Using Intelligent Personal Annotations to Improve Human Activity Recognition for Movements in Natural Environments

A. Akbari, R. Solis, *Student Member, IEEE,* R. Jafari, *Senior Member, IEEE,* and B. J. Mortazavi, *Member, IEEE*

***Abstract*— Personal tracking algorithms for health monitoring are critical for understanding an individual's life-style and personal choices in natural environments (NE). In order to train such tracking algorithms in NE, however, annotated data is needed, particularly when tracking a variety of activities of daily living. These algorithms are often trained in laboratory settings, with expectations that they will perform equally well in NE, which is often not the case; they must be trained on annotated data collