) respectively. We point out that this was the first work to demonstrate the feasability of using wearable sensors to detect cocaine use in lab settings.

Heart rate and heart rate variability are two most extensively used features in health monitoring using wearables. We wanted to comment on our experience in the use of these features in design of experiments and target activity detection pipelines. We observed that cocaine causes an increase in heart rate, but so do other confounding activities like physical exercise and stress. In our experiments we observed that the heart rate ranges for different activities had less overlap, leading to easy detection of cocaine use from non cocaine activties. Our initial approach was to create a balanced dataset by selecting positive (cocaine use windows) and negative examples (non-cocaine activity windows) matched by heart rate. This approach led to throwing away many data examples since we did could not find matching heart rates, which seemed wasteful. We redesigned our data collection protocols such that non-cocaine activties such as physical exercise exhibited an overlap in heart rate

with heart rates from cocaine use for each subject respectively. This led to a better validation of our cocaine use detection pipeline. This insight is crucial to designing experiments when relying on heart rate or heart rate influenced features to detect target activities.

# CHAPTER 5

# DOMAIN ADAPTATION TECHNIQUES TO IMPROVE LAB-TO-FIELD GENERALIZABILITY IN COCAINE USE DETECTION

In the last chapter, we demonstrated the feasibility of using wearable sensors to detect cocaine use in lab setting. However, it is clear that many aspects of these lab-based data collection procedures have poor ecological validity. When activities are scripted or controlled, the proportion of time subjects spend performing target activities (including cocaine intake) will be significantly distorted. The way that subjects consume cocaine under scripted and controlled conditions also may not be representative of their behavior in the real world settings. Indeed, data collected under controlled lab conditions typically encompass a very limited number of the different contexts relative the real world settings. These factors can lead to significant differences between the distribution of features extracted from wearable sensors in the lab and the field. We refer to real world settings as field settings in this chapter. Additionally, the groups of subjects that participate in lab and field cocaine studies are typically different, leading to a further loss in performance when there is significant between subject variability in any aspect of behavior.

Another persistent problem in lab-to-field generalization is the mismatch in the techniques employed to gather ground truth activity labels. In our cocaine study, the ground truth data available in the lab is often fine-grained, including precise start and end times. In the field, subjects are often asked to self report cocaine use, but these self reports are known to be unreliable. Instead, cocaine use studies typically rely on urine toxicology (utox) tests as a gold standard for establishing cocaine use within a specified time period (*i.e. the prior 24 hours*). However, utox testing alone can not localize the exact time intervals correspond-

ing to cocaine use. Hence, in cocaine use detection, the ground-truth labels available in the lab are typically not available at the same level of temporal granularity in the field.

In summary, differences in experiment design, data collection and subject populations gives rise to systematic differences in cocaine use datasets gathered in lab and field settings. Despite these differences, we would like to deploy the lab-based cocaine use detection model to detect cocaine use in field settings. Directly deploying a lab-based cocaine detection model in field settings will lead to poor generalization performance. The research question we address in this chapter is how can we generalize a cocaine use detection model developed in lab setting to field settings.

The primary contributions of this chapter are, we identify prior probability shift, which results from different class distributions at train and test time, as a factor that affects lab-to-field generalizability for cocaine use detection. We present methodology to assess and evaluate domain adaptation techniques for mitigating prior probability shift. We identify covariate shift, which results from differences in the distribution of features at train and test time, as a factor that affects lab-to-field generalizability for cocaine use detection. We present methodology to assess and evaluate domain adaptation techniques for mitigating covariate shift. We identify label granularity shift, a problem we define as the result of changes in the temporal granularity of labels across source and target domains. We develop domain adaptation techniques to handle label granularity shift. To the best of our knowledge, this last problem has not been addressed before in the context of personalized health monitoring using wearables. We note that between-subjects variability is not a distinct factor, but can be a contributor to both prior probability shift and covariate shift.

This chapter begins by describing the experimental protocol used to gather data in the field study (Section 5.1). We compare and contrast the field dataset with the lab dataset in Section 5.2. We then describe three factors that directly affect deploying a lab-based cocaine use detection model on field data (Section 5.3). This is followed by a description of our approach to mitigating the effects of these three factors (Section 5.4). Lastly, we

present results on cocaine use detection in field data (Section 5.6), review related work (Section 5.7) and present conclusions (Section 5.8).

## 5.1 Field Study Protocol

As part of the same NIDA approved study, we collected data from five medically healthy, non-treatment seeking, experienced cocaine users in their natural environments while performing day-to-day activities. Subjects participated in the study for a period of eleven days. All subjects reviewed and signed a consent form approved by the local institutional review board. All participants were compensated monetarily for their time.

On the first day of the study (the habituation day), the recruited subjects were briefed on the usage, upkeep and maintenance of devices. We used the same sensors and data logging procedures as described in our lab study in Section 4.2.1. The study involved 10 clinical visits including the habituation day visit. Clinical visits were not conducted on weekends and other holidays. During the course of the study, participants were instructed to wear the sensor continuously while they performed day-to-day activities (except while showering). During each clinical visit, subjects met with the experimenters to provide urine samples, download data and swap recharged devices. Subjects reported periods of cocaine use along with the monetary value of cocaine used. This information was entered on the subject's cellphone using an ecological momentary assessment (EMA) protocol. These entries were verified by the experimenter as part of compliance with the study protocol. In this field study, the subjects were not asked to report on any activity other than cocaine use.

## 5.2 Field Dataset

In Table 5.1, we report summary statistics of the field dataset. For the purpose of the field study, we give the self-reported time spent on cocaine use activities and assume that time not self-reported as cocaine related activities corresponds to non-cocaine activities. The study resulted in a total of 37 days of field data (data from some weekend days was not

| Dataset | # Subjects | Mean age | Cocaine use | Non-cocaine activities |
|---|---|---|---|---|
| Field Study | 5 | $46.8 \pm 3$ | 151h 46m | 739h 25m |
| Lab Study | 10 | $43.7 \pm 6$ | 56h 59m | 29h 23m |

Table 5.1: Total number of hours of cocaine use and non-cocaine activities over all subjects in field and lab datasets respectively. Field statistics related to time of cocaine use are based on self report.

captured due to devices running out of power between visits to the study coordinator). For comparision purposes, we also report summary statistics for the lab dataset.

For each field day, we perform ECG peak detection, feature extraction and feature aggregation as described in Chapter 4. We computed histogram-based features on five minute sliding windows with zero overlap. One reason for using longer time windows is that we observed subjects consumed relatively larger quantities of cocaine leading to longer durations of cocaine related metabolism. Ideally, we would like to predict the presence of cocaine in each sliding window. By using longer but fewer time windows we hope to minimize the number of false positives by accumulating more evidence. The bin boundaries for histogram-based feature aggregation were computed using data from the lab study only. Specifically, we computed bin boundaries by collapsing all sessions from all lab subjects into one lab set and computed the bin boundaries on this lab set.

## 5.3 Factors Limiting Lab-To-Field Generalization

In this section, we describe three factors that can have a significant impact on lab-to-field generalization performance and discuss how they can be assessed given samples of data from the lab and from the field. Here data samples, in both lab and field datasets, refers to ALL features (from Chapter 4) using histogram-based feature aggregation in five minute sliding windows.

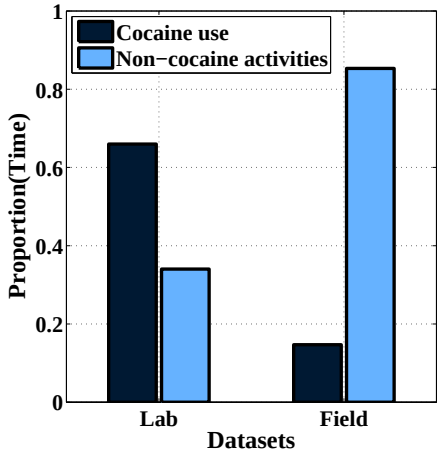
### 5.3.1 Prior Probability Shift

During the lab-based component of our study, each subject spent roughly the same amount of time performing various activities, and we have access to precise timestamps corresponding to periods of cocaine use and non-cocaine activities. During field-based data collection, subjects self-reported (via EMA's) consuming cocaine for a small fraction of the total time. The difference in the amount of time subjects spend performing various activities in the lab and field environments results in prior probability shift. Prior probability shift is defined as a systematic difference in the label proportions present in train and test datasets. The likelihood of significant prior probability shift increases as the ecological validity of lab-based data collection decreases.

The severity of prior probability shift can be easily characterized in terms of the difference between the proportion of labels of each type in the lab and in the field. In our study, the base inference of interest is the prediction of cocaine use over five minute windows, so the degree of prior probability shift is directly reflected in the proportion of time that subjects spend consuming cocaine. In Figure 5.1a, we summarize the lab and field datasets in terms of the amount of time subjects spend on cocaine use versus non-cocaine activities. As expected, a smaller fraction of time is spent on cocaine use in the field setting (about $17\%$), while the lab-based data collection protocol significantly over-represents the proportion of time spent on cocaine use (about $66\%$).

### 5.3.2 Covariate Shift

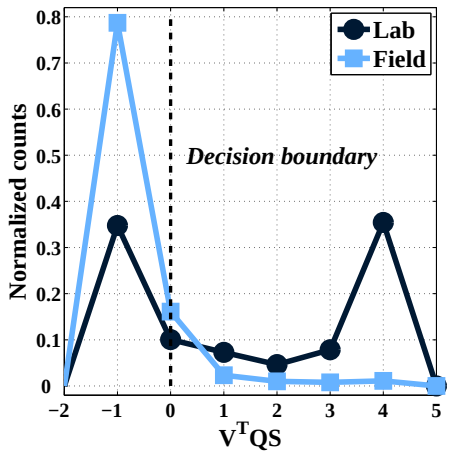
Cocaine administration in the lab-based component of our study was restricted to one day when subjects were administered cocaine intravenously while not performing any other activities. Non-cocaine activities were scripted and performed by subjects in a very limited number of contexts that are not representative of the complexity of natural field environments. However, performing cocaine and non-cocaine activities in new contexts can result in significant changes in the per-class feature distributions. This problem is referred to



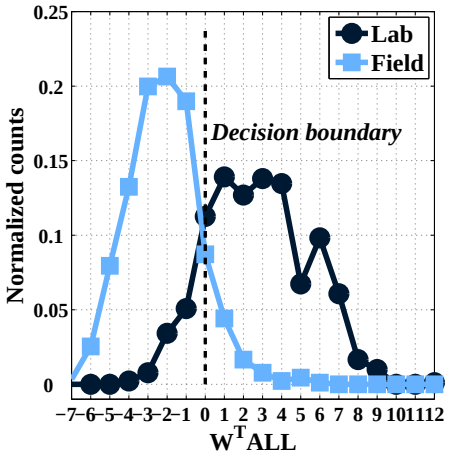

(a)

| Features | Accuracy |
|---|---|
| RR interval | 0.57 ± 0.0066 |
| QT distance | 0.52 ± 0.0108 |
| QTc distance | 0.67 ± 0.0107 |
| QS distance | 0.75 ± 0.0087 |
| PR distance | 0.64 ± 0.0081 |
| T-wave height | 0.52 ± 0.0088 |
| ALL features | 0.87 ± 0.0067 |

(b)




(c)




(d)

Figure 5.1: (a) Proportion of time spent on cocaine and non-cocaine activities in lab and field environments respectively. Quantifying covariate shift between lab and field datasets: (b) Mean accuracy ± standard error for the task of discriminating lab data from field data. Distribution of lab and field classifier scores for (c) QS feature and (d) all features

as covariate shift. Covariate shift is defined as a systematic difference between the feature distributions contained in training and test datasets. There is an increased possibility of significant covariate shift when moving from lab-based training data to field-based test data.

The severity of covariate shift can be assessed by comparing the distribution of features in lab and field data. Simple histograms can reveal the presence of significant covariate

shift when they have an effect on the marginal distributions of the features. The effects of covariate shift may be more subtle, affecting the joint distribution of features while leaving the univariate marginal distributions mostly invariant. This scenario can be assessed by drawing equal sized samples of lab and field data, and fitting a classification model that aims to discriminate the data collected in the lab from the data collected in the field. If the two distributions coincide, the expected accuracy achieved on this task will be $50\%$. As the feature distributions diverge, the classification accuracy will increase toward $100\%$.

In Figure 5.1b, we report the classification accuracy for discriminating lab versus field data for a variety of ECG-based features used for cocaine use detection. We assess the classification ability of these features when used individually and when they are used in combination. The model used is $l_2$ regularized logistic regression (details in Section 2.1.1) with hyper-parameters set via 10-fold crossvalidation. We see that all accuracies are greater than $0.5$, suggesting the presence of covariate shift.

Among the individual features, the QS interval obtains the best accuracy indicating that it carries the most information with respect to the task of discriminating lab data from field data. In Figure 5.1c, we show histograms of the QS classifier score function values when applied to the lab and field datasets. If $\mathbf{v}$ and $v_0$ are the optimal weight vector and bias parameters learned for a logistic regression model, then the classifier score function is simply $v_0 + \mathbf{v}^T\mathbf{x}$ (see Equation 2.2 for details). For single features, the score function value is a scaled and shifted version of the raw feature value, so Figure 5.1c reflects the class conditional QS distributions for the lab and field datasets. We can see that the score function values are fairly distinct, with particularly low overlap for high values of the score function.

In Figure 5.1d, we show histograms of the logistic regression score function values for the lab and field datasets when training using all features. In this case, the score function is a linear combination of all of the feature values. We can see that there is substantially less overlap between the score function values when using all features, which is consistent with

the increase in classification accuracy when using all features. This is strong evidence for a significant multivariate covariate shift effect between the lab and field datasets. However, it also shows that the lab and field feature distributions are not completely disjoint. As we will see, the presence of some overlap is required for the application of instance weighting methods to correct for covariate shift.

### 5.3.3 Label Granularity Shift

In the lab setting, subjects were closely monitored, and the precise times and amounts of cocaine consumed are all known exactly. In the field, subjects self-reported periods of cocaine use as well as the dollar amount of the cocaine consumed. However, for this subject population, self-reports of the activity of interest can be quite unreliable. We present evidence of unreliable self-reporting in Table 5.2. To obtain a measurement that can be considered ground truth for whether subjects consumed cocaine on a given day, urine samples were collected during each visit for the duration of the study. A semi-quantitative urine toxicology test (utox) is performed on these samples. A positive utox test indicates presence of cocaine (and its metabolite – benzoylecgonine) with values ranging from $300$ng/mL to $> 5000$ng/mL and negative utox test indicates absence of cocaine with values $< 300$ng/mL. Benzoylecgonine has an elimination half-life of roughly 13 hours thus providing ground-truth evidence for the consumption of cocaine in the period preceding the administration of the test. For purposes of clinical decision making utox values above $5000$ (below $300$) are cutoff at $5000$ ($300$) respectively and are only reported as $> 5000$ng/mL ($< 300$ng/mL).

We define label granularity shift as a difference between the temporal granularity at which ground truth labels are defined across domains. There is clearly a significant shift in temporal label granularity between the lab and the field settings in our cocaine use study. As with prior probability shift and covariate shift, label granularity shift is a systemic problem in many mHealth study designs. It arises due to the fact that it is impractical for subjects in field-based data collection protocols to provide labels at the same level of tem-

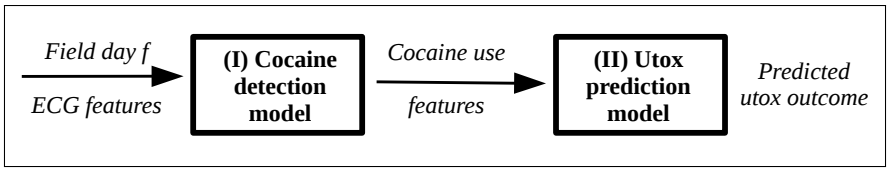



Figure 5.2: Proposed two stage processing pipeline

poral granularity that is possible in lab-based data collection protocols where subjects are closely monitored (and activity sessions are often video recorded). Methods for detecting such shifts are not necessary as their presence is obvious from the study design, but methods for adapting detection models across large temporal discrepancies are required to enable accurate lab-to-field generalization. In the next section, we turn to the problem of mitigating each of these three problems.

## 5.4 Mitigating Dataset Shifts

In this section, we present methods for mitigating factors affecting lab-to-field generalizability of cocaine use detection. Given ECG data from a subject on a field day, *f*, our goal is to predict whether the subject used cocaine on that day. We propose a two-stage data processing and prediction pipeline for this problem as shown in Figure 5.2. In the first stage, we use a cocaine use detection model to predict cocaine use at a fine grain level (*e.g., 5-minute windows*). In the second stage, we use a utox prediction model which rolls up the fine grain cocaine predictions into coarse grain cocaine predictions (*e.g., a predicted utox outcome for field day f*).

In the following sections, we describe dataset reweighting methods from the domain adaptation literature for dealing with prior probability shift and covariate shift. These reweighting methods are introduced in the first stage of the processing pipeline. We address the problem of label granularity shift in the second stage of the processing pipeline where we convert cocaine use predictions to utox predictions.

### 5.4.1 Base Classifier

For lab-to-field generalizability, consider we are given a dataset $\mathcal{D} = \{x_n, y_n\}_{n=1:N}$ of $N$ labeled examples. Let $X_i \in \mathbb{R}^D$ be a random variable representing a feature vector for data case $i$. Let $Y_i \in \{-1, +1\}$ be a binary random variable representing a label for data case $i$. We use logistic regression as a base classifier. To accommodate for dataset shifts we introduce a per data case importance weight in the objective function. More details in using importance weighting in logistic regression objective function is in Section 2.1.3.

### 5.4.2 Prior Probability Shift

Prior probability shift is characterized by different proportions of class labels in the lab and field data. Let $P_L(y)$ be the probability distribution of labels from the lab, and $P_F(y)$ be the distribution of labels from the field. To mitigate prior probability shift, we learn the base classifier using instance weights that correct for the difference between the class proportions in the lab and field datasets.

Specifically, we instantiate instance specific weights $\delta_i(y_i, x_i)$ as shown below where $\hat{P}_F(y_i)$ is an estimate of the prior probability of label $y_i$ under the field data distribution, and $\hat{P}_L(y_i)$ is an estimate of the prior probability of label $y_i$ under the lab data distribution. These weights correct the distribution of labeled instances in the lab data so that it matches the label distribution of the field data.

$$\delta_i(y_i, x_i) = \frac{\hat{P}_F(y_i)}{\hat{P}_L(y_i)} \tag{5.1}$$

Recall that in the cocaine study, $x_i$ corresponds to ECG features in 5-minute sliding windows and $y_i$ are its associated labels. Hence $\hat{P}_L(y)$ can easily be estimated from the available lab data. We do not have direct access to 5-minute labels from the field, so we instead estimate $\hat{P}_F(y)$ based on the proportion of time that subjects self-reported consuming cocaine. While not perfect due to issues with self report, this estimate is likely to be much closer to the true time spent on cocaine consumption in the field than the lab proportions.

### 5.4.3 Covariate Shift

Covariate shift is characterized by significant differences in $P_L(x)$ and $P_F(x)$, the lab and field feature distributions. Learning under covariate shift has also been addressed by incorporating appropriate importance weights during training. The importance weights needed to correct for covariate shift are the ratio of the probability densities of test to train sets $\frac{P_F(x)}{P_L(x)}$ [102]. These weights can correct for the mismatch between lab and field distributions when the discrepancy between the distributions is moderate, but there is still overlap between the support of the distributions.

While early approaches to computing the importance weights attempted to model the individual densities directly, a better approach is to directly estimate the density ratio. This can be accomplished by learning a classifier to discriminate between feature vectors from the field (positive class), and the lab (negative class), exactly as was done in Section 5.3.2. If we define $Q(x_i)$ to be the probabilistic output of a classification model for discriminating between lab and field feature vectors, then the importance weights are defined as

$$\delta_i(y_i, x_i) = \frac{1}{(1 - Q(x_i))} \tag{5.2}$$

In our experiments, we use an $l_2$ regularized logistic regression model to estimate $Q(x_i)$ learned using $5$-fold cross validation. Note that estimating this model only relies on ECG features and does not rely on availability of cocaine use labels in either the lab or field data.

### 5.4.4 Label Granularity Shift

Label granularity shift is defined as a change in the temporal granularity of the class labels from the lab to the field. To address this problem, we propose a two-stage approach. We first learn a model on the lab data to predict label probabilities at a temporal granularity of 5-minute windows. Prior probability shift or covariate shift corrections can be applied as described above during the learning of this first stage model. The output of the first stage

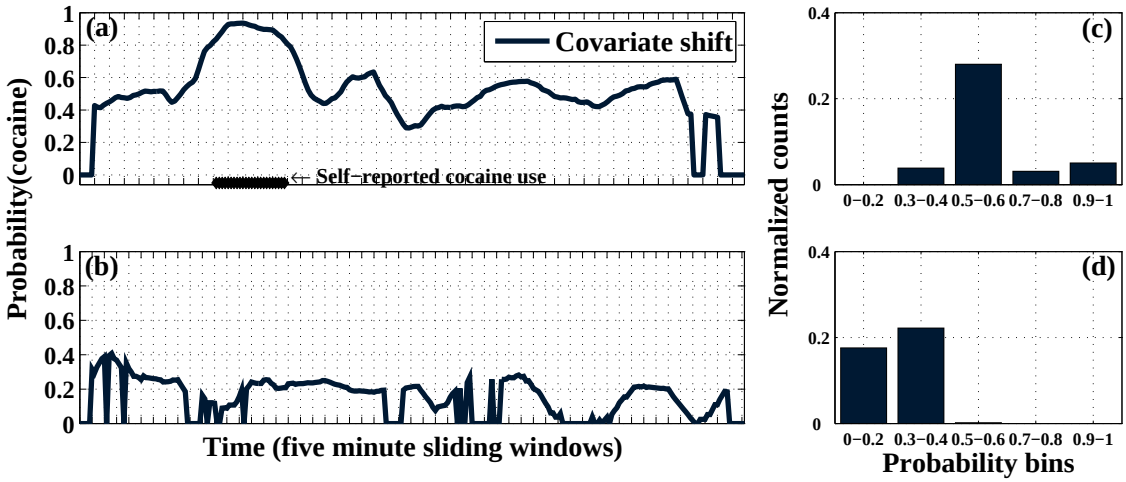



Figure 5.3: (a–b) Predicted probability of cocaine use for two sample field days. (c–d) Histogram features that represent cocaine use for the same two sample field days.

model is a time series of predicted cocaine use probabilities for each subject and each field day.

We then extract features from each time series of predicted probabilities and learn a second-stage model that predicts utox outcome from the extracted features. In this work, we use a simple histogram feature extractor that compresses the time series of cocaine use prediction over five minute windows into a histogram that indicates the proportion of windows that fall into each bin. The bins correspond to ranges of cocaine use probabilities. In our experiments, we used five equally spaced bins.

Figure 5.3 illustrates the basic concept. The left plots show the predicted probability of cocaine use for each five minute window on two sample field days. The right plots show the extracted histogram descriptors. The top plots correspond to a day with cocaine use, while the bottom plots correspond to no cocaine use. We can see from the left plots that time series of predictions for both field days are noisy, but the period of cocaine use is reasonably localized by the first stage cocaine use detection model. While the histogram descriptor discards the temporal information about when periods of increased cocaine use probability occur, the fact that they have occurred is clearly captured by the descriptor.

| **Self-report** | **utox** $< 5000$ **ng/mL** | **utox** $\geq 5000$ **ng/mL** |
|---|---|---|
| Cocaine use | 2 | 24 |
| No cocaine use | 7 | 4 |

Table 5.2: Characterizing the field dataset (37 days) by utox outcomes and subjects' self-reporting

The last step in handling label granularity shift is to learn a utox prediction model that maps the histogram descriptors to utox outcomes. We again use $l_2$ regularized logistic regression as the classifier. For our experiments, we convert utox results of $5000$ng/mL and above to positive instances and utox results below $5000$ng/mL to negative instances. This is a reasonable grouping of utox outcomes since it aligns with the threshold used in clinical decision making to determine significant amounts of cocaine *i.e.* utox $\geq 5000$ng/mL. A lower threshold could be used, but would result in even more imbalanced data for this particular study. The breakdown of positive and negative cases and how they correspond to self report is shown in Table 5.2. We can see that on a total of four days, no cocaine was reported, but the utox results showed significant cocaine consumption. This grouping results in a ground truth labeling based on utox with $28$ positive days and $9$ negative days. Though the number of positive and negative instances appear to be small, this is typical of many drug studies where the cost to obtain such data can be very high. An interesting observation is lower right corner entry where users report no cocaine use for four days but the urine test outcome is positive with significant amounts of cocaine. This further adds evidence that self-report is unreliable for this subject population.

The need to create a compressed representation comes from availability of few labeled examples from the field. Recall that in the field data we have one label corresponding urine test outcome for every $24$ hours. Hence using the cocaine predictions over five minute windows as is would result in more feature than labeled examples available to train the utox prediction model. This is an instance of the curse of dimensionality problem where we have more features than labeled examples. By creating compressed representations we

have fewer features than labeled examples making it possible to learn model parameters. We note that if a greater number of field days were available to estimate the utox prediction model, a richer feature set could be used in this stage of the pipeline.

## 5.5 Empirical Protocols

In this section, we describe the different cocaine use detection (Stage I) and utox prediction (Stage II) models we experimented with, as well as several different application scenarios motivated by potential use cases. Lastly, we describe the evaluation metrics used to assess performance.

### 5.5.1 Stage I: Cocaine detection models

We use a penalized $l_2$ logistic regression classifier as the base classifier for cocaine use detection on five minute windows. We choose the penalty, $\lambda$, by performing a leave-one-subject-out importance-weighted cross validation on the lab dataset [32]. We experimented with the default base classifier and three extensions that incorporate the prior probability shift and covariate shift mitigation approaches described below:

1. **Default:** In this model, we do not account for any type of dataset shift by setting all $\delta_i(x_i, y_i) = 1$.

2. **Prior probability shift:** In this model, we handle prior probability shift by setting $\delta_i(x_i, y_i)$ according to Equation 5.1.

3. **Covariate shift:** In this model, we handle covariate shift by setting $\delta_i(x_i, y_i)$ according to Equation 5.2.

4. **Both shifts:** In this model, we handle both covariate shift and prior probability shift by setting $\delta_i(x_i, y_i)$ to the product of their respective importance weights.[1]

### 5.5.2 Stage II: Utox prediction models

We use $l_2$ regularized logistic regression as the base classifier for utox prediction models as well. We choose the logistic regression penalty, $\lambda$, by performing a $5$-fold cross validation on the training dataset. We consider several different feature sets to predict utox outcomes as described below:

1. **Utox-default:** This model uses the cocaine use probability histogram features as described in the Section 5.4 and illustrated in Figure 5.3. At the utox prediction level, this model does not account for any type of dataset shift i.e. $\delta_i(x_i, y_i) = 1$.

2. **EMA-based classifier:** This model does not use any wearable sensor data, but instead relies on subjective self-report for features. We extract three pieces of information for each field day including self-reported cocaine use in hours, self-reported monetary value of cocaine consumed, and compute elapsed time between the last cocaine use event and the time of the utox test. For field days in which this information is missing, we set these features to zero.

3. **Predict majority class:** This model does not use any features from either wearable sensors or self-reporting. It simply predicts the majority class on the training data. This model takes advantage of the class imbalance in field utox outcomes.

### 5.5.3 Application Scenarios

To evaluate the performance of the model variations described in the previous sections, we investigated several scenarios that reflect possible real-world use cases for the appli-

[1]Note that the product combination rule assumes that the two types of shifts are independent. In many real world applications this may not be the case since one underlying latent source may give rise to multiple types of dataset shift. We leave further investigation of this point to future work.

| Scenarios | Lab dataset | Prior access | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Preceding field days within subject | | | Field days from other subjects | | | Test field day | |
| | | ECG | Self-report | Utox | ECG | Self-report | Utox | ECG | Self-report |
| **A** | ✓ | – | – | – | – | – | – | – | – |
| **B** | ✓ | ✓ | ✓ | – | – | – | – | – | – |
| **C** | ✓ | ✓ | ✓ | – | ✓ | ✓ | ✓ | ✓ | ✓ |
| **D** | ✓ | ✓ | ✓ | ✓ | – | – | – | ✓ | ✓ |

Table 5.3: This table describes four application scenarios that assume different access to prior field data

cation of a wireless cocaine use monitoring system. The primary goal is to predict utox outcomes on a daily basis. We assume that predictions are made at the end of each day.

The four scenarios that we focus on in this work are summarized in Table 5.3. In all four scenarios, we assume we always have access to lab data. This implies that all cocaine use detection models have access to the exact same lab dataset in all scenarios. However, the instance specific weights $\delta_i(x_i, y_i)$ used to mitigate dataset shifts change depending on what type of field data we have prior access to. Across all four scenarios, we are interested in handling dataset shifts in the cocaine use detection model, hence the utox prediction model always operates in *utox-default* mode. We first describe each scenario in detail. We present results for each scenario in the next section.

**1. Scenario A - Strict Lab-to-Field:** In this scenario, we assume we only have access to lab data *i.e.* no prior access to field data of any type (Table 5.3, Scenario A). The best we can do in this scenario is to train a cocaine use detection model while not accounting for any type of dataset shift (*i.e.* the default model).

Since we assume no prior field data in this scenario, we construct a synthetic utox training set from lab data to train the utox prediction model. Specifically, we process the lab data to obtain daily cocaine use probability histogram descriptors as shown in Figures 5.3c–d. We assume that lab days with cocaine use sessions correspond to positive utox outcomes, and days with only non-cocaine activities correspond to negative utox outcomes. While

utox values were not recorded in the lab, sufficient cocaine was consumed by subjects that the tests on those days would have been positive. This synthetic utox training dataset has exactly twenty instances (one day with cocaine use and one without for each of ten subjects).

To make utox predictions under this scenario, we first use the lab data to train the cocaine prediction model. We then form the synthetic utox training dataset and train a utox prediction model. We then apply the cocaine use detection model to each test field day's ECG data to produce cocaine use prediction curves and extract the daily cocaine use histogram features. Finally, we apply the trained utox prediction model to the daily cocaine use histogram features.

**2. Scenario B - Unlabeled/Weakly Labeled Field Data:** In this scenario, we assume we have prior access to two types of field data: ECG data and self-reported cocaine use (Table 5.3, Scenario B). In particular, we assume that for each field subject, we have prior access to ECG and self-reported cocaine use for field days preceding the test field day. For test field days for which there are no preceding field days (*i.e.* the very first field day within each subject), we revert to using the default model to make predictions like in scenario A.

Since we have no prior access to any data from the test field day, we use ECG and self-reported cocaine use from preceding field days to estimate importance weights for mitigating dataset shifts in the first stage of the processing pipeline. We handle label granularity shift in the second stage of the processing pipeline. We follow the same steps as in scenario A to predict utox outcomes for each test field day including training the utox model on synthetic data derived from the lab as this scenario assumes we do not have prior access to utox measurements from the field.

**3. Scenario C - Across Subjects:** In this scenario, we assume we have prior access to both ECG and self-reported cocaine use data from prior field days for the test subject, as well as ECG, self-reported cocaine use, and utox for all field days from other subjects (Table 5.3, Scenario C). Importantly, we have no access to utox outcomes for the test subject.

In this scenario, we estimate importance weights for prior probability shift and covariate shift by using all available data from the test subject and all of the available lab data, similar to Scenario B. But, unlike Scenario B there are two important differences: one, in this scenario we use data from the test field day along with data from preceding field days to compute importance weights for covariate shift and prior probability shift; two, this scenario assumes prior access to utox measurements from other field subjects. The ECG data from other field subjects is processed to extract histogram feature descriptors and the labeled data cases are added to the synthetic utox dataset extracted from the lab subjects when estimating the utox prediction model.

**4. Scenario D - Personalization:** In this scenario, we assume we have access to ECG, self-reported cocaine use data, and utox measurements from prior field days for the test subject (Table 5.3, Scenario D). We use prior field data exactly as in scenario C, but with additional utox data cases coming from the test subject's prior field days instead of field days from other subjects. This scenario thus models the online construction of personalized cocaine use detection models.

### 5.5.4 Evaluation metrics

We report the mean accuracy and standard error for utox outcome prediction over all 37 test field days, as well as the area under ROC curve (AUROC), which is less sensitive to class imbalance. We use the probabilities output by the utox prediction model as input to the AUROC computation.

## 5.6 Results

In this section, we present the results of applying the dataset shift mitigation approaches to the four utox prediction application scenarios. We present classification accuracies for all four scenarios along with standard error bars in Figures 5.4a–d. We present AUROC results for each scenario in Figure 5.4f–i respectively.

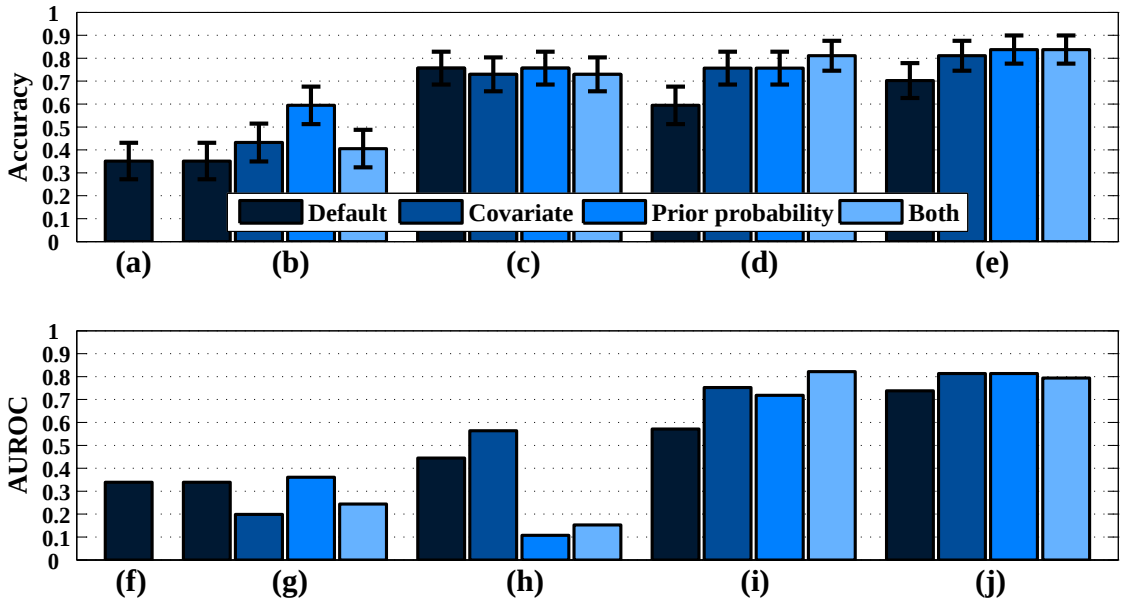



Figure 5.4: (a–e) Mean utox classification accuracies and standard errors over 37 field days (f–j) AUROC for utox prediction. Each subfigure (left-to-right) corresponds to four scenarios and a variant of scenario D respectively.

**1. Scenario A - Strict Lab-to-Field:** In scenario A, the default model has an accuracy of $35\%$ and an AUROC of $0.3$, which translates to thirteen correctly classified field days out of $37$ days. The performance of the default model, which does not account for any dataset shifts, is understandably low since the field dataset was observed to have significant shifts relative to the lab dataset in terms of both both class proportions and feature distributions.

**2. Scenario B - Unlabeled/Weakly Labeled Field Data:** In scenario B, the performance of the default model is identical to its performance in scenario A since this model does not make use of the available unlabeled and weakly labeled data. While the covariate shift and prior probability shift models result in improved accuracy relative to the default model ($43\%$ and $60\%$, respectively), their performance in terms of AUROC is worse for the covariate shift model and the same for the prior probability shift model.

**3. Scenario C - Across Subjects:** In scenario C, all models improve significantly in terms of mean accuracy with the introduction of labeled utox data from other field subjects.

All of the models (including the default model that does not account for dataset shifts at all) achieve an accuracy above $70\%$.

To explain this uniform accuracy increase, we also applied the baseline classifier that simply predicts the training set majority class for all test instances. This classifier achieves an accuracy of $76\%$ due to the class balance on the field data, the same performance achieved by the default classifier. Thus, a significant effect of introducing utox data from other subjects is to decrease the initial prior probability shift between the data used to train the utox model and the field data it is applied to at test time.

Interestingly, the AUROC performance of the covariate shift model increases significantly under Scenario C, where it outperforms all the other models, while the prior probability shift model performance actually decreases.

We also evaluate the EMA-based utox prediction model in this scenario, which performs slightly worse than guessing the majority class at $70\%$. This directly follows from the unreliability in subjective self-reporting. For eight of the 34 field days that tested positive for cocaine (*i.e.* utox $> 300$ng/mL), either the monetary amount of cocaine consumed or the self-reported cocaine use time was missing.

**4. Scenario D - Personalization:** In scenario D, the switch to personalized models leads to further improvements in terms of mean accuracy, with the model that accounts for both prior probability shift and covariate shift obtaining $81\%$ accuracy and an AUROC above $0.8$. In this scenario, all of the models for mitigating dataset shift strongly outperform the default model in terms of both accuracy and AUROC. This suggests that in the presence of between subject variability, methods for mitigating dataset shift are most helpful when applied to the problem of learning personalized models.

**5. Utox-Level Prior Probability Shift:** As a final experiment, we extend the techniques to handle dataset shifts to the utox prediction level as well. Up until now we have assumed the utox prediction model operated in *utox-default* mode. However, since we know that there is prior probability shift at the utox prediction level of the model as well,

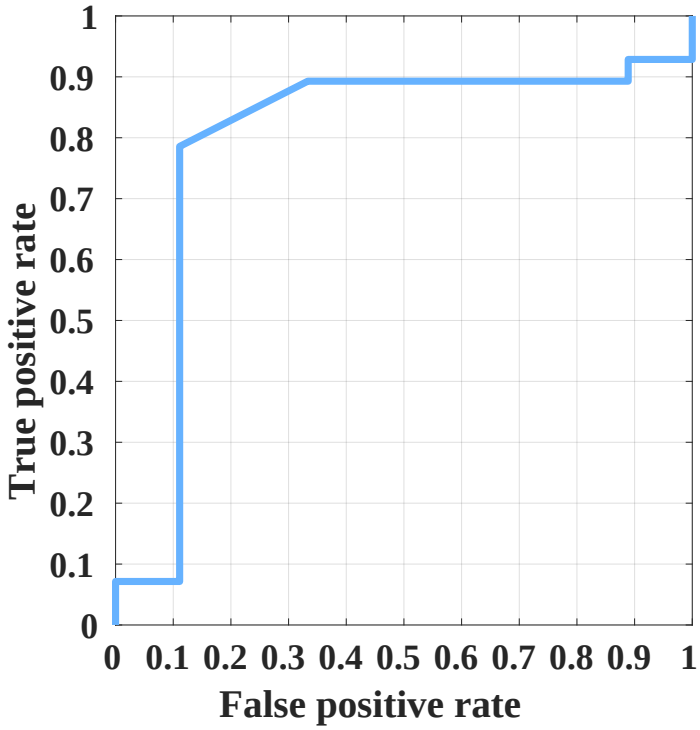



Figure 5.5: Receiver Operating Characteristics curve when applying BOTH shifts to cocaine prediction model and only prior probability shift to utox prediction model. Handling dataset shifts at both stages of the pipeline achieves a sensitivity of $80\%$ and specificity of $90\%$ respectively

we explore the application of a second level of prior probability shift mitigation during the learning of the utox prediction model. We compute importance weights by computing the prior distribution of positive and negative instances in the utox train set. Specifically, positive utox instances in the train set are assigned weights as:

$$\delta_i(x_i, y_i = +1) = \frac{\text{Proportion of preceding field days with positive utox}}{\text{Proportion of train set with positive utox}} \tag{5.3}$$

and negative utox instances are assigned weights computed using proportions of negative utox outcomes.

We apply the updated model to scenario D only. For test field days which have no preceding field days we revert to using *utox-default* prediction model. We present accuracy and AUROC results for this variant in Figures 5.4e, j respectively.

As we can see, handling prior probability shift in both the cocaine use detection stage and utox prediction stage achieves the best accuracy of any approach considered at $84\%$ (31 field days correctly classified), while achieving an AUROC of $0.81$. We present the ROC curve for this specific analysis in Figure 5.5 which shows that it achieves a sensitivity of $80\%$ and a specificity of $90\%$.

## 5.7 Related Work

A common approach to handling prior probability shift is to augment the learning of classification models using instance weights that better match the label distribution on the training set to that of the test set. Once the weights are specified, standard cost sensitive learning methods can be applied to learn the models with the instance weights [29, 56, 50, 106].

The covariate shift problem has been studied in a number of areas including human physical activity recognition [32]. A common approach to dealing with covariate shift is to again learn models with instance weights. The instance weights are selected to provide a better match between the training set feature distribution and the test set feature distribution. The weights are often derived from density ratios between the training and test feature distributions. In early work in this area, the feature distributions were estimated for the training and test sets, and the density ratios were computed explicitly. Later work observed that it is much more efficient to directly estimate the density ratio [109]. Other work, including that of Hachiya et al. [32] and Bickel et al. [11] account for covariate shift while learning the primary classifier in a joint optimization procedure with a specialized model. In this paper, we use the two-stage approach of directly estimating density ratios, followed by the application of instance weighted classification models.

Finally, we are not aware of any prior work on the temporal label granularity shift problem, although there are a number of related problems in mobile health and ubiquitous computing. For example, the temporal label uncertainty problem occurs when the time stamps associated with event labels are noisy or uncertain. The segmentation boundary uncertainty problem occurs when there is noise or uncertainty associated with the start and end time stamps of activity sessions [76, 53]. Approaches to these problems are not well matched to our setting as in our case the field labels provided by utox assessment are only available at a daily resolution.

## 5.8 Conclusions

We identified three systematic differences in lab and field cocaine use datasets making it challenging to directly deploy a lab-based cocaine use detection model in field settings. We handled prior probability shift and covariate shift by assigning importance weights to reweight the lab data distribution to better match that of the field data distribution. While both these shifts have been handled in isolation in the past, they have never been handled together in the context of cocaine use detection using wearables. Lastly, we handled a novel label granularity shift by combining cocaine predictions over $24$ hour periods to predict the urine test outcome. Only by handling the label granularity shift it is possible to reliably evaluate cocaine use detection in field settings, but we are left with the option of handling the other two dataset shifts. Hence when comparing the performance of lab-based models versus field-based models that handle both prior probability and covariate shift, we observe a significant improvement in performance from $35\%$ to $81\%$. This performance is better than both guessing the majority class as well as relying on just subject self report to predict utox outcomes. Through this work we provide evidence that wearable sensors can be used in conjunction with other sources to reliably detect cocaine use in field settings, albeit at a course granularity.

We performed an assessment of our framework in several real-world scenarios, each with access to different data. We primarily experimented with three data sources: ECG, self-reported hours of cocaine use and utox outcomes. While ECG and self-report of cocaine use was primarily made available to the Stage I - cocaine use detection model, the utox outcome was made available only to the Stage II - utox prediction models. We observed that having access to only ECG and self-report of cocaine use leads to a small boost in performance, but well below the field majority class prediction (Scenario B). However, also having access to utox outcomes leads to a large boost in performance relative to baseline methods (Scenarios C and D). We observe that this trend holds even for lab-based models that only handle label granularity shift. These results suggest that in order to detect cocaine use, we require access to labeled examples (utox outcomes) in order to train good prediction models. We can further improve performance by personalizing these models using labeled data per user (Scenario D and E). This adds evidence that personalized models perform better than population level models for the problem of cocaine use detection.

# CHAPTER 6

# HIERARCHICAL ACTIVE LEARNING TO ADDRESS LABEL SCARCITY

From the previous chapter as well as other applications [5, 73], it has been demonstrated that personalized models perform better on average than non-personalized models. But in order to develop personalized models, we require access to at least a few labeled examples per user. Furthermore, we would like these labeled examples to come from real world settings. While off-the-shelf wearable technology can be readily deployed leading to an abundance of unlabeled data, the availability of ground truth labels in real world settings is low.

The vast majority of prior work has relied on either users to self-report labels of interest [104] or require experimenters to follow study participants in order to make notes of users' activities [17]. While the latter approach does not scale and is simply impractical, there are also issues with the former approach. The manual effort is prohibitive when users are asked to log start and end times of target activities or to segment streams of sensor data into multiple activities. These labeled data collection approaches can be burdensome to users, require users to supply multiple labels for the same activity, and can suffer from recall bias and label noise (*e.g., start and end times are misreported* [4]). These factors affect the quality of ground truth labels and consequently the performance models trained using the data. The research question we address in this chapter is can we learn activity detection models with small numbers of carefully selected labeled examples using active learning.

The primary contributions of this chapter are, we develop active learning techniques to minimize the number of labeled examples required to train activity detection models. We

develop a framework to leverage similarity between users to further reduce the number of labeled examples required per user. We evaluate these techniques on a publicly available human activity recognition dataset.

Active learning has been demonstrated to perform as well as supervised machine learning techniques but with fewer labeled examples [63]. Choosing and labeling a small number of high utility examples minimizes the labeling effort from an end user point of view. Typically in wearable sensing applications, the labels are requested for a window of sensor data (*e.g., one minute window*) which further minimizes recall bias and label noise.

We develop active learning methods in the pool-based setting, which assumes that we have access to a pool of unlabeled data examples (*e.g., one minute windows of sensor data*). The active learner is allowed to evaluate the entire pool to choose an example to be queried for a label. While this setup is unrealistic for real-world health applications where sensor data continuously arrives in a stream, we use the pool-based setting as an initial experimental test bed and leave the evaluation of these techniques in more real world stream-based setting to future work.

The rest of this chapter is organized as follows. We begin by introducing the problem of human activity recognition in Section 6.1. Following this, we present two approaches to active learning: personalized active learning (Section 6.2) and group-based active learning (Section 6.3). We present empirical protocols in Section 6.4 followed by results in Section 6.5 and future work in Section 6.6. Lastly, we discuss related work on active learning for wearable sensing in Section 6.7.

## 6.1 Human Activity Recognition using Wearable Sensors

In ubiquitous and pervasive computing, the goal of human activity recognition (HAR) is to accurately recognize various activities performed by humans in natural settings using data from wearable sensors. Since the late 90's, research in HAR has focused on detecting postures and motions from daily activities (*e.g., walking, biking*) using a variety of devices

equipped with one or more sensors such as accelerometers, gyroscopes, location and physiology sensors [61]. What makes this problem challenging is that there are observable differences between repetitions of the same activity by individuals as well as significant variability between individuals when performing similar activities [9, 121]. Despite these challenges, HAR is an active area of research with many applications [61].

The two dominant machine learning approaches to HAR are supervised learning and unsupervised learning. In unsupervised learning, the goal is to cluster wearable sensor data into various activities [108]. The advantage of unsupervised HAR is there is no need for users to supply ground truth labels, but the disadvantage is that these methods are not robust when it comes to developing personalized HAR models. Unsupervised models have been demonstrated to perform suboptimally when the number of activities is unknown or when the space of hyperparameters is not fully explored [57, 100]. Supervised learning, on the other hand, assumes access to both wearable sensor data and ground truth labels to develop personalized HAR models. One of the biggest challenges is availability of ground truth labels. We propose to leverage techniques from active learning to collect ground truth labels for human activity recognition.

### 6.1.1 Extrasensory Dataset

The human activity recognition dataset we study was collected at the University of California, San Diego and is called the ExtraSensory dataset [110]. The dataset is the first large scale HAR dataset that is publicly available. It includes $60$ users, $300$K minutes and about $116$ reported activity types. Subject participation in the study varied from two to fourteen days in natural settings. The study subjects wore a smartwatch and carried a smartphone. Both devices were equipped with accelerometer and gyroscope sensors. In addition, the smartphone was equipped with GPS tracking and a microphone. One version of the dataset has features computed over one minute windows of sensor data. In total $175$ features are organized into five groups: smartphone accelerometer and gyroscope (52),

| S. No | Activity | # Users | # Target activity Samples | # Other activity Samples | Best reported Performance |
|---|---|---|---|---|---|
| 1 | Sleep | 38 | 42955 | 134045 | 0.89 |
| 2 | Computer | 38 | 23698 | 153302 | 0.71 |
| 3 | Drive | 24 | 5034 | 171966 | 0.87 |
| 4 | Surf internet | 28 | 11641 | 165359 | 0.63 |

Table 6.1: List of target activities along with number of users, data example counts along with best reported performance from [110]

smartwatch accelerometer and gyroscope (46), microphone (26), location information (17) and features pertaining to phone status (34).

Study participants provided labels of activities via the study app running on the smartphone. Activities ranged from physical activities, social, transportation and routine mundane activities. Among the many activities for which labels were provided, we focus on a suite of four activities. We chose these four activities since the labels are reliable across users, the activities are performed in isolation, and lastly the number of labels supplied by users are large enough to simulate different variants of active learning. In Table 6.1 we provide the number of users, number of positive examples, number of negative examples and best reported performance in a binary classification setting for the four activities.

Note that the number of users, and as a consequence the positive and negative example count, do not exactly match the published numbers since we recreate the entire preprocessing pipeline from the paper. In our experiments, we disqualify users that do not have at least $100$ minutes of sensor data and users that do not have at least $5$ minutes of either target or non-target activities. Nevertheless, the numbers are very close to published numbers in [110]. We point out that the 'surfing internet' activity did not have any overlap with 'computer' activity. While no explanation is given, we speculate that the former is primarily happening on the smart phone while the latter is work performed primarily on a laptop or personal computer. We note that choice of 'sleep' and 'drive' activities for active learning is not practical since it involves querying users for labels at a time when they are

most unlikely to provide labels. Regardless, we use these activities as a test bed to evaluate our methods.

## 6.2 Personalized Active Learning

Our goal is to minimize the number of labels required per user. The most straightforward approach to collecting labels for each user is to develop one active learning model per user. This per-person model is personalized and the modeling effort is focused on the specific user's needs. In the very first iteration, the active learner picks an example at random, but in subsequent iterations it picks examples with high utility. In the context of active learning we define utility as how beneficial or profitable an example is with respect to learning the decision surface. For example in uncertainty-based active learning, an example with high uncertainty (*i.e. entropy*) is more likely to benefit in learning the decision surface. The prediction model is retrained after each query and is subsequently used to assign utility scores to unlabeled examples in the pool. More details about the active learning algorithm are in Section 2.1.6.

From the above description, there are two issues that an active learner encounters. Both issues stem from the fact that in the initial iterations, the active learner has access to no (or very few) labeled examples. When starting active learning with no labeled examples the active learner has no knowledge of the decision surface. This is also referred to as the cold start problem in active learning [96]. As a result, the active learner can assign suboptimal utility scores to unlabeled examples in the pool. This can lead to poor performance of the active learner in the first few iterations until the active learner has seen enough labeled examples to start to identify the decision surface. This is particularly problematic in wearable sensing applications where the goal is to achieve good performance using only few labeled examples. To address these issues, we combine transfer learning with active learning to perform transfer active learning.

Transfer active learning is a technique to transfer domain knowledge from a source to a target domain followed by active learning in the target domain to further tune the prediction model to improve performance. This framework is directly applicable to personalized health monitoring where the prediction models learned on other users (source domain) can be transferred to a new user (target domain) followed by active learning to personalize the prediction model to each user. This framework has the added advantage of mitigating the uncertainty of the active learner in the initial iterations by relying on the prediction model transferred from the source domain.

The transfer active learning framework has the same four components as standard active learning with one subtle difference in issuing the initial query. Recall that during the very first iteration of active learning, an unlabeled example was chosen at random. In transfer active learning, we use the prediction model from the source domain to issue this very first query. The insight is that there are some commonalities in the way in which humans perform certain activities, and the transfer active learning framework exploits these commonalities to accelerate active learning.

We introduce transfer learning directly in the objective function of the classification model. We presented transfer learning for logistic regression in Section 2.1.2. For convenience we include the objective function below,

$$\mathcal{L}(W, b|\mathcal{D}) = -\sum_{n=1}^{N} \log\left(1 + \exp(-y_n(W^\top x_n + b))\right) + \lambda\|W - W_p\|_2^2 \qquad (6.1)$$

where, $\mathcal{D} = \{x_n, y_n\}_{1:N}$ is the set of actively learned labeled examples, $\lambda$ is the penalization parameter on deviation of $W$ from the prior model parameters, $W_p$. When transfer active learning is operating in the initial iterations ($\mathcal{D} = \varnothing$; $W$ is initialized to random values) then the contribution to the objective function comes exclusively from the second term. As actively learned labeled examples become available, the primary contribution shifts to the first term.

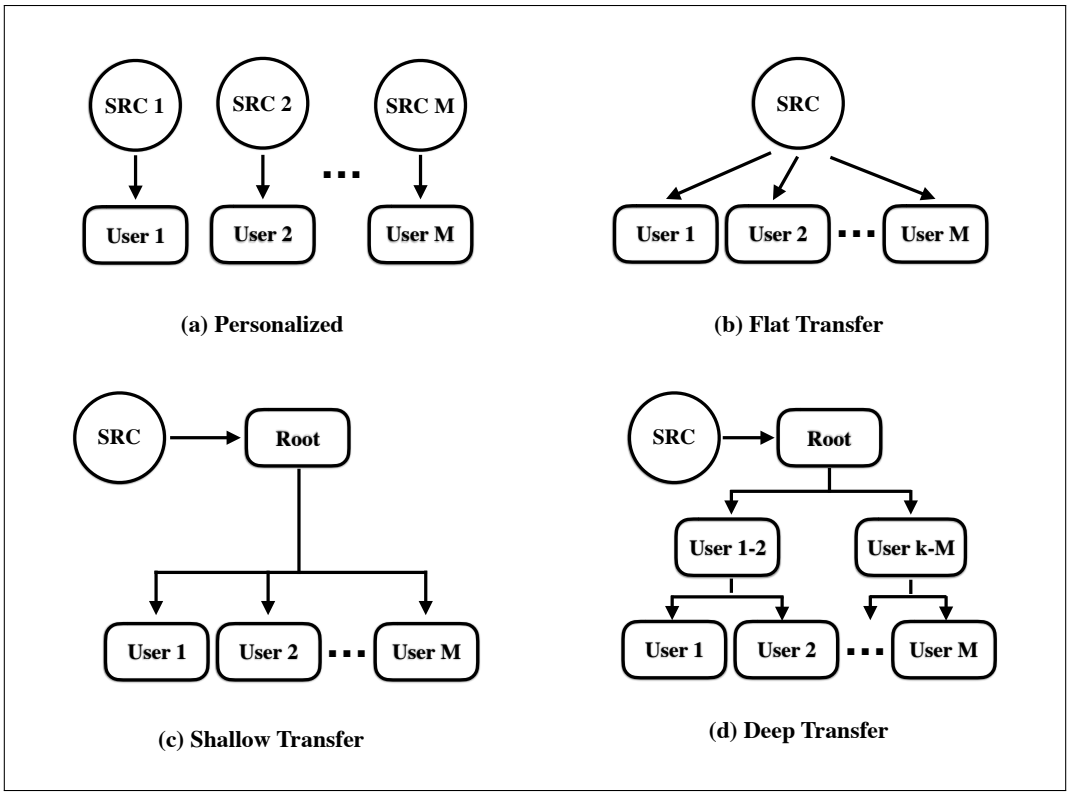



Figure 6.1: Variants of active learning. (a) personalized active learning (b)–(d) group-based active learning with flat, shallow and deep transfer. Here SRC refers to the source domain model.

---

**Algorithm 1** Personalized active learning

**procedure** PERSONALIZED ACTIVE LEARNING($N$, $Data$, $Budget$, $Query$)
  $x_{train}, y_{train} \leftarrow [\,], [\,]$
  $x_{pool}, y_{pool} \leftarrow Data(N)$
  $x_{SRC}, y_{SRC} \leftarrow Data(\neg N)$
  $W_{SRC} \leftarrow Classifier(x_{SRC}, y_{SRC}, \varnothing)$
  $W_N \leftarrow W_{SRC}$
  **while** $Budget \neq 0$ & $x_{pool} \neq \varnothing$ **do**
    $x, y \leftarrow Query(x_{pool}, y_{pool}, W_N)$
    $x_{train}, y_{train} \leftarrow x_{train} \cup x, y_{train} \cup y$
    $W_N \leftarrow Classifier(x_{train}, y_{train}, W_{SRC})$
    $x_{pool}, y_{pool} \leftarrow x_{pool} - x, y_{pool} - y$
    $Budget \leftarrow Budget - 1$
  **end while**
  **return** $W_N$
**end procedure**

Personalized active learning with transfer is graphically represented in Figure 6.1a. In the absence of any prior knowledge on user populations, a standard approach is for each user to have their own prior model. This is denoted in the figure by SRC. Examples of this type of transfer include using data from $M-1$ users to learn prior model parameters while performing active learning on the $M^{th}$ user. We investigate the alternative of using a single common prior model for all users in Section 6.3.3. Examples of this type of transfer include using data from a similar dataset gathered during a different phase of the study or from another publicly available dataset (provided the features match).

We present the pseudocode for personalized active learning in Algorithm 1. This algorithm is executed separately for each user denoted by $N$. In line #3, we create sample pool for active learning. In lines #4 and #5, we learn a SRC model using data examples from other users (also referred to as between user model). The main active learning loop runs from lines #7 through #13. The very first query is issued using the between-user model by setting $W_N$ to $W_{SRC}$ in line #6. In each iteration of active learning, we choose an example to query using the current classification model (denoted by $W_N$). The classification model is updated in each iteration in line #10 using the actively learned examples. This update uses the prior model, $W_{SRC}$, to perform transfer learning like in Equation 6.1.

The advantage of personalized active learning is that we develop one prediction model per user, which can lead to better personalization. The drawback is that each user may require many labeled examples to achieve good performance. We address this problem by leveraging the similarities between users to further minimize the number of queries.

## 6.3 Group-based Active Learning

When users have very similar activity patterns, a natural approach is to group users based on activity patterns and develop one active learning model per group. In this setup, queries issued to users in a group only benefit users within that group. Relaxing this assumption allows for sharing of labeled examples between groups, which further minimizes

the number of redundant queries. Sharing labeled examples between groups is especially useful when users partially overlap in similarity space, but not strongly enough to be in the same group. Leveraging similarities within and between groups can further minimize the number of queries per group while still achieving good performance. This we refer to as group-based active learning.

In this framework, we assume that all users' unlabeled sample pools are available simultaneously to query. This changes the problem description. We would now like to use active learning to improve the overall performance for all users simultaneously. We perform group-based active learning in three steps that we explain below.

### 6.3.1 Step I: Grouping Users

The first step is to group users based on their similarities. We learn user groupings using only their activity patterns in an unsupervised manner and ideally we would want groups that overlap to be organized closer to each other. Our approach to learn user groupings is via hierarchical agglomerative clustering. In this approach, users are grouped pairwise based on similarity scores in an iterative fashion. In each iteration of the algorithm, the pair of groups of users that are most similar are combined into a new group. The algorithm proceeds to merge groups in a hierarchical, bottom-up fashion until it reaches the root where the last merge occurs. The resulting hierarchical clustering has the following interpretation: the first merge corresponds to the pair of most similar users and the last merge corresponds to the pair most dissimilar groups. All merges in between the first and last merge proceed in a greedy fashion. More details about this clustering algorithm is presented in Section 2.1.7.

The results of hierarchical agglomerative clustering are often presented in a dendrogram as shown in Figure 6.2. In this example, five users are represented as individual leaf nodes in the dendrogram. Unlike a regular dendrogram, in this dendrogram we introduce non-leaf nodes corresponding to each merge and a root node corresponding to the last merge

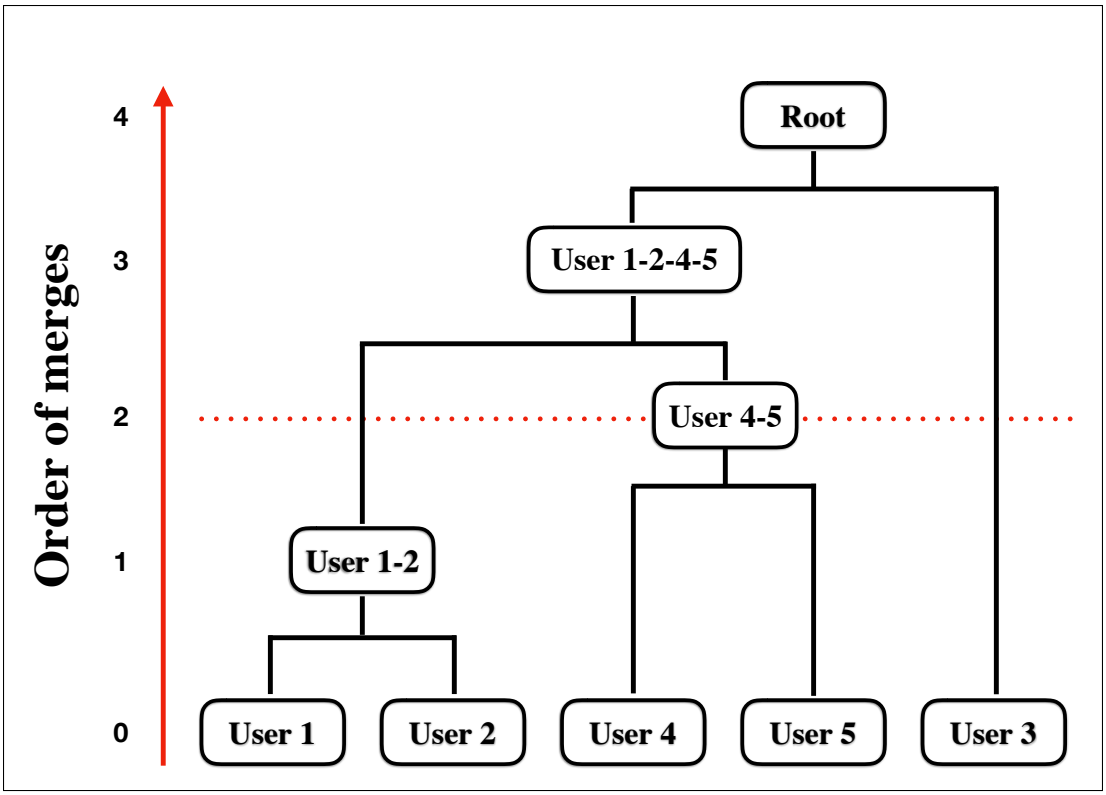



Figure 6.2: Example dendrogram of five users as output by hierarchical agglomerative clustering

in order to facilitate group-based active learning. The interpretation is that users 1 and 2 are more similar than any other pair of users in this example dataset. Hence, in the very first iteration, users 1 and 2 are merged to form a new group which is labeled as 'user 1-2', in the second iteration users 4 and 5 are merged to form another group 'user 4-5' and so on. Also, as evident in this dendrogram user 3 has much less overlap with users 1, 2 when compared to users 4, 5. This information can be inferred from the order of the merges in the dendrogram. At the root of the dendrogram is the result of merging group 'user 1-2-4-5' with user 3. This clustering algorithm has the added advantage of not needing to specify the number of clusters.

Hierarchical agglomerative clustering groups users using a similarity matrix that can be precomputed and cached. Each entry in the similarity matrix represents the similarity between pairs of users rather than pairs of data examples. A default approach is to compute

statistics on data examples for each user and then to compute the similarity between pairs of users using these statistics. Common statistics include moments [48, 118], probability densities [10], quantization profiles with hard (*e.g., kNN* [60]) and soft (*e.g., GMM* [116]) clustering of data examples to cluster centers. Some of these statistics are sensitive to outliers and others require additional hyper parameters to be tuned.

Our approach to computing similarity between users is fully data-driven, but computationally expensive with large numbers of users. Our approach computes the similarity between pairs of users as a discriminability score, $D$. Large values of $D$ imply that the users are dissimilar and vice versa. We treat the performance on a binary classification task of distinguishing between pairs of users' activity patterns as a proxy for $D$. Here the discriminability score is very similar to the techniques used to assess covariate shift (discussed in Section 5.3.2), but with the interpretation that smaller $D$ implies more similarity. This approach is very robust, easy to compute, less sensitive to outliers and no additional hyperparameters are introduced.

### 6.3.2 Step II: Active Learning over Groups

The second step is to perform active learning over groups. We leverage the dendrogram presented in the previous step as a data structure to perform group-based active learning.

In order to perform group-based active learning we need to specify the number of groups. Equivalently, we can specify the height at which to slice the dendrogram. Slicing the dendrogram at the leaf nodes (specified at height zero on the y-axis in Figure 6.2) results in each user forming their own group. Slicing the dendrogram at the root (specified at height 4 on the y-axis in Figure 6.2) results in all users forming a single group. Each slice through the dendrogram will result in one or more groups. The groups can be made up of leaf nodes, non-leaf nodes or a mix of both. The higher we slice the dendrogram, the fewer groups exist in our dataset and vice versa.

Given a slice of the dendrogram, we develop one active learning model for each group in the slice. If a group is a leaf node, we use the sample pool associated to that user to perform active learning. If a group is a non-leaf node, we combine the sample pools of all users under that non-leaf node to create the pool to perform active learning. We perform active learning over the groups in a round robin fashion. Hence, in each iteration we sweep through left-to-right to update the prediction models of each group.

For example, in the dendrogram with five users, slicing the dendrogam at height 2, bottom-up, results in 3 groups. This slice is represented by a red dotted line in Figure 6.2. This slice results in partitioning the dataset into three groups: group 'user 1-2', group 'user 4-5' and user 3. We combine the sample pools of users 1,2 to create a pool that corresponds to the respective group 'user 1-2'. Similarly we create a pool for group 'user 4-5'. One iteration of group-based active learning proceeds as follows. An unlabeled example is chosen from the pool of group 'user 1-2' and queried for a label. This labeled example is now used to update the prediction model corresponding to group 'user 1-2' only. We follow the same steps to update the prediction models for groups 'user 4-5' and user 3 respectively. Since there are only three groups, the group-based active learner alternates between these three groups until the querying budget is exhausted.

### 6.3.3 Step III: Transfer Learning between Groups

The third step is to allow sharing of labeled examples between groups. Again, we take advantage of the dendrogram structure to transfer knowledge on labeled examples between groups. Note that each node in the dendrogram is associated with a parent node with the exception of the root node. Our approach is to transfer knowledge on labeled examples between siblings nodes via parent nodes hierarchically. We perform transfer learning via parameter transfer as described in Section 2.1.2. We include the objective function with parameter transfer below,

$$\underset{W,b}{\operatorname{argmin}} \sum_{n=1}^{N} \log\left(1+\exp(-y_n(b+W^\top x_n))\right) + \lambda\|W-W_p\|_2^2 + \lambda\|b-b_p\|_2^2 \qquad (6.2)$$

In order to perform parameter transfer we train a prediction model at each parent node using the labeled examples available for all children. We treat the parent model parameters as the prior model parameters, $W_p$, when updating model parameters for children nodes. For the root node, we transfer knowledge on labeled examples from the source domain.

Continuing with the example grouping in Figure 6.2, the prediction model parameters at the root node will serve as prior model parameters for nodes 'user 1-2-4-5' and user 3 respectively. The prediction model parameters at node 'user 1-2-4-5' will serve as prior model parameters for nodes 'user 1-2' and 'user 4-5'. The prediction model parameters at node 'user 1-2' will serve as prior model parameters for nodes user 1 and 2. Similar parameter transfer occurs for users 4 and 5 respectively. Note that active learning is only performed on groups 'user 1-2', 'user 4-5 'and user 3 in a round robin fashion. At any given iteration, the prediction model at node 'user 1-2-4-5' will be trained only using actively learned examples from groups 'user 1-2' and 'user 4-5' respectively.

We interleave the model updates in the dendrogram with querying in active learning. Within a single iteration of active learning all prediction models in the hierarchy (from root to all nodes in the dendrogram) are updated in a top-down fashion. This is an expensive operation, but is essential for all models to benefit from all subsequent queries. We compared the benefit of hierarchical updates (referred to as deep transfer below) with two other updates schemes that basically differ in the number of models and subsequently how knowledge on labeled examples is transferred between groups.

**1. Group-based active learning with flat transfer**

This is the most simple approach to group-based active learning. We first group users and then perform active learning per group in a round robin fashion among all groups at any given slice of the dendrogram. We transfer knowledge from a single source domain model to all groups. Graphically, this approach is shown in Figure 6.1b. In this setup, each

user forms its own group and all groups transfer knowledge on labeled examples from a common source domain model SRC. This model allows for sharing of knowledge within groups, but only indirectly between groups via the SRC model. Knowledge on labeled examples is transferred between groups by restricting each group's model parameters to be as close to the source domain.

**Algorithm 2** Group-based active learning with flat transfer

1: **procedure** GROUP-BASED ACTIVE LEARNING FLAT TRANSFER($Data$, $Budget$, $Query$, $Groups$)
2: $N \leftarrow$ Number of users
3: $x_{SRC}, y_{SRC} \leftarrow Data(1:N)$
4: $W_{SRC} \leftarrow Classifier(x_{SRC}, y_{SRC}, \varnothing)$
5: $x_{pool}[1:Groups], y_{pool}[1:Groups] \leftarrow Data(1:N, Groups)$
6: $x_{train}[1:Groups], y_{train}[1:Groups] \leftarrow [\,], [\,]$
7: **while** $Budget \neq 0$ **do**
8: **for** $g \leftarrow 1$ to $Groups$ **do**
9: **if** $x_{train}[g] = \varnothing$ **then**
10: $W[g] \leftarrow W_{SRC}$
11: **end if**
12: $x, y \leftarrow Query(x_{pool}[g], y_{pool}[g], W[g])$
13: $x_{train}[g], y_{train}[g] \leftarrow x_{train}[g] \cup x, y_{train}[g] \cup y$
14: $W[g] \leftarrow Classifier(x_{train}[g], y_{train}[g], W_{SRC})$
15: $x_{pool}[g], y_{pool}[g] \leftarrow x_{pool}[g] - x, y_{pool}[g] - y$
16: $Budget \leftarrow Budget - 1$
17: **if** $Budget = 0$ **then**
18: **Break**
19: **end if**
20: **end for**
21: **end while**
22: **return** $W[1:Groups]$
23: **end procedure**

We present the pseudocode for group-based active learning with flat transfer in Algorithm 2. This algorithm is executed for all $N$ users simultaneously. The number of groups is provided as an input to the algorithm. In lines #3 and #4, we learn a SRC model using ten (five positive and five negative) randomly chosen data examples from $N$ users. In line #5, we create sample pools for all groups from $N$ users' data for active learning. The main

active learning loop runs from lines #7 through #21. The very first query for each group is issued using the SRC model by setting $W[g]$ to $W_{SRC}$ in line #10. In each iteration of active learning, we choose an example to query using the current classification model (denoted by $W[g]$). The classification model is updated in each iteration in line #14 using the actively learned examples. This update uses the prior model, $W_{SRC}$, to perform transfer learning like in Equation 6.1.

---

**Algorithm 3** Group-based active learning with shallow transfer

1: **procedure** GROUP-BASED ACTIVE LEARNING SHALLOW TRANSFER($Data$, $Budget$, $Query$, $Groups$)
2: $N \leftarrow$ Number of users
3: $x_{SRC}, y_{SRC} \leftarrow Data(1:N)$
4: $W_{SRC} \leftarrow Classifier(x_{SRC}, y_{SRC}, \varnothing)$
5: $x_{pool}[1:Groups], y_{pool}[1:Groups] \leftarrow Data(1:N, Groups)$
6: $x_{train}[1:Groups], y_{train}[1:Groups] \leftarrow [\,], [\,]$
7: $x_{root}, y_{root} \leftarrow [\,], [\,]$
8: $W_{ROOT} \leftarrow W_{SRC}$
9: **while** $Budget \neq 0$ **do**
10: **for** $g \leftarrow 1$ to $Groups$ **do**
11: **if** $x_{train}[g] = \varnothing$ **then**
12: $W[g] \leftarrow W_{ROOT}$
13: **end if**
14: $x, y \leftarrow Query(x_{pool}[g], y_{pool}[g], W[g])$
15: $x_{root}, y_{root} \leftarrow x_{root} \cup x, y_{root} \cup y$
16: $x_{train}[g], y_{train}[g] \leftarrow x_{train}[g] \cup x, y_{train}[g] \cup y$
17: $W_{ROOT} \leftarrow Classifier(x_{root}, y_{root}, W_{SRC})$
18: $W[g] \leftarrow Classifier(x_{train}[g], y_{train}[g], W_{ROOT})$
19: $x_{pool}[g], y_{pool}[g] \leftarrow x_{pool}[g] - x, y_{pool}[g] - y$
20: $Budget \leftarrow Budget - 1$
21: **if** $Budget = 0$ **then**
22: **Break**
23: **end if**
24: **end for**
25: **end while**
26: **return** $W[1:Groups]$
27: **end procedure**

---

**2. Group-based active learning with shallow transfer**

In this approach, we transfer knowledge on labeled examples between groups via the root node and the root node transfers knowledge on labeled examples from the source domain. Graphically, this approach is shown in Figure 6.1c. In this setup, each user forms its own group and all groups transfer knowledge on labeled examples from the root node. The root node falls backs to the source domain model SRC. This model allows for sharing of knowledge on labeled examples both within and between groups, but assumes all labeled examples are useful to all groups.

We present the pseudocode for group-based active learning with flat transfer in Algorithm 3. This algorithm is executed for all $N$ users simultaneously. The number of groups is provided as an input to the algorithm. In lines #3 and #4, we learn a SRC model using ten (five positive and five negative) randomly chosen data examples from $N$ users. In line #5, we create sample pools for all groups from $N$ users' data for active learning. The main active learning loop runs from lines #9 through #25. The very first query for each group is issued using the model at the root node by setting $W[g]$ to $W_{ROOT}$ in line #12. In each iteration of active learning, we choose an example to query using the current classification model (denoted by $W[g]$). We update both the model at the root node as well as the classification model for the $g^{th}$ group in each iteration of active learning. We first update the model at the root node in line #17 using all actively learned examples from all groups. This update uses the prior model, $W_{SRC}$, to perform transfer learning like in Equation 6.1. Second, the group-level classification model is updated in line #18 using the actively learned examples available to that group only. This update uses the prior model, $W_{ROOT}$, to perform transfer learning like in Equation 6.1.

**3. Group-based active learning with deep transfer**

In this last approach, we transfer knowledge on labeled examples between groups via a deep hierarchical structure. The root node transfers knowledge on labeled examples from the source domain. Graphically, this approach is shown in Figure 6.1d. In this setup, each user forms its own group and each group transfers knowledge on labeled examples

**Algorithm 4** Group-based active learning with deep transfer

**procedure** GROUP-BASED ACTIVE LEARNING DEEP TRANSFER($Data$, $Budget$, $Query$, $Groups$)
    $N \leftarrow$ Number of users
    $M \leftarrow$ Number of nodes in dendrogram
    $x_{SRC}, y_{SRC} \leftarrow Data(1:N)$
    $W_{SRC} \leftarrow Classifier(x_{SRC}, y_{SRC}, \varnothing)$
    $x_{pool}[1:Groups], y_{pool}[1:Groups] \leftarrow Data(1:N, Groups)$
    $x_{train}[1:Groups], y_{train}[1:Groups] \leftarrow [\,], [\,]$
    $x_{root}, y_{root} \leftarrow [\,], [\,]$
    **while** $Budget \neq 0$ **do**
        **for** $g \leftarrow 1$ to $Groups$ **do**
            **if** $x_{train}[g] = \varnothing$ **then**
                $W_{parent} \leftarrow Get - Parent - Model(g)$
                $W[g] \leftarrow W_{PARENT}$
            **end if**
            $x, y \leftarrow Query(W[g], x_{pool}[g], y_{pool}[g])$
            $x_{train}[g], y_{train}[g] \leftarrow x_{train}[g] \cup x, y_{train}[g] \cup y$
            **for** $m \leftarrow 1$ to $M$ **do**
                $x_{node}, y_{node} \leftarrow$ Get-Node-Examples$(m, x_{train}[1:Groups], y_{train}[1:Groups])$
                **if** $m =$ROOT **then**
                $W_{parent} \leftarrow W_{SRC}$
                **else**
                $W_{parent} \leftarrow Get - Parent - Model(m)$
                **end if**
                $W[m] \leftarrow Classifier(x_{node}, y_{node}, W_{parent})$
            **end for**
            $x_{pool}[g], y_{pool}[g] \leftarrow x_{pool}[g] - x, y_{pool}[g] - y$
            $Budget \leftarrow Budget - 1$
            **if** $Budget$ = 0 **then**
                **Break**
            **end if**
        **end for**
    **end while**
    **return** $W[1:Groups]$
**end procedure**

from its sibling via its parent. The root node falls backs to the source domain model SRC. This model allows for sharing of knowledge on labeled examples both within and between groups. Sharing of information between groups is local and meaningful. Prediction models at different levels of the hierarchy have access to different pieces of information. The nodes at the lower level are more group-focused whereas nodes closer to the root are learning model parameters that benefits all users.

We present the pseudocode for group-based active learning with flat transfer in Algorithm 4. This algorithm is executed for all $N$ users simultaneously. The number of groups is provided as an input to the algorithm. We denote the total number of nodes in the dendrogram as $M$. In lines #4 and #5, we learn a SRC model using ten (five positive and five negative) randomly chosen data examples from $N$ users. In line #6, we create sample pools for all groups from $N$ users' data for active learning. The main active learning loop runs from lines #9 through #32. The very first query for each group is issued using the model at its parent node by setting $W[g]$ to $W_{PARENT}$ in line #13. In each iteration of active learning, we choose an example to query using the current classification model (denoted by $W[g]$). We update all models in the dendrogram by starting at the root node and moving top-down and left to right. Each update requires access to two pieces of information: one, the labeled examples available to each node from its respective leaf nodes (shown here as a function 'Get-Node-Examples'); two, a prior model that it can transfer from (shown here as a function 'Get-Parent-Model'). This update is performed in a for loop that runs from lines #17 to #25. All model update use the prior model, $W_{PARENT}$, to perform transfer learning like in Equation 6.1.

## 6.4 Empirical Protocols

In this section, we discuss the empirical protocols used to generate results in the next section.

### 6.4.1 Train and Test Data Partitioning

For each user, we randomly partitioned the data samples into $k$ stratified folds. In our experiments, we partitioned the data into five folds per user. For baseline methods (explained below), we perform straight $k$-fold cross-validation. For active learning methods, we treat the data examples from $k-1$ folds as the sample pool and test on the $k^{th}$ fold. We repeat the above protocol for $k$ folds. Hence the comparison between baseline methods and actively learned models is fair since we are evaluating our methods on the same held out test sets.

### 6.4.2 Data Preprocessing, Feature Extraction and Label Assignment

We followed the exact same preprocessing steps as specified in [110]. We explain the steps below briefly. Specifically, from the available set of features we chose the relevant $175$ features as mentioned in Section 6.1.1. Following this, windows where one or more sensor groups were completely missing were filtered out.

Data within each user was normalized to have zero mean and unit standard deviation. For personalized active learning where we develop one model per user, we normalize the dataset using only statistics computed on the $k-1$ folds. For group-based active learning, we normalize the dataset using statistics computed on the $k-1$ folds from all users since we assume data from all users is available simultaneously. Following this, any NaN's present in the dataset were replaced by zeros.

All one minute windows pertaining to target activities (listed in Table 6.1) were assigned a positive label and all other activities were assigned a negative label. Additionally, we enforced a constraint that the 'sleep' activity should span twenty consecutive minutes or longer. Activities reported as sleep for less than twenty minutes were considered 'lying down' and hence assigned a negative label.

### 6.4.3 Baseline Methods

We compare the performance of active learning approaches to two baseline methods.

**1. Within-User:** This follows a straight within-user evaluation protocol. We train a prediction model on $k-1$ folds and evaluate on the held out $k^{th}$ fold for each user.

**2. Between-User:** This follows the leave-one-user-out evaluation protocol. We train a prediction model on data from $M-1$ users and test on the held out $M^{th}$ user. Using all data examples from $M-1$ users, $\sim 40,000$ labeled examples on average, leads to a between-user performance that is very similar to the within-user performance leaving no room for improvement via active learning. Additionally, the methods we propose in this chapter obviates the need to collect large quantities of labeled examples from $M-1$ users in the first place. Hence, in order to simulate real world settings we only use ten labeled examples (five positive and five negative) uniformly sampled at random from $M-1$ users. While, five positive examples all come from the same activity, the five negative examples come from a diverse set of activities. Our rationale for choosing ten labeled examples is that it is more practical to obtain ten minutes of ground truth labels in real world settings. Additionally, in many active learning scenarios we use this between-user model to perform transfer learning. Hence, by assuming only ten labeled examples the boost in performance from transfer learning is minimal and only serves to warm start active learning. We hypothesize that the performance, across all methods, would likely improve if we assume we have access to more than ten labeled examples to begin with.

We view the baseline methods as two extremes of access to labeled examples. At one end is the within-user protocol which has access to large quantities ($\sim 80\%$) of labeled examples from $M^{th}$ user and at the other end is the between-user protocol which has no access to labeled examples from the $M^{th}$ user. For both baseline methods we perform hyperparameter selection by performing another stratified 5-fold cross validation using training data only.

### 6.4.4 Active Learning Evaluation Protocols

We evaluated active learning techniques on the human activity recognition dataset. Each technique differed in how information was transferred between users and whether a query benefited a single user or multiple users.

Across all evaluation protocols we used penalized logistic regression with transfer (described in Section 2.1.2) as the prediction model. We investigated two querying strategies: uncertainty simply using entropy and random querying. For entropy-based methods, we use the current prediction model to compute entropies of all unlabeled examples in the sample pool. Following this, we pick the example with the highest entropy.

For each protocol we explain the use of data, initial query choice, subsequent queries, total budget, hyperparameter tuning and the prior model parameters, $W_p$, used in transfer learning. We provide a comparison of the different evaluation protocols in Table 6.2 as well.

#### 6.4.4.1 Personalized Active Learning

This is the standard version of active learning were we develop one active learning model per user. For each user we use the data from $k-1$ folds as the sample pool and test on the held out $k^{th}$ fold. We use the between-user model as a prior model ($W_p$ is set to between-user model parameters) that we transfer from. In the very first iteration, we compute utilities for unlabeled examples using the prior model and pick the example which has the highest utility. For second query and later, we use the active learning prediction model to compute utilities for unlabeled examples in the sample pool. The active learning prediction model is retrained after each query using only the actively learned examples.

We observe that active learning is sensitive to the penalty parameter as the prediction model's performance varies significantly for different penalties. Personalized active learning models start with a penalty from the prior model and they are re-tuned after every $20$ iterations during active learning. During retuning we perform $5$-fold cross validation on ac-

tively labeled examples to pick the best penalty from a range of $1e^{-4}$ to $1e^{+4}$. This retuning is triggered only when there are at least five positive and five negative actively learned examples. We perform personalized active learning for each target activity for a total budget of $100$ labeled examples per user.

#### 6.4.4.2 Group-based Active Learning with Flat Transfer

In this protocol we perform group-based active learning but transfer knowledge on labeled examples from a common prior model. The sample pool and test partitions are similar to personalized active learning with one difference: the sample pools are available simultaneously to query. Hence, when grouping users we can combine sample pools from multiple users to create a single sample pool. In order to transfer knowledge on labeled examples we create a proxy dataset as if it were from the source domain. We train a common prior model using this source domain dataset. Specifically, we create this dataset by choosing five positive and five negative examples uniformly at random from $k-1$ folds of $M$ users. Importantly, we remove these ten labeled examples from the respective sample pools so that they are not reused during active learning.

Group-based active learning models start with a penalty from the common prior model and they are re-tuned after every $M^{th}$ iteration during active learning, where $M$ is the number of users in each target activity respectively. Retuning is triggered and performed like in personalized active learning. We perform group-based active learning for each target activity for a total budget of $M \times B_T$ labeled examples where, $B_T$ is the budget for target activity $T$. Note while this number might be large, when compared to personalized active learning, it applies to all users in the dataset performing the target activity.

Lastly, we perform group-based active learning over $g$ groups in a round robin fashion. We learn a grouping of users into $g$ groups using hierarchical agglomerative clustering (outlined in Section 6.4.6). When $g$ is set to $M$ (slicing the dendrogram at the leaf nodes) essentially each user forms its own group. The first query for each group is issued using

the common prior model. Subsequent queries are issued using active learning models for each group respectively. In this setup, each group issues a maximum of $\frac{M \times B_T}{g}$ queries irrespective of the number of users within each group. While this is fair to groups with roughly equal number of users it might be unfair to groups with large disparities (*e.g., two groups with* $M-1$ *users in group one and one user in group two*). The rationale for this approach is that one single user in group two is significantly different from the rest of the population that he/she requires more queries to achieve similar performance. In the results section, we discuss the effect of the number of groups, $g$, in group-based active learning for different active learning protocols.

Computing performance in group-based active learning requires some additional work. We compute the performance of users within a group using the prediction model associated to that group. For groups that do not have access to a prediction model we utilize the prior model to assess performance. This typically happens in the very first iteration of round robin sampling when some groups do not yet have access to labeled examples.

#### 6.4.4.3 Group-based Active Learning with Shallow Transfer

In this protocol, we perform group-based active learning with two types of transfer. The first transfer is from the source domain to target domain via a common prior model. Hence, the prior model at the root node is the common prior model learned from the source domain. The second transfer is between groups in the target domain via the root node of the dendrogram. Hence, the prior model for each group in the dendrogram is the prediction model from the root.

We perform group-based active learning like in the flat transfer case but each queried example will directly benefit: one, the respective group that issued the query; two, the root node. In order to facilitate transfer of knowledge on labeled examples, we first update the prediction model at the root node and then update the prediction model of each group respectively. Note that unlike the flat transfer case we need to update the active learning

| Protocol | Sample pool | Test set | Transfer model | No. of AL models | Budget | HP | No. of model updates |
|---|---|---|---|---|---|---|---|
| Person-alized AL | $k-1$ folds | $k^{th}$ fold | Between -subjects | $M$ | 100 | Every $20^{th}$ iteration | 1 |
| Group + Flat Transfer | $k-1$ folds of $M$ users | $k^{th}$ fold of $M$ users | Source domain model | $g$ | $M \times B_T$ | Every $M^{th}$ iteration | $g$ |
| Group + Shallow Transfer | $k-1$ folds of $M$ users | $k^{th}$ fold of $M$ users | Source domain model | $g$ | $M \times B_T$ | Every $M^{th}$ iteration | $g+1$ |
| Group + Deep Transfer | $k-1$ folds of $M$ users | $k^{th}$ fold of $M$ users | Source domain model | $g$ | $M \times B_T$ | Every $M^{th}$ iteration | $g+g-1$ |

Table 6.2: Table comparing the four variants active learning. Here $k$ is the number of folds in the dataset, $M$ is the number of users in each target activity, $g$ is the number of groups in group-based active learning and $B_T$ is the budget per target activity $T$

models of all groups after each query (even groups that did not issue the query) since each group uses the root model as a prior model, which gets updated after each query. Hence, all groups indirectly benefit from each query issued in group-based active learning.

The total number of models to be updated after each query is $g+1$, where $g$ is the number of groups. We perform hyperparameter tuning separately for each of the $g+1$ models using the same criterion as active learning with flat transfer. We compute the performance of users within a group using the prediction model associated to that group. For groups that do not have access to a prediction model we utilize the prediction model from the root node to assess performance. All other details are the same as the flat transfer case.

#### 6.4.4.4 Group-based Active Learning with Deep Transfer

The last evaluation protocol is the group-based active learning with deep transfer. Again here we transfer knowledge on labeled examples from the source to target domain via the root node in the dendrogram. The critical difference is the transfer of knowledge on labeled examples between groups in the target domain. Here we leverage the full dendrogram structure.

We perform group-based active learning like in the shallow transfer case but each queried example will directly benefit: one, the respective group that issued the query; two, the root node; three all nodes along the path from the root to the group. In order to facilitate transfer of knowledge on labeled examples, we first update the prediction model at the root node, followed by updating the prediction models in between the root node and group level (layer-by-layer update from left to right) and then finally update the prediction model of each group respectively. Very similar to the shallow transfer case, we need to update the active learning models of all nodes in the the dendrogram since each node serves as a parent to another node or is the node associated to the group itself. Hence, all nodes in the dendrogram indirectly benefit from a single query issued in this framework.

The total number of models to be updated after each query ranges from just $1$ (*i.e. a single group at the root node*) to $M + M - 1$ (*i.e. each user forms its own group*). We compute the performance of users within a group using the prediction model associated to that group. For groups that do not have access to a prediction model we utilize the prediction model from the immediate parent node (grandparent if no prediction model exists at the parent node and so on) to assess performance. All other details are the same as the shallow transfer case.

### 6.4.5 Evaluation Metric and Reporting Results

Due the sample imbalances in both classes we report balanced accuracy as in [110]. Balanced accuracy (BA) is computed as,

$$BA = \frac{1}{2}\left(TPR + TNR\right)$$

where, $TPR$ and $TNR$ are true positive rate and true negative rate respectively. This metric ranges between $0$ to $1$ interpreted as greater balanced accuracy is better performance. We repeat each data analysis five times with different random seeds to mitigate the effects of train, test partition and querying strategies in active learning. We compute the balanced

accuracy per user as a mean over five repetitions and five folds. In the results section we only report the mean balanced accuracy over users along with standard error bars.

### 6.4.6 Hierarchical agglomerative clustering

We perform hierarchical agglomerative clustering using a precomputed similarity matrix. The similarity matrix is very similar to a gram matrix in kernel methods with diagonal entries set to zero and is symmetric. The number of rows and columns correspond to the users per activity. For each pair of users we compute the similarity score as the balanced accuracy to discriminate between pair of users. Lower balanced accuracy implies that the users are more similar and vice versa.

To compute balanced accuracy for a pair of users we assign a positive label to user $i$'s data examples and negative label to user $j$'s data examples. We perform a stratified $5$-fold cross validation to compute the mean balanced accuracy. The stratification is to enforce that the target activity is uniformly represented across all five folds for both users. We compute one similarity matrix per target activity listed in Table 6.1. To avoid peeking, we compute this similarity matrix using only data from $k-1$ folds (*i.e. the sample pool*) when performing active learning on the $k^{th}$ fold.

## 6.5 Results

We present results for target activities listed in Table 6.1. For each activity we compare performance of baseline methods to active learning methods. We present an in-depth analysis for the 'sleep' activity only.

### 6.5.1 Sleep Activity

In this section, we compare the performance of baseline methods to active learning methods for 'sleep' activity.

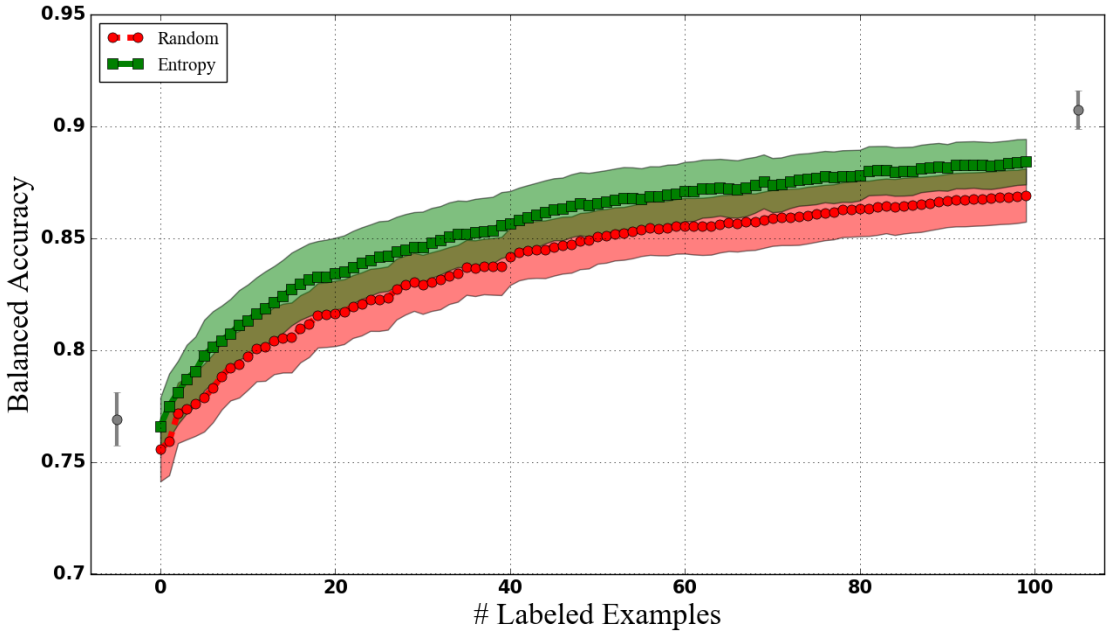



Figure 6.3: Comparing performance of baseline methods to personalized active learning for sleep activity. Here 'B' is between-subjects and 'W' is within-subjects. The lines plots correspond to entropy and random querying strategies in active learning.

#### 6.5.1.1 Baseline Methods and Personalized Active Learning

In Figure 6.3 we plot the mean balanced accuracy for the two baseline methods at the two ends of the plot along with standard error bars. To the extreme left is the between-subjects (B) results and the to the extreme right is the within-subjects (W) results.

The within-subjects performance is at $0.91$ which is slightly more than the best reported $(0.89)$ performance in [110]. The reason for this difference can be attributed to the difference in the number of data examples between the two analysis. Two other sources of difference are: (1) Vaizman et al., weighted data examples in the objective function when performing classification. These weights were used to make the classifier aware of the class imbalance, much like importance weights to handle prior probability shift in Section 5.4.2. In our analysis we did not utilize instance level weighting (2) Vaizman et al., also set the hyperparameter to be a constant $1$ for all classification tasks whereas we performed hyper-

parameter tuning. The between-subjects performance is at $0.77$. There is performance gap of $0.14$ which we hope to close with active learning methods.

Between the two baseline methods we plots the results for personalized active learning. On the x-axis is a budget of $100$ labeled examples We make two plots corresponding to the entropy and random querying strategies respectively. Each line plot is the mean over $38$ users in sleep activity and the ribbons correspond to one standard error. In personalized active learning, we transfer knowledge on labeled examples from the between-subjects model. Hence the performance of personalized active learning is very similar to the between-subjects' performance at lower budgets ($\leq 10$ labeled examples). The second observation is that entropy-based methods are on average performing statistically significantly better then random querying for all active learning budgets. Lastly, the best performance of personalized active learning at at $0.88$ which is $2\%$ from the within-subjects performance.

We attain this best performance by developing $38$ active learning models, one per user, with a total budget of $3800$ labeled examples ($38$ users $\times 100$ labeled examples per user). Essentially, we have requested that each user label about $100$ minutes of sensor data, which is not practical in real world settings. We would like to further minimize the number of labeled examples per user by exploiting the similarities between users as well as transferring knowledge on labeled examples between groups of users via group-based active learning.

#### 6.5.1.2 Group-based Active Learning

In this section, we present results from group-based active learning. First, we perform hierarchical agglomerative clustering using a precomputed similarity matrix. We compute similarity matrix as outlined in Section 6.4.6. The similarity matrix for sleep activity is shown in Figure 6.4a. The rows and columns correspond to $38$ sleep activity users. This is a symmetric matrix with diagonal entries carrying no relevant information. The heat map should be interpreted as lighter shades imply more similarity. We find some users to be more similar than others with balanced accuracies ranging from $0.72$ to $1.0$. We

acknowledge that there is not a lot of similarity between users since the dataset is collected over relatively short span of 14 days with lot of variance and sources of noise. Additionally this matrix captures similarities between pairs of users that have very little overlap in the set of activities performed.

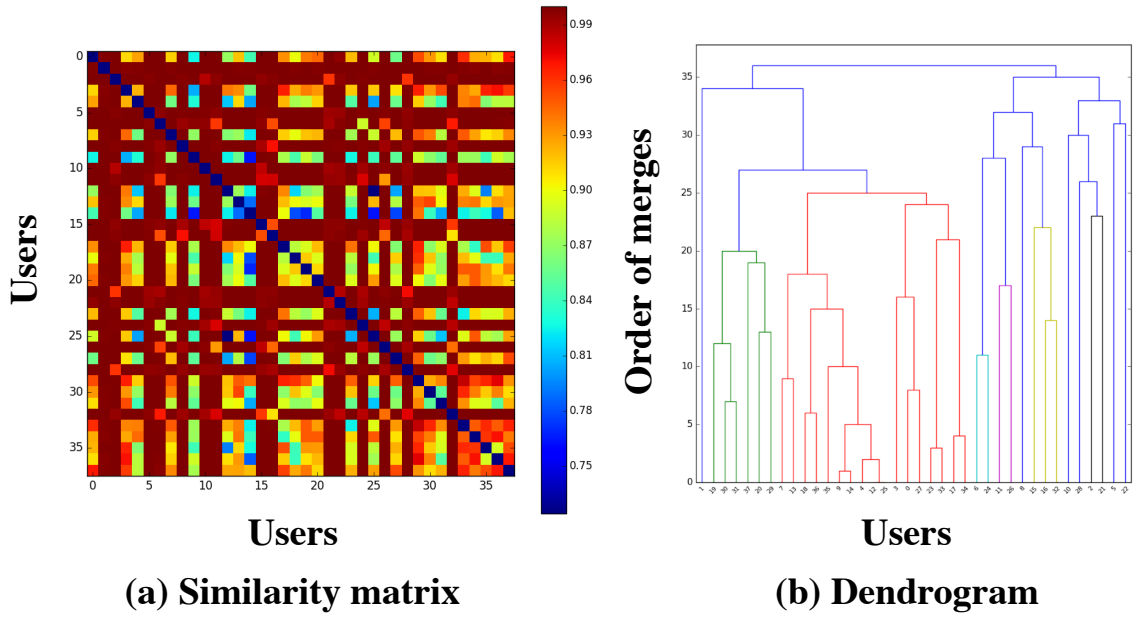



Figure 6.4: (a) Similarity matrix computed for 38 sleep users (b) dendrogram for 38 users in sleep activity

We use this similarity matrix to perform hierarchical agglomerative clustering. We present the clustering results in a dendrogram shown in Figure 6.4b. On the x-axis are the 38 users for sleep activity are arranged based on similarity score. On the y-axis we show the order of merges, bottom-up, all the way to the root node. At a high level there appears to be two groups (shown in red and green) and the remaining users are so disparate that the merges only happen near the root. We can now slice this dendrogram at 38 possible locations on the y-axis, each of which results in grouping of users into $g$ groups.

We present results from group-based active learning with flat transfer. This is the most simple case where each group has its own active learning model and all groups transfer

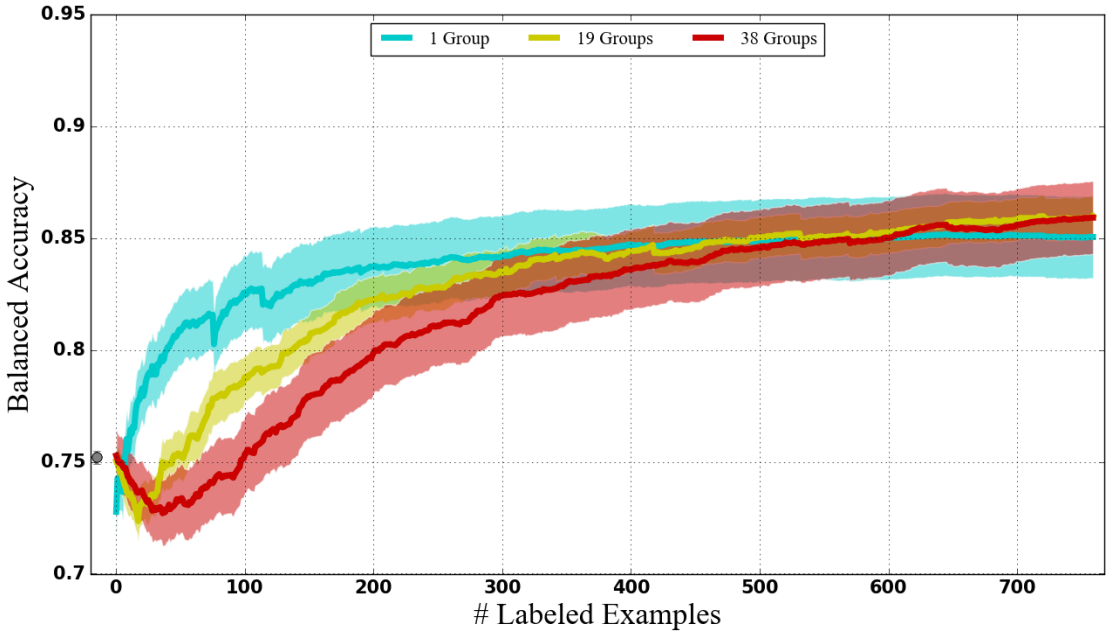



Figure 6.5: Comparing performance of 1, 19 and 38 groups in group-based active learning with flat transfer for sleep activity

knowledge on labeled examples from a common prior model from the source domain. We present results from three groupings that are of interest,

1. $g = 1$ group; we slice the dendrogram at the root node and assume all users belong to this one group.

2. $g = 38$ groups; we slice the dendrogram at the leaf nodes and assume each user forms its own group

3. $g = 19$ groups; we slice the dendrogram such that there are 19 groups, roughly half way between 1 and 38 groups.

We present the results for these three groupings in Figure 6.5. The x-axis is the total labeling budget for sleep activity which is $\sim 760$ (38 users $\times 20$ examples per user). The line plots here correspond to the entropy-based querying strategy which is significantly better than random querying. At a labeling budget of 1, only group one has issued a query and the other $g - 1$ groups rely on the prior model to estimate performance on the test set.

Hence in this setup at labeling budgets of $g \times 1, g \times 2, ..., g \times 20$ all groups should have queried at least once, twice, ..., twenty times respectively.

From this plot we observe that performing group-based active learning with flat transfer improves performance over personalized active learning. For comparison, at $20$ iterations, personalized active learning performed at $\sim 0.83$ but group-based active learning ranges between $0.85$ and $0.86$. Importantly there is substantial variability between different groups at smaller labeling budgets specifically between $1$ to $200$ labeled examples. Group-based active learning with a single group performs substantially better than other groupings at smaller labeling budgets. At larger labeling budgets the personalized models show small improvements over the single group model. The differences in performances can be attributed to the number of labeled examples versus the number of model parameters to be learned. The single group model only trains one model at the root node but the $38$ group model trains $38$ models in the dendrogram structure using the same number of labeled examples. This discrepancy is much more pronounced at smaller labeling budgets.

Another observation is that unlike personalized active learning, transfer learning in group-based active learning is not working very well. We attribute the initial drop in performance to the choice of hyperparameter $\lambda$. Recall that the hyperparameter used in the first couple of iterations (until retuning) is the same as the one used in the prior model. Typically this hyperparameter is small and hence allows for more deviation of the active learning model parameters from the prior model. Setting the hyperparameter to be large *e.g.,* $1e^{+4}$ will ensure that the group-based active learning performance will be very similar to the prior model, but this indirectly affects subsequent queries issued. This is a trade off we encounter in group-based active learning.

We make a similar plot for group-based active learning with shallow transfer. In Figure 6.6, the three lines plots correspond to groups $1$, $19$ and $38$ respectively. We point out that transfer learning between groups is not working well, at least for $19$ and $38$ groups, since we indirectly transfer knowledge of labeled examples between groups via the root node.

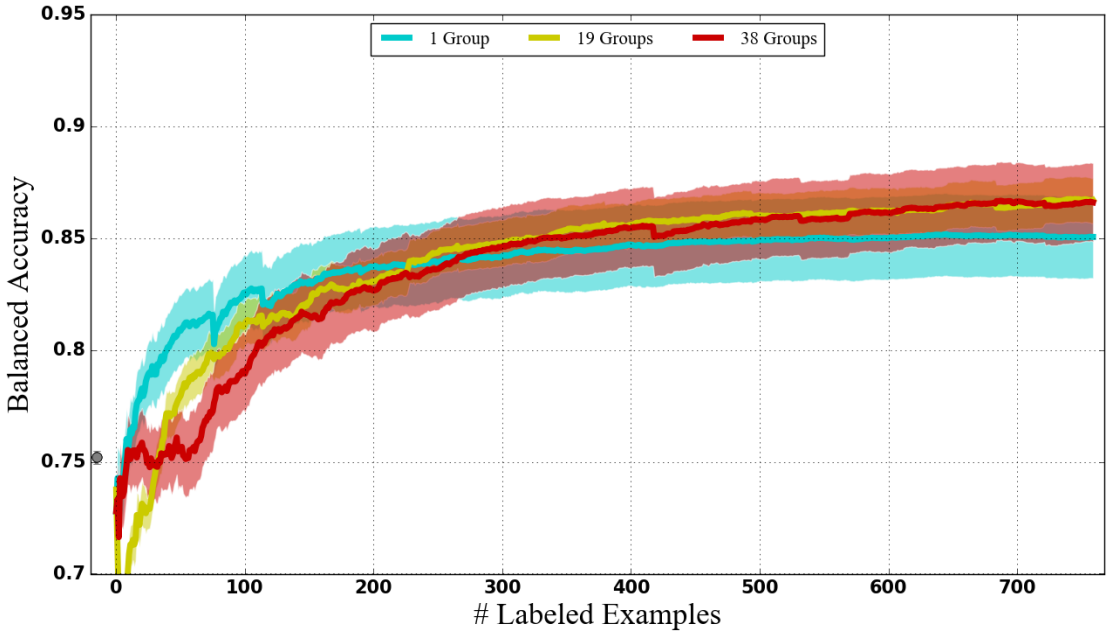



Figure 6.6: Comparing performance of 1, 19 and 38 groups in group-based active learning with shallow transfer for sleep activity

But the performance quickly stabilizes and starts to monotonically increase with as few as ten labeled examples. We observe very similar trends as before but the gap between different groupings at lower labeling budgets is closed in the shallow transfer case. We observe that this gap is further closed in the deep transfer case. In Figure 6.7, the three lines plots correspond to groups 1, 19 and 38 respectively when performing group-based active learning with deep transfer.

Determining the ideal number of groups per target activity is a very challenging problem. Based on the observed trends in Figure 6.5 we could start with a single group and create additional groups on a need basis as active learning progresses. This also poses a problem since there is no principled way to know when to switch between groups and whether to divide or merge groups. These problems are exacerbated by the availability of a small number of actively learned examples. An alternate approach is to minimize the variance in performance across different groupings and choose a fixed grouping for a given target activity. This ensures that while performance is sub optimal at some labeling

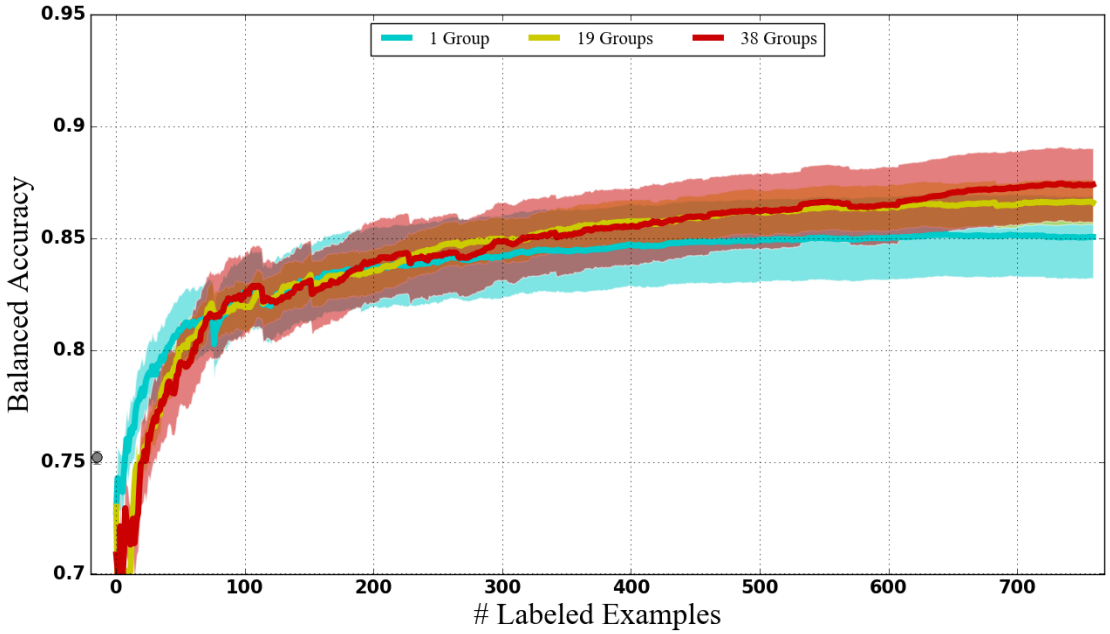



Figure 6.7: Comparing performance of 1, 19 and 38 groups in group-based active learning with deep transfer for sleep activity

budgets, overall it can only deviate $\epsilon$ from the best possible performance. We observed this phenomenon in group-based active learning with both shallow and deep transfer. This phenomenon is more pronounced in the deep transfer case. We hypothesize that sharing knowledge on labeled examples between groups in the hierarchical structure reduces the variability in performance across different groupings, especially at smaller labeling budgets. Hence, the non-leaf nodes in the deep transfer case serve dual purposes: one, to transfer knowledge on labeled examples between children; two, to capture similarity between groups of users.

We performed a head to head comparison between all three variants of group-based active learning. We assessed the deviation in performance between all possible groupings as a function of the number labeled examples. While the performance of all three variants eventually converge towards the end, the most interesting observations are at lower labeling budgets. We plot the standard deviation of performance over all possible groupings in Figure 6.8 up to a labeling budget of 350. As hypothesized, the deviation is much larger for

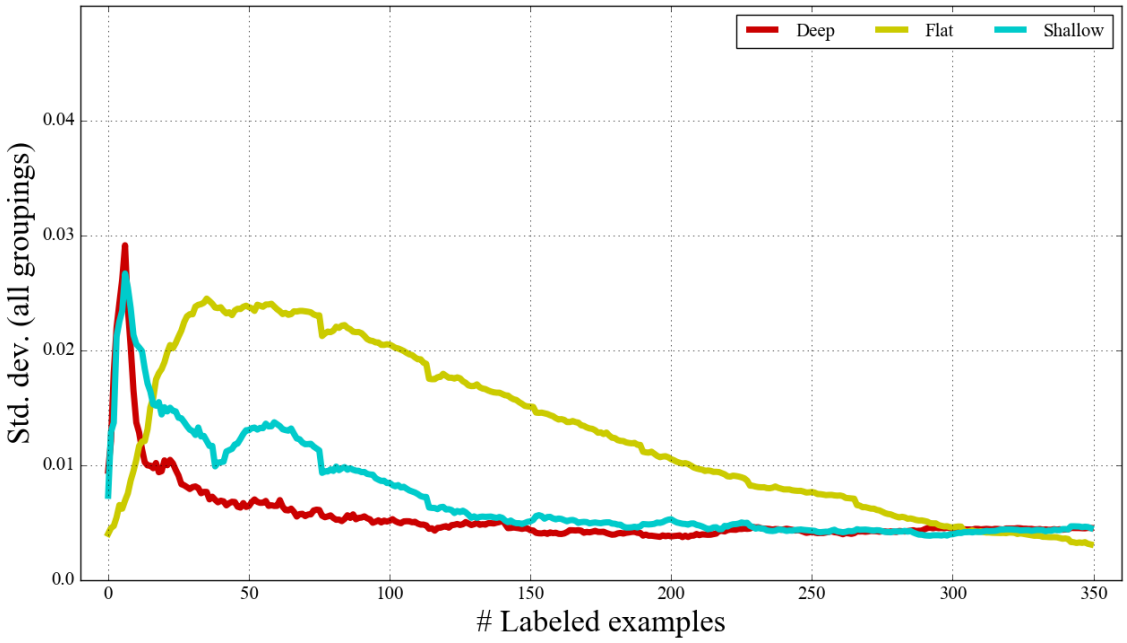



Figure 6.8: Plot of standard deviatio of performance across 38 groups in group-based active learning with flat, shllow, deep transfer respectively for sleep activity

flat transfer and requires more labeled examples to stabilize. In comparison, deep transfer has higher deviation for a very short period, $\leq 15$ labeled examples. The trend observed for shallow transfer performs in between the two extremes.

In conclusion, we believe that group-based active learning with deep transfer minimizes the deviation in performance across all possible groupings. Thereby removing one additional hyperparameter: the number of groups $g$. Hence, for group-based active learning we propose to slice the dendrogram at the leaf nodes. The reason being that the models at the leaf node are more personalized than other groupings. This assumes that each user forms its own group and we transfer knowledge on labeled examples between groups via the hierarchy.

Finally, we compare the performance of personalized active learning to group-based active learning with deep transfer. In Figure 6.9 we plot the balanced accuracies of the two active learning methods as a function of the total number of labeled examples. For

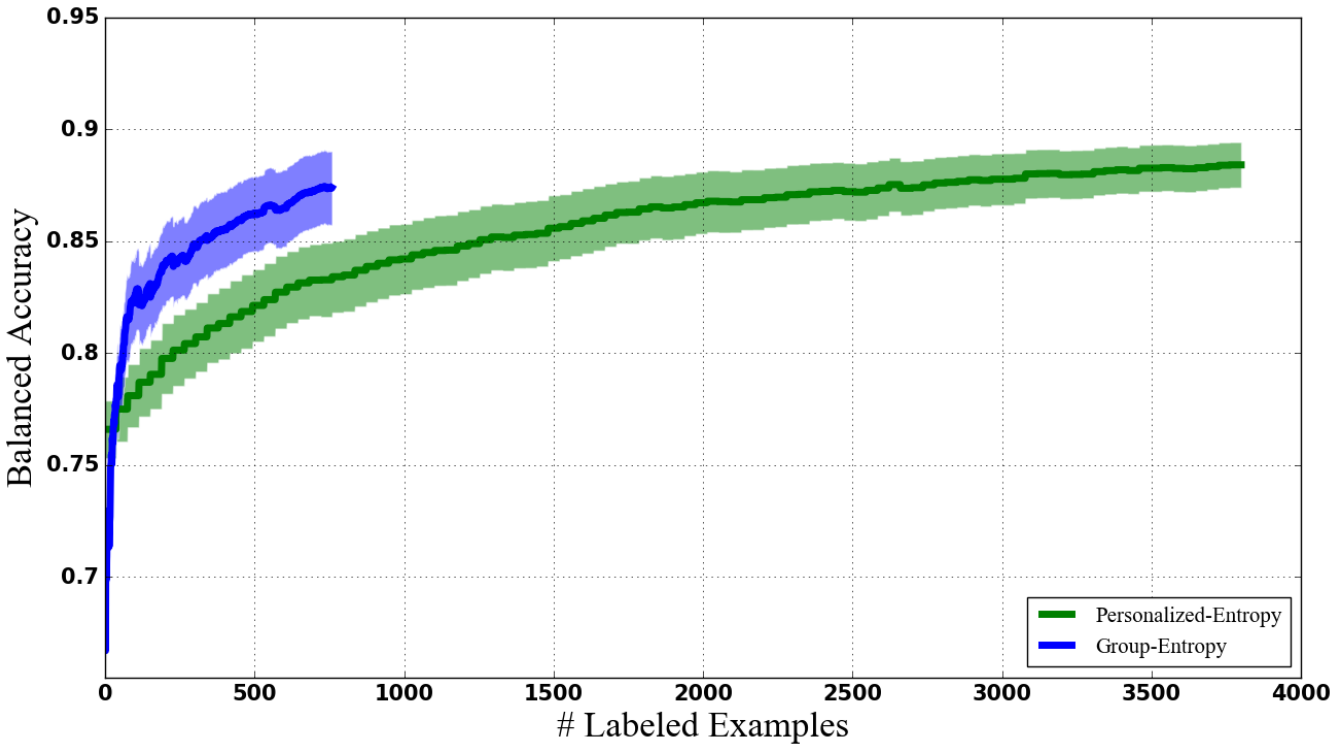



Figure 6.9: Comparing performance of group-based active learning with deep transfer (760) to personalized active learning (3800) as a function of number of labeled examples for sleep activity

both variants we only plot the entropy-based querying strategy. As mentioned above, we only compare the performance for $g = M$ groups, where $M$ is the number of users in the target activity (*i.e. each user forms its own group*). To make the performance comparable across the two settings we roll out the personalized active learning performance to a budget of $3800$. Specifically, we convert the personalized active learning performance from a budget of $100$ labeled examples to a budget of $3800$ labeled examples by replacing each personalized active learning performance entry with $38$ copies of it. This is reflected as small steps in the performance curve in Figure 6.9.

We observe that group-based active learning starts at a much lower performance but quickly surpasses personalized active learning. The cross-over happens with as few as $50$ labeled examples, which translates to each user labeling about a minute and half of their sensor data. Group-based active learning performs at $0.874$ with $20$ minutes of labeled

sensor data per user versus $0.884$ for personalized active learning with $100$ minutes of labeled sensor data per user. Group-based active learning close the performance gap with much fewer labeled examples.

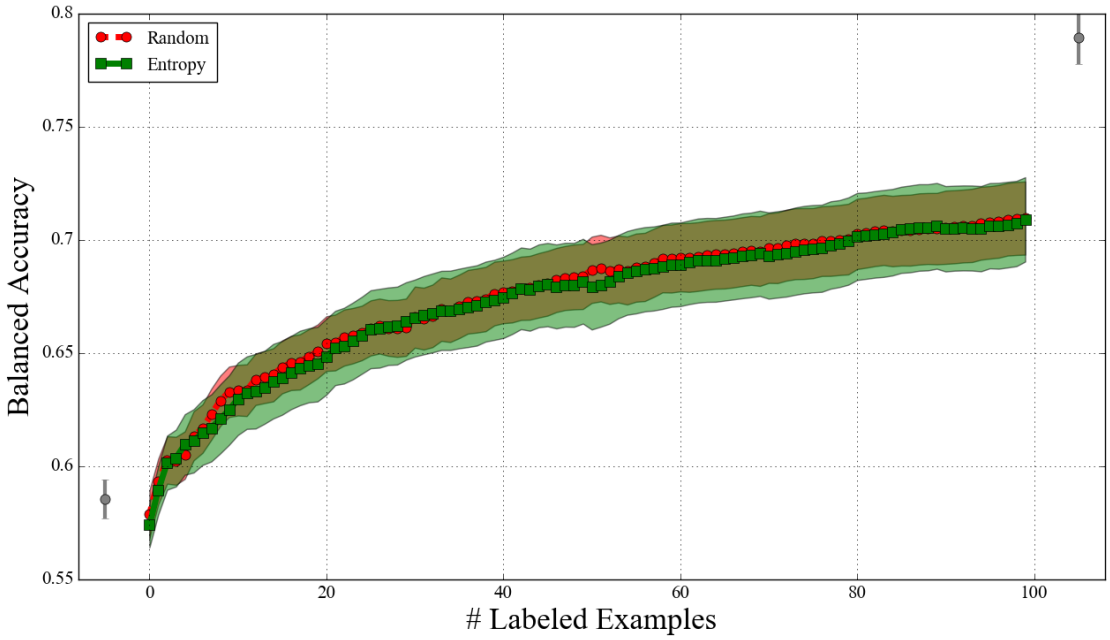



Figure 6.10: Comparing performance of baseline methods to personalized active learning for computer activity

### 6.5.2 Computer Activity

In this section we compare the performance of baseline methods to active learning methods for computer activity.

#### 6.5.2.1 Baseline Methods and Personalized Active Learning

From Figure 6.10 the within-subjects performance is at $0.79$ and the between-subjects performance is at $0.58$. There is performance gap of $0.19$ which we hope to close with active learning methods. In between the two baseline methods we plots the results for personalized active learning. Each line plot is the mean over $38$ users and the ribbons correspond to standard error bars. For this activity the entropy-based methods perform only

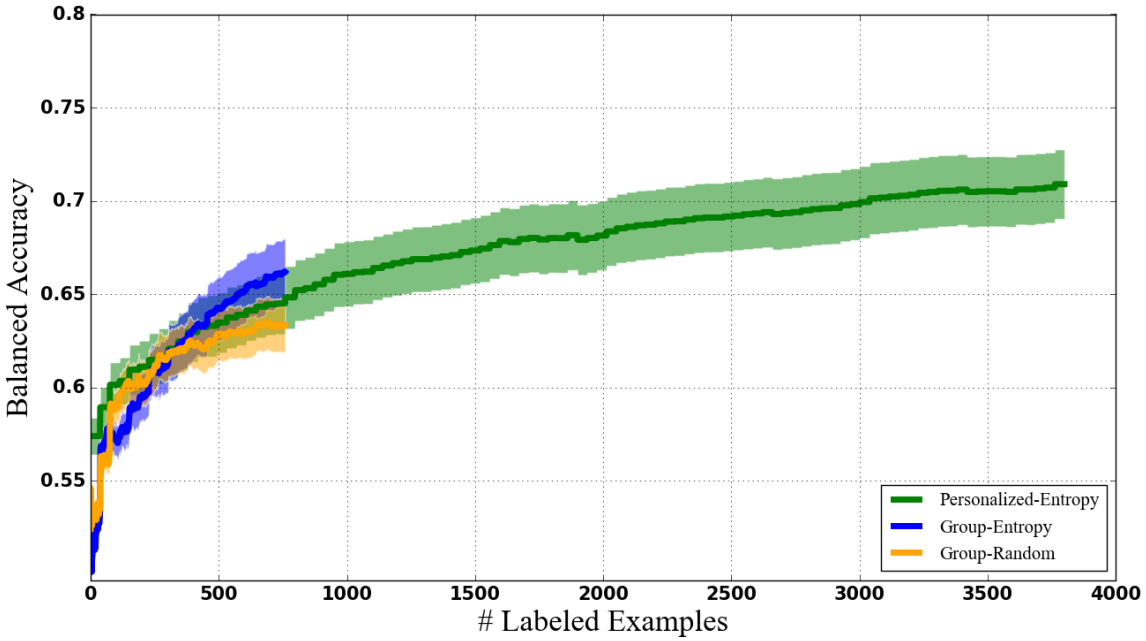



Figure 6.11: Comparing performance of group-based active learning with deep transfer (760) to personalized active learning (3800) as a function of number of labeled examples for computer activity

as well as random querying for all active learning budgets. Lastly, the best performance of personalized active learning at at $0.70$ which is $9\%$ from the within-subjects performance.

#### 6.5.2.2 Group-based Active Learning

In Figure 6.11, we compare the performance of personalized active learning to group-based active learning with deep transfer. We plot the balanced accuracies of the two active learning methods as a function of the total number of labeled examples. For both variants we only plot the entropy-based querying strategy. As mentioned above, we only compare the performance for $g = M$ groups, where $M$ is the number of users in the target activity. To make the performance comparable across the two settings we roll out the personalized active learning performance to a budget of $3800$. We observe that group-based active learning starts at a much lower performance but quickly surpasses personalized active learning. Group-based active learning performs at $0.661$ with $20$ minutes of labeled sensor per user data versus $0.708$ for personalized active learning with $100$ minutes of labeled sensor data

per user. For comparison we also plot the performance of random querying which performs significantly worse.

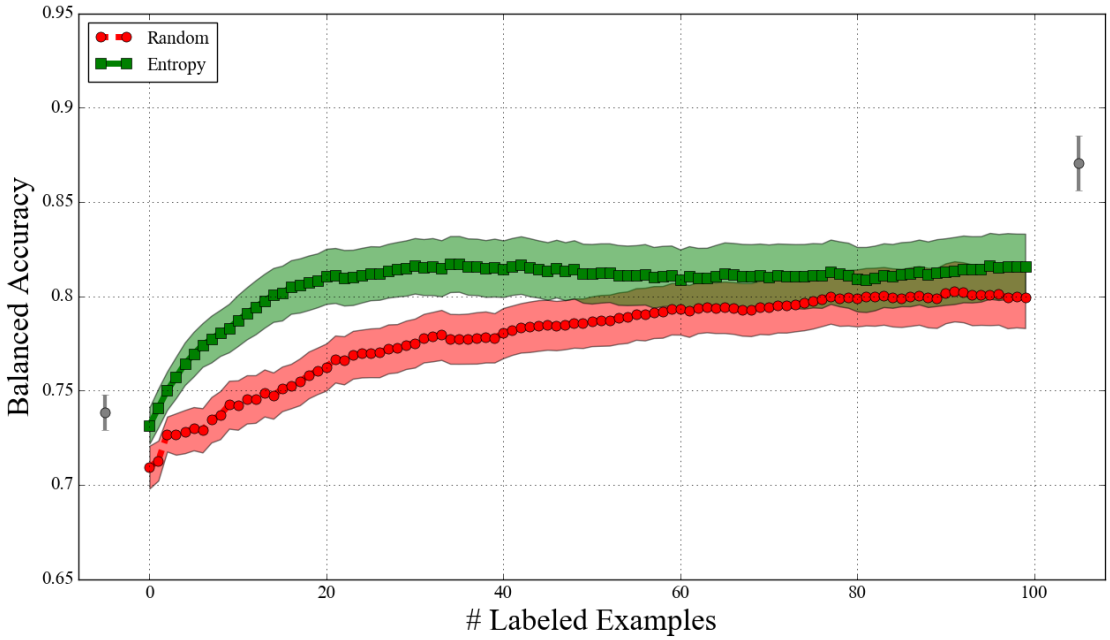



Figure 6.12: Comparing performance of baseline methods to personalized active learning for drive activity

### 6.5.3 Drive Activity

In this section we compare the performance of baseline methods to active learning methods for drive activity.

#### 6.5.3.1 Baseline Methods and Personalized Active Learning

From Figure 6.12 the within-subjects performance is at $0.87$ and the between-subjects performance is at $0.74$. There is performance gap of $0.13$ which we hope to close with active learning methods. In between the two baseline methods we plots the results for personalized active learning. Each line plot is the mean over $24$ users and the ribbons correspond to standard error bars. For this activity the entropy-based methods perform better than random random querying at lower active learning budgets and eventually the

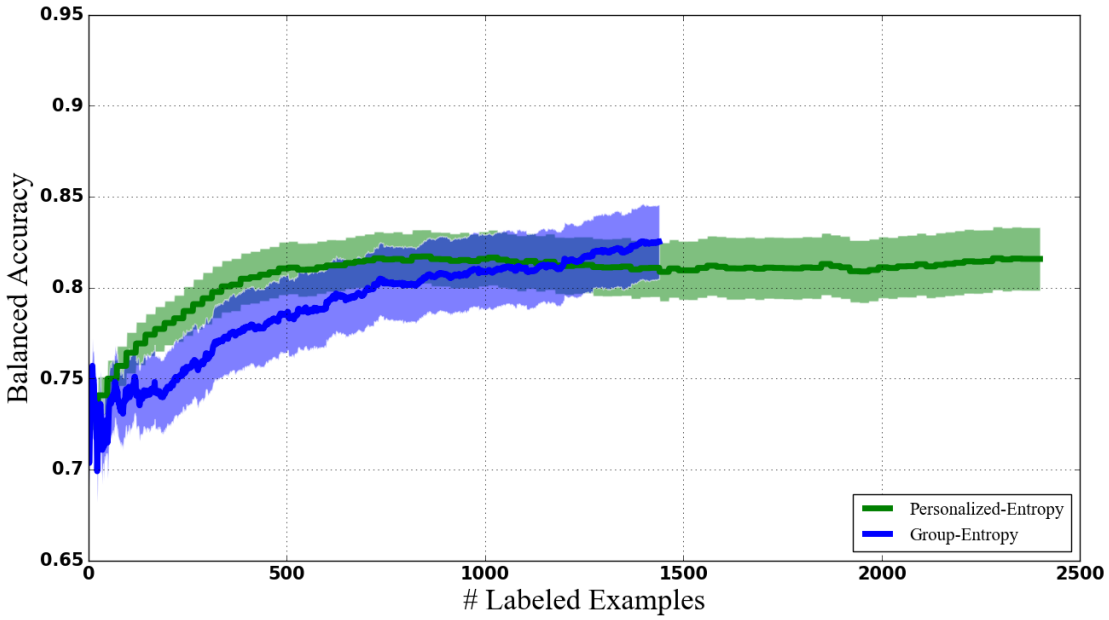



Figure 6.13: Comparing performance of group-based active learning with deep transfer (1440) to personalized active learning (2400) as a function of number of labeled examples for drive activity

two performances converge. We also observe that there is a small dip in performance in entropy-based methods between 60 and 100 labeled examples. We hypothesize that this dip is when the active learner starts to label examples which alter the decision surface leading to erroneous predictions. Lastly, the best performance of personalized active learning at at 0.817 which is 6% from the within-subjects performance.

#### 6.5.3.2 Group-based Active Learning

In Figure 6.13, we compare the performance of personalized active learning to group-based active learning with deep transfer. We plot the balanced accuracies of the two active learning methods as a function of the total number of labeled examples. For both variants we only plot the entropy-based querying strategy. As mentioned above, we only compare the performance for $g = M$ groups, where $M$ is the number of users in the target activity. To make the performance comparable across the two settings, we roll out the personalized active learning performance to a budget of 2400. Specifically, we convert the personalized

active learning performance from a budget of 100 labeled examples to a budget of 2400 labeled examples by replacing each personalized active learning performance entry with 24 copies of it. We observe that group-based active learning starts at a lower performance but surpasses personalized active learning around 1000 labeled examples. Group-based active learning performs at 0.825 with 60 minutes of labeled sensor data per user versus 0.817 for personalized active learning with 100 minutes of labeled sensor data per user.

In comparison to sleep activity, the budget is larger (20 vs. 60) and fewer users (38 vs. 24) for drive activity. Vaizman et al, do not detail the transportation modalities that fall under 'drive - I am the driver' activity. There could be substantial difference between driving a car versus a bike versus a motor cycle. Hence despite 24 users participating in this activity, the sub-activities could be very diverse.

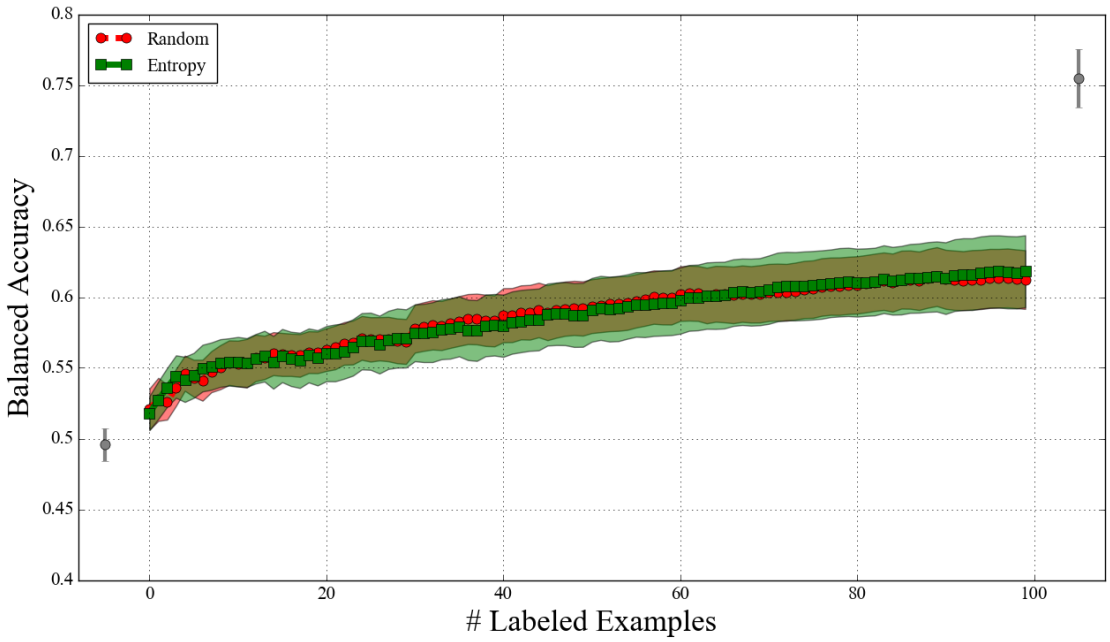



Figure 6.14: Comparing performance of baseline methods to personalized active learning for surfing the internet activity

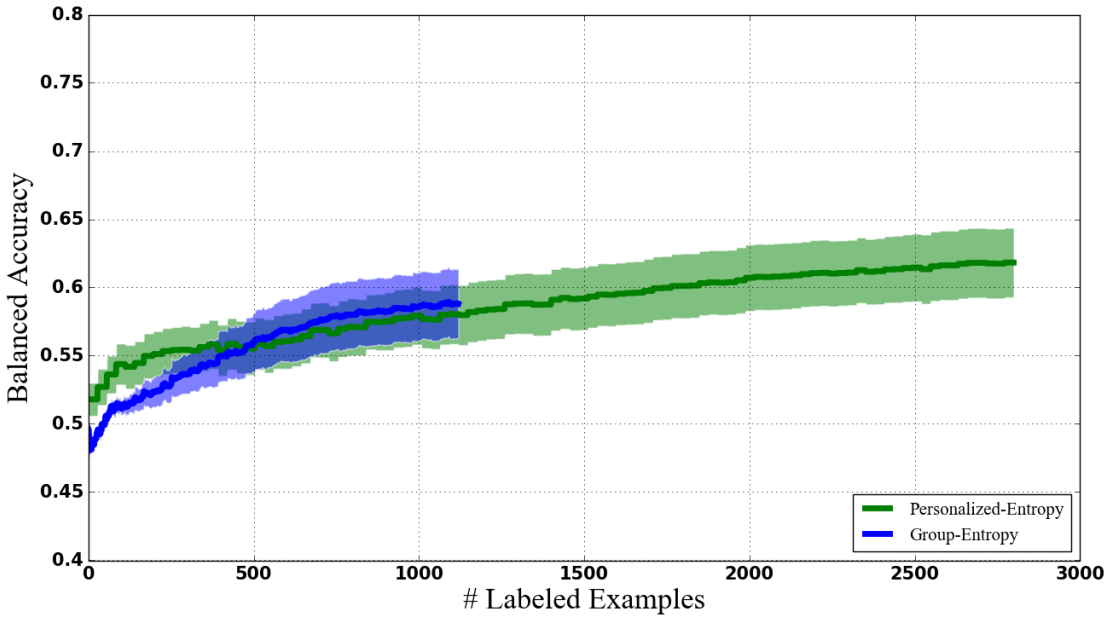



Figure 6.15: Comparing performance of group-based active learning with deep transfer (1120) to personalized active learning (2800) as a function of number of labeled examples for surfing the internet activity

### 6.5.4 Surfing the Internet Activity

In this section we compare the performance of baseline methods to active learning methods for surfing the internet activity.

#### 6.5.4.1 Baseline Methods and Personalized Active Learning

From Figure 6.14 the within-subjects performance is at $0.76$ and the between-subjects performance is at $0.49$. There is performance gap of $0.27$, which we hope to close with active learning methods. In between the two baseline methods we plots the results for personalized active learning. Each line plot is the mean over $28$ users and the ribbons correspond to standard error bars. For this activity the entropy-based methods perform only as well as random querying for all active learning budgets. Lastly, the best performance of personalized active learning at at $0.61$ which is $15\%$ from the within-subjects performance.

#### 6.5.4.2 Group-based Active Learning

In Figure 6.15, we compare the performance of personalized active learning to group-based active learning with deep transfer. We plot the balanced accuracies of the two active learning methods as a function of the total number of labeled examples. For both variants we only plot the entropy-based querying strategy. As mentioned above, we only compare the performance for $g = M$ groups, where $M$ is the number of users in the target activity. To make the performance comparable across the two settings we roll out the personalized active learning performance to a budget of $2800$. We observe that group-based active learning starts at a much lower performance but surpasses personalized active learning at $500$ labeled examples. Group-based active learning performs at $0.589$ with $40$ minutes of labeled sensor data versus $0.618$ for personalized active learning with $100$ minutes of labeled sensor data. There is a small boost in performance, but it is not statistically significant.

## 6.6 Future Work

In order to deploy this active learning framework in real world applications, we discuss three lines of future work.

**1. Stream-based active learning:** The immediate future work is to switch to stream-based active learning. This change is necessary since data continuously arrives in a stream from multiple sensors. Performing active learning in the stream-based setting is very challenging since most querying strategies are developed to be evaluated only in a pool. Even if we ignore the memory constraints and store all data examples to create a pool, it is unlikely for users to provide labels to data examples further away from the current timestamp. This requires us to develop new querying strategies for sensor data streams. Recent work has demonstrated the feasibility of stream-based active learning to engage visitors with avatars [93], label video frames [66], and adapt prediction models to evolving concepts [52, 119].

**2. Non-parametric models:** Recall that in group-based active learning we first grouped users based on similarity scores and then performed active learning on groups. This ap-

proach is limited in that the number of groups is predetermined and remains fixed throughout group-based active learning. A more robust approach to group-based active learning is to develop non-parametric models. These models create $g$ groups that best explain the partitions in the dataset while also simultaneously learning prediction model parameters for each group, thereby performing group-based active learning in one step rather than two. Furthermore, $g$, can grow as more data becomes available which fits very well with the stream-based active approach described earlier. Examples include Dirichelet process models like in [115, 51]

**3. Proactive learning:** Up until now, we assumed that the labeling oracle will always respond, provide the correct label, the cost to obtain a label is uniform, and there exists a single oracle. Relaxing these assumptions leads to proactive learning [25, 114]. Proactive learning directly applies to problems in wearable sensing since users are likely to be unresponsive in certain time windows, are genuinely confused about ground truth labels when performing multiple activities and respond to incentives by providing high quality labels at higher costs. We could perform group-based active learning using multiple oracles replacing the round robin schedule with choosing an oracle to provide a label as well.

## 6.7 Related Work

Most prior work in active learning for wearable sensing concerns the human activity recognition task. We discuss related work separately for pool-based and stream-based active learning.

Longstaff et al., propose pool-based active and semi-supervised learning techniques to collect labels [65]. Specifically, they used data from a between-subjects model as a base classifier and chose new examples to be added to the labeled set either using active learning or semi-supervised learning. The conclusion was that active learning performed better than other techniques only when there existed a performance gap when starting with a between-subjects model. They also noted the difficulty of implementing an active learning model

and emphasized user interaction as a potential problem. Saeedi et al., perform collaborative active learning with a panel of experts rather than a single oracle [94]. This is very similar to query by committee querying strategy in active learning [101].

Stream-based active learning is also referred to as online active learning. Hoque et al., used active learning to label clusters of activities in smart home settings [41]. Their method assigns raw streams of sensor data from multiple, overlapping activities into separate clusters. They minimize the number of labels required by asking users to only label clusters. By default all data examples belonging to a cluster take on the cluster label with no option to create new clusters or reassign examples to another cluster.

Another very similar approach to online active learning is to start with a supervised machine learning model (like our prior model), monitor sensor data arriving in streams and chunk them into segments. Lastly, use active learning to selectively query for a label for each segment [70, 19]. In a variant of this pipeline, the segments are clustered first to already existing clusters and a query is issued only when the segment forms a new cluster, therby minimizing the number of labels [2, 1]. An assumption in this line of work is that activities are performed in sequence and hence determining breakpoints is crucial to the segmenting step. Additionally, every distinct pair of activities will trigger new queries, *e.g., sitting and eating versus sitting and drinking*, since the assumption is that each posture-activity pair forms a separate cluster.

Transfer active learning is typically performed sequentially by first performing transfer learning followed by active learning [90]. Recent work has shown that it is possible to combine both transfer and active learning into a single framework of active transfer learning [112]. A closely related line of work to transfer domain knowledge is the dataset shift problem where unlabeled data from target domain is reweighed to match the marginal distribution in the source domain. We investigated similar techniques in Chapter 5 under domain adaptation for lab-to-field generalizability without active learning but under scenarios where small amounts of labeled data are available.

## 6.8 Conclusions

In this chapter, we investigated active learning techniques to collect ground truth labels from users in wearable sensor applications. Among the many challenges related to label scarcity, we addressed one challenge: minimizing the number of required ground truth labels while achieving comparable performance to baseline methods, which typically had access to many more labeled examples. As a proof of concept, we demonstrated the performance of active learning techniques on a set of activities in a publicly available dataset. We first showed that personalized active learning performance continuously improves as more labeled examples become available with performances matching that of supervised machine learning for some activities. Following this, we presented a novel hierarchical active learning framework that leveraged similarities between and within groups of users. We showed that this framework can achieve a comparable performance to personalized active learning while ranging from a $70\%$ reduction in labeling effort for the 'sleep' activity to a $21\%$ reduction in labeling effort for the 'surfing the internet' activity.

We evaluated our hierarchical active learning framework on a set of four activities. The point of these experiments in this chapter was to demonstrate that the hierarchical approach to transfer active learning is effective in reducing labeling effort for a diverse set of activities. From these results, it obviously works for homogeneous activities (*e.g., sleep*) but requires more labeled examples for heterogeneous activities (*e.g., drive*). And yet performs only slightly better than personalized active learning for 'computer' and 'surfing the internet' activity. Another reason for this variability in performance could be the choice of sensing modality in detecting a target activity of interest. Maybe there is not enough information that can be leveraged from wrist band sensor and smartphone in order to detect computer and internet activity. This is also reflected in the best reported performance from [110] and listed in Table 6.1 for 'computer' and 'surfing the internet' activities at $0.71$ and at $0.63$ respectively.

One of the challenges we faced in developing these techniques is that the activity of interest shows significant variability across users. Few activities of interest to health monitoring are rarely performed by individuals making it hard to train and evaluate personalized models (*e.g., eating, drinking*). Even in cases of high frequency of an activity, the distribution of this activity among users in the dataset is non-uniform (*e.g., in the Vaizman et. al., dataset users report sleep activity ranging from* $30$ *minutes to* $50$ *hours*). This irregularity makes it challenging to evaluate personalized active learning techniques especially for users that have a smaller representation. Another outcome of the limited representation of an activity is that it introduces significant variance in the few limited contexts in which it is performed. We encountered this problem when evaluating active learning methods on 'sleep' activity. Specifically we performed active learning on data from week one and tested on data from week two. We observed that performance was substantially worse since there was significant covariate shift between train and test data. This led us to partition the data into five folds for our experiments. We made similar observations for other activities as well.

Lastly, as we noted earlier the homogeneity of the activity also plays an important role especially in group-based active learning. We observed that when performing group-based active learning on the 'sit' activity, which had as many labeled examples as sleep activity, the performance trends were starkly different. On further scrutiny we discovered that sit was one of the seven mutually exclusive activities labeled by users (along with sleep, walk, bike, stand, etc) but could take on any secondary activity like eat, computer, internet, drive, etc. This made detecting the 'sit' activity very challenging especially when using features from a wrist worn device that was used to perform other secondary activities.

# BIBLIOGRAPHY


[1] Abdallah, Zahraa Said, Gaber, Mohamed Medhat, Srinivasan, Bala, and Krishnaswamy, Shonali. Cbars: Cluster based classification for activity recognition systems. In *International Conference on Advanced Machine Learning Technologies and Applications* (2012), Springer, pp. 82–91.

[2] Abdallah, Zahraa Said, Gaber, Mohamed Medhat, Srinivasan, Bala, and Krishnaswamy, Shonali. Streamar: incremental and active learning with evolving sensory data for activity recognition. In *Tools with Artificial Intelligence (ICTAI), 2012 IEEE 24th International Conference on* (2012), vol. 1, IEEE, pp. 1163–1170.

[3] Adams, Roy, Saleheen, Nazir, Thomaz, Edison, Parate, Abhinav, Kumar, Santosh, and Marlin, Benjamin. Hierarchical span-based conditional random fields for labeling and segmenting events in wearable sensor data streams. vol. 48 of *Proceedings of Machine Learning Research*, PMLR, pp. 334–343.

[4] Adams, Roy J, and Marlin, Benjamin M. Learning time series detection models from temporally imprecise labels. *arXiv preprint arXiv:1611.02258* (2016).

[5] Ali, Amin Ahsan, Hossain, Syed Monowar, Hovsepian, Karen, Rahman, Md Mahbubur, Plarre, Kurt, and Kumar, Santosh. mpuff: automated detection of cigarette smoking puffs from respiration measurements. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks* (2012), ACM, pp. 269–280.

[6] Amft, Oliver, Junker, Holger, and Troster, Gerhard. Detection of eating and drinking arm gestures using inertial body-worn sensors. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on* (2005), IEEE, pp. 160–163.

[7] Apple. Apple. `https://www.apple.com/watch`.

[8] Bari, Rummana, Adams, Roy J, Rahman, Md Mahbubur, Parsons, Megan Battles, Buder, Eugene H, and Kumar, Santosh. rconverse: Moment by moment conversation detection using a mobile respiration sensor. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2*, 1 (2018), 2.

[9] Barshan, Billur, and Yurtman, Aras. Investigating inter-subject and inter-activity variations in activity recognition using wearable motion sensors. *The Computer Journal 59*, 9 (2016), 1345–1362.

[10] Berkhin, Pavel. A survey of clustering data mining techniques. In *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[11] Bickel, Steffen, Brückner, Michael, and Scheffer, Tobias. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning* (2007), ACM, pp. 81–88.

[12] Billauer, Eli. Peak detection. `http://billauer.co.il/peakdet.html`.

[13] Bouhenguel, R., and Mahgoub, I. A risk and incidence based atrial fibrillation detection scheme for wearable healthcare computing devices. In *Pervasive Computing Technologies for Healthcare, Proceedings of the 6th International Conference on* (2012), pp. 97–104.

[14] Bouhenguel, Redjem, Mahgoub, Imad, and llyas, Mohammad. An energy efficient model for monitoring and detecting atrial fibrillation in wearable computing. In *Body Area Networks, Proceedings of the 7th International Conference on* (2012), pp. 59–65.

[15] Chatterjee, Soujanya, Hovsepian, Karen, Sarker, Hillol, Saleheen, Nazir, al'Absi, Mustafa, Atluri, Gowtham, Ertin, Emre, Lam, Cho, Lemieux, Andrine, Nakajima, Motohiro, et al. mcrave: continuous estimation of craving during smoking cessation. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (2016), ACM, pp. 863–874.

[16] Chelba, Ciprian, and Acero, Alex. Adaptation of maximum entropy capitalizer: Little data can help a lot. *Computer Speech & Language 20*, 4 (2006), 382–399.

[17] Chun, Keum San, Bhattacharya, Sarnab, and Thomaz, Edison. Detecting eating episodes by tracking jawbone movements with a non-contact wearable sensor. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2*, 1 (2018), 4.

[18] Cleland, Ian, Han, Manhyung, Nugent, Chris, Lee, Hosung, Zhang, Shuai, McClean, Sally, and Lee, Sungyoung. Mobile based prompted labeling of large scale activity data. In *International Workshop on Ambient Assisted Living* (2013), Springer, pp. 9–17.

[19] Cowan, Brendan, Suhara, Yoshihiko, Toda, Hiroyuki, and Koike, Yoshimasa. Active learning based on geographical orientation for automatic transportation mode estimation.

[20] Dalal, Navneet, and Triggs, Bill. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 1, IEEE, pp. 886–893.

[21] de Lannoy, Gael, De Decker, Arnaud, Verleysen, Michel, et al. A supervised learning approach based on the continuous wavelet transform for r spike detection in ecg. In *BIOSIGNALS (1)* (2008), pp. 140–145.

[22] de Lannoy, Gael, François, Damien, Delbeke, Jean, and Verleysen, Michel. Weighted conditional random fields for supervised interpatient heartbeat classification. *Biomedical Engineering, IEEE Transactions on 59*, 1 (2012), 241–247.

[23] de Lannoy, Gaël, Frénay, Benoît, Verleysen, Michel, and Delbeke, Jean. Supervised ecg delineation using the wavelet transform and hidden markov models. In *4th European Conference of the International Federation for Medical and Biological Engineering* (2009), pp. 22–25.

[24] Déniz, Oscar, Bueno, Gloria, Salido, Jesús, and De la Torre, Fernando. Face recognition using histograms of oriented gradients. *Pattern Recognition Letters 32*, 12 (2011), 1598–1603.

[25] Donmez, Pinar, and Carbonell, Jaime G. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM conference on Information and knowledge management* (2008), ACM, pp. 619–628.

[26] el Kaliouby, Rana, Picard, Rosalind, and BARON-COHEN, SIMON. Affective computing and autism. *Annals of the New York Academy of Sciences 1093*, 1 (2006), 228–248.

[27] Fitbit. Fitbit. `https://www.fitbit.com`.

[28] Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. *The elements of statistical learning*, vol. 1. Springer Series in Statistics, 2001.

[29] Fumera, Giorgio, and Roli, Fabio. Cost-sensitive learning in support vector machines. *Convegno Associazione Italiana per L'Intelligenza Artificiale* (2002).

[30] Ghasemzadeh, Hassan, Ostadabbas, Sarah, Guenterberg, Eric, and Pantelopoulos, Alexandros. Wireless medical-embedded systems: A review of signal-processing techniques for classification. *Sensors Journal, IEEE 13*, 2 (2013), 423–437.

[31] Haapalainen, Eija, Kim, SeungJun, Forlizzi, Jodi F., and Dey, Anind K. Psychophysiological measures for assessing cognitive load. In *Ubiquitous computing, Proceedings of the 12th ACM international conference on* (2010), pp. 301–310.

[32] Hachiya, Hirotaka, Sugiyama, Masashi, and Ueda, Naonori. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing 80* (2012), 93–101.

[33] Hale, Sharon L, Lehmann, Michael H, and Kloner, Robert A. Electrocardiographic abnormalities after acute administration of cocaine in the rat. *The American journal of cardiology 63*, 20 (1989), 1529–1530.

[34] Hasenjäger, M, and Ritter, H. Active learning in neural networks. In *New learning paradigms in soft computing*. Springer, 2002, pp. 137–169.

[35] Hassan, Eman. Recall bias can be a threat to retrospective and prospective research designs. *The Internet Journal of Epidemiology 3*, 2 (2006), 339–412.

[36] Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. *The elements of statistical learning 2nd edition*. New York: Springer, 2009.

[37] Heckerman, David, Breese, John, and Rommelse, Koos. Troubleshooting under uncertainty. Tech. rep., Technical Report MSR-TR-94-07, Microsoft Research, 1994.

[38] Hirstein, William, Iversen, Portia, and Ramachandran, VS. Autonomic responses of autistic children to people and objects. *Proceedings of the Royal Society of London B: Biological Sciences 268*, 1479 (2001), 1883–1888.

[39] Ho, Joyce, and Intille, Stephen S. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (2005), ACM, pp. 909–918.

[40] Hong, Jin-Hyuk, Ramos, Julian, and Dey, Anind K. Understanding physiological responses to stressors during physical activity. In *Ubiquitous Computing, Proceedings of the 2012 ACM Conference on* (2012), pp. 270–279.

[41] Hoque, Enamul, and Stankovic, John. Aalo: Activity recognition in smart homes using active learning in the presence of overlapped activities. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on* (2012), IEEE, pp. 139–146.

[42] Hosmer Jr, David W, and Lemeshow, Stanley. *Applied logistic regression*. John Wiley & Sons, 2004.

[43] Hossain, Syed Monowar, Ali, Amin Ahsan, Rahman, Md Mahbubur, Ertin, Emre, Epstein, David, Kennedy, Ashley, Preston, Kenzie, Umbricht, Annie, Chen, Yixin, and Kumar, Santosh. Identifying drug (cocaine) intake events from acute physiological response in the presence of free-living physical activity. In *Proceedings of the 13th international symposium on Information processing in sensor networks* (2014), pp. 71–82.

[44] Houlsby, Neil, Huszár, Ferenc, Ghahramani, Zoubin, and Lengyel, Máté. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745* (2011).

[45] Hovsepian, Karen, al'Absi, Mustafa, Ertin, Emre, Kamarck, Thomas, Nakajima, Motohiro, and Kumar, Santosh. cstress: towards a gold standard for continuous stress assessment in the mobile environment. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing* (2015), ACM, pp. 493–504.

[46] Hu, Sheng, Wei, Hongxing, Chen, Youdong, and Tan, Jindong. A real-time cardiac arrhythmia classification system with wearable sensor networks. *Sensors 12*, 9 (2012), 12844–12869.

[47] Hughes, NP, and Tarassenko, L. Automated qt interval analysis with confidence measures. In *Computers in Cardiology, 2004* (2004), pp. 765–768.

[48] Jambu, Michel, and Lebeaux, Marie-Odile. *Classification automatique pour l'analyse des données*, vol. 1. Dunod Paris, 1978.

[49] Jané, R, Blasi, A, García, J, and Laguna, P. Evaluation of an automatic threshold based detector of waveform limits in holter ecg with the qt database. In *Computers in Cardiology 1997* (1997), pp. 295–298.

[50] Japkowicz, Nathalie, and Stephen, Shaju. The class imbalance problem: A systematic study. *Intelligent data analysis 6*, 5 (2002), 429–449.

[51] Joshi, Ajjen, Ghosh, Soumya, Betke, Margrit, Sclaroff, Stan, and Pfister, Hanspeter. Personalizing gesture recognition using hierarchical bayesian neural networks. In *30TH IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR 2017)* (2017), IEEE.

[52] Kapoor, Ashish, and Horvitz, Eric J. On discarding, caching, and recalling samples in active learning. *arXiv preprint arXiv:1206.5274* (2012).

[53] Kirkham, Reuben, Khan, Aftab, Bhattacharya, Sourav, Hammerla, Nils, Mellor, Sebastian, Roggen, Daniel, and Ploetz, Thomas. Automatic correction of annotation boundaries in activity datasets by class separation maximization. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication* (2013), ACM, pp. 673–678.

[54] Koller, Daphne, and Friedman, Nir. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[55] Kuhn, Harold W. The hungarian method for the assignment problem. *Naval research logistics quarterly 2*, 1-2 (1955), 83–97.

[56] Kukar, Matjaz, Kononenko, Igor, et al. Cost-sensitive learning with neural networks. In *ECAI* (1998), Citeseer, pp. 445–449.

[57] Kwon, Yongjin, Kang, Kyuchang, and Bae, Changseok. Unsupervised learning for human activity recognition using smartphone sensors. *Expert Systems with Applications 41*, 14 (2014), 6067–6074.

[58] Lafferty, John D., McCallum, Andrew, and Pereira, Fernando C. N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning* (2001), pp. 282–289.

[59] Laguna, Pablo, Mark, Roger G, Goldberg, A, and Moody, George B. A database for evaluation of algorithms for measurement of qt and other waveform intervals in the ecg. In *Computers in Cardiology 1997* (1997), pp. 673–676.

[60] Lai, Jim ZC, and Huang, Tsung-Jen. An agglomerative clustering algorithm using a dynamic k-nearest-neighbor list. *Information Sciences 181*, 9 (2011), 1722–1734.

[61] Lara, Oscar D, and Labrador, Miguel A. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorials 15*, 3 (2013), 1192–1209.

[62] Levin, K.H., Copersino, M.L., Epstein, D., Boyd, S.J., and Gorelick, D.A. Longitudinal ECG changes in cocaine users during extended abstinence. *Drug Alcohol Depend 95*, 1-2 (2008), 160–163.

[63] Lewis, David D, and Catlett, Jason. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 148–156.

[64] Little, Roderick JA, and Rubin, Donald B. *Statistical Analysis with Missing Data*. Wiley-Interscience, 2002.

[65] Longstaff, Brent, Reddy, Sasank, and Estrin, Deborah. Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on-NO PERMISSIONS* (2010), IEEE, pp. 1–7.

[66] Loy, Chen Change, Hospedales, Timothy M, Xiang, Tao, and Gong, Shaogang. Stream-based joint exploration-exploitation active learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), IEEE, pp. 1560–1567.

[67] Magnano, A.R., Talathoti, N.B., Hallur, R., Jurus, D.T., Dizon, J., Holleran, S., M., Bloomfield. D., Collins, E., and Garan, H. Effect of acute cocaine administration on the QTc interval of habitual users. *The American journal of cardiology 97*, 8 (2006), 1244–1246.

[68] Mairal, Julien, Bach, Francis, Ponce, Jean, and Sapiro, Guillermo. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res. 11* (2010), 19–60.

[69] Martínez, Juan Pablo, Almeida, Rute, Olmos, Salvador, Rocha, Ana Paula, and Laguna, Pablo. A wavelet-based ecg delineator: evaluation on standard databases. *Biomedical Engineering, IEEE Transactions on 51*, 4 (2004), 570–581.

[70] Miu, Tudor, Missier, Paolo, and Plötz, Thomas. Bootstrapping personalised human activity recognition models using online active learning. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on* (2015), IEEE, pp. 1138–1147.

[71] Nagai, Yoko, Goldstein, Laura H, Fenwick, Peter BC, and Trimble, Michael R. Clinical efficacy of galvanic skin response biofeedback training in reducing seizures in adult epilepsy: a preliminary randomized controlled study. *Epilepsy & Behavior 5*, 2 (2004), 216–223.

[72] Natarajan, Annamalai, Angarita, Gustavo, Gaiser, Edward, Malison, Robert, Ganesan, Deepak, and Marlin, Benjamin M. Domain adaptation methods for improving lab-to-field generalization of cocaine detection using wearable ecg. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (2016), ACM, pp. 875–885.

[73] Natarajan, Annamalai, Gaiser, Edward, Angarita, Gustavo, Malison, Robert, Ganesan, Deepak, and Marlin, Benjamin. Conditional random fields for morphological analysis of wireless ecg signals. In *Proceedings of the 5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics* (2014), ACM, pp. 370–379.

[74] Natarajan, Annamalai, Parate, Abhinav, Gaiser, Edward, Angarita, Gustavo, Malison, Robert, Marlin, Benjamin, and Ganesan, Deepak. Detecting cocaine use with wearable electrocardiogram sensors. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing* (2013), pp. 123–132.

[75] Nguyen, Thai, Adams, Roy J, Natarajan, Annamalai, and Marlin, Benjamin M. Parsing wireless electrocardiogram signals with context free grammar conditional random fields. In *Wireless Health* (2016), pp. 149–156.

[76] Nguyen-Dinh, Long-Van, Roggen, Daniel, Calatroni, Alberto, and Troster, G. Improving online gesture recognition with template matching methods in accelerometer data. In *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on* (2012), IEEE, pp. 831–836.

[77] Nike. Jawbone. `https://jawbone.com/`.

[78] Nocedal, Jorge, and Wright, Stephen J. *Numerical optimization*. Springer verlag, 1999.

[79] O'Brien, Charles P. Evidence-based treatments of addiction. *Focus 9*, 1 (2011), 107.

[80] Olshausen, Bruno A, and Field, David J. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research 37*, 23 (1997), 3311–3325.

[81] on Drug Abuse, National Institute. National institute on drug abuse. `https://www.drugabuse.gov/publications/drugfacts/cocaine`.

[82] on Drug Abuse, National Institute. National institute on drug abuse. `https://www.drugabuse.gov/publications/research-reports/cocaine/what-are-long-term-effects-cocaine-use`.

[83] Pablo, Laguna, Raimon, Jané, Eudald, Bogatell, and David, Vigo Anglada. Qrs detection and waveform boundary recognition using ecgpuwave. `http://physionet.org/physiotools/ecgpuwave/`.

[84] Pan, Jiapu, and Tompkins, Willis J. A real-time qrs detection algorithm. *Biomedical Engineering, IEEE Transactions on 32*, 3 (1985), 230–236.

[85] Parate, Abhinav, Chiu, Meng-Chieh, Chadowitz, Chaniel, Ganesan, Deepak, and Kalogerakis, Evangelos. Risq: Recognizing smoking gestures with inertial sensors on a wristband. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services* (2014), ACM, pp. 149–161.

[86] Pärkkä, Juha, Ermes, Miikka, Korpipää, Panu, Mäntyjärvi, Jani, Peltola, Johannes, and Korhonen, Ilkka. Activity classification using realistic data from wearable sensors. *Information Technology in Biomedicine, IEEE Transactions on 10*, 1 (2006), 119–128.

[87] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[88] Penberthy, Jennifer K, Ait-Daoud, Nassima, Vaughan, Michelle, and Fanning, Tasmin. Review of treatment for cocaine dependence. *Current Drug Abuse Reviews 3*, 1 (2010), 49–62.

[89] Poh, Ming-Zher, Loddenkemper, Tobias, Swenson, Nicholas C, Goyal, Shubhi, Madsen, Joseph R, and Picard, Rosalind W. Continuous monitoring of electrodermal activity during epileptic seizures using a wearable sensor. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE* (2010), IEEE, pp. 4415–4418.

[90] Rai, Piyush, Saha, Avishek, Daumé III, Hal, and Venkatasubramanian, Suresh. Domain adaptation meets active learning. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing* (2010), Association for Computational Linguistics, pp. 27–32.

[91] Riboni, Daniele, Sztyler, Timo, Civitarese, Gabriele, and Stuckenschmidt, Heiner. Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (2016), ACM, pp. 1–12.

[92] Rocach, L, and Maimon, O. Clustering methods data mining and knowledge discovery handbook. *Springer US* (2005), 321.

[93] Rosenthal, Stephanie, Bohus, Dan, Kamar, Ece, Horvitz, Eric, and Redmond, WA. Look versus leap: Computing value of information with high-dimensional streaming evidence. In *IJCAI* (2013).

[94] Saeedi, Ramyar, Sasani, Keyvan, and Gebremedhin, Assefaw H. Co-meal: Cost-optimal multi-expert active learning architecture for mobile health monitoring. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics* (2017), ACM, pp. 432–441.

[95] Sano, Akane, and Picard, Rosalind W. Stress recognition using wearable sensors and mobile phones. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on* (2013), IEEE, pp. 671–676.

[96] Schein, Andrew I, Popescul, Alexandrin, Ungar, Lyle H, and Pennock, David M. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* (2002), ACM, pp. 253–260.

[97] Schein, Andrew I, and Ungar, Lyle H. Active learning for logistic regression: an evaluation. *Machine Learning 68*, 3 (2007), 235–265.

[98] Schwartz, AB, Janzen, D, Jones, RT, and Boyle, W. Electrocardiographic and hemodynamic effects of intravenous cocaine in awake and anesthetized dogs. *Journal of electrocardiology 22*, 2 (1989), 159–166.

[99] Schwartz, Bryan G, Rezkalla, Shereif, and Kloner, Robert A. Cardiovascular effects of cocaine. *Circulation 122*, 24 (2010), 2558–2569.

[100] Seiter, Julia, Amft, Oliver, Rossi, Mirco, and Tröster, Gerhard. Discovery of activity composites using topic models: An analysis of unsupervised methods. *Pervasive and Mobile Computing 15* (2014), 215–227.

[101] Settles, Burr. Active learning literature survey. *University of Wisconsin, Madison 52*, 55-66 (2010), 11.

[102] Shimodaira, Hidetoshi. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference 90*, 2 (2000), 227–244.

[103] Sughondhabirom, Atapol, Jain, Diwakar, Gueorguieva, Ralitza, Coric, Vladimir, Berman, Robert, Lynch, Wendy J, Self, David, Jatlow, Peter, and Malison, Robert T. A paradigm to investigate the self-regulation of cocaine administration in humans. *Psychopharmacology 180*, 3 (2005), 436–446.

[104] Sztyler, Timo, Carmona, Josep, Völker, Johanna, and Stuckenschmidt, Heiner. Self-tracking reloaded: Applying process mining to personalized health care from labeled sensor data. In *Transactions on Petri Nets and Other Models of Concurrency XI*, vol. 9930. Springer-Verlag Berlin Heidelberg, 2016, pp. 160–180. http://www.springer.com/de/book/9783662534007.

[105] Thomaz, Edison, Essa, Irfan, and Abowd, Gregory 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* (2015), ACM, pp. 1029–1040.

[106] Ting, Kai Ming. A study on the effiect of class distribution using cost-sensitive learning. In *International Conference on Discovery Science* (2002), Springer, pp. 98–112.

[107] Tong, Simon, and Koller, Daphne. Support vector machine active learning with applications to text classification. *Journal of machine learning research 2*, Nov (2001), 45–66.

[108] Trabelsi, Dorra, Mohammed, Samer, Chamroukhi, Faicel, Oukhellou, Latifa, and Amirat, Yacine. An unsupervised approach for automatic activity recognition based on hidden markov model regression. *IEEE Transactions on Automation Science and Engineering 10*, 3 (2013), 829–835.

[109] Tsuboi, Yuta, Kashima, Hisashi, Hido, Shohei, Bickel, Steffen, and Sugiyama, Masashi. Direct density ratio estimation for large-scale covariate shift adaptation. *Information and Media Technologies 4*, 2 (2009), 529–546.

[110] Vaizman, Yonatan, Ellis, Katherine, and Lanckriet, Gert. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Computing 16*, 4 (2017), 62–74.

[111] Valenza, Gaetano, Nardelli, Mimma, Lanata, Antonio, Gentili, Claudio, Bertschy, Gilles, Paradiso, Rita, and Scilingo, Enzo Pasquale. Wearable monitoring for mood recognition in bipolar disorder based on history-dependent long-term heart rate variability analysis. *Biomedical and Health Informatics, IEEE Journal of 18*, 5 (2014), 1625–1635.

[112] Wang, Xuezhi, Huang, Tzu-Kuo, and Schneider, Jeff. Active transfer learning under model shift. In *International Conference on Machine Learning* (2014), pp. 1305–1313.

[113] Wikipedia. $QT_c$ interval. `http://en.wikipedia.org/wiki/QT_interval`.

[114] Wray, Kyle Hollins, and Zilberstein, Shlomo. A pomdp formulation of proactive learning. In *AAAI* (2016), pp. 3202–3208.

[115] Xue, Ya, Liao, Xuejun, Carin, Lawrence, and Krishnapuram, Balaji. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research 8*, Jan (2007), 35–63.

[116] Yu, Kai, Yu, Shipeng, and Tresp, Volker. Soft clustering on graphs. In *Advances in neural information processing systems* (2006), pp. 1553–1560.

[117] Zephyr. Bioharness 3. http://www.zephyr-technology.com/products/bioharness-3/.

[118] Zhang, Tian, Ramakrishnan, Raghu, and Livny, Miron. Birch: an efficient data clustering method for very large databases. In *ACM Sigmod Record* (1996), vol. 25, ACM, pp. 103–114.

[119] Zhu, Xingquan, Zhang, Peng, Lin, Xiaodong, and Shi, Yong. Active learning from stream data using optimal weight classifier ensemble. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 40*, 6 (2010), 1607–1621.

[120] Zong, W, Moody, GB, and Jiang, D. A robust open-source algorithm to detect onset and duration of qrs complexes. In *Computers in Cardiology, 2003* (2003), pp. 737–740.

[121] Zunino, Andrea, Cavazza, Jacopo, and Murino, Vittorio. Revisiting human action recognition: Personalization vs. generalization. *arXiv preprint arXiv:1605.00392* (2016).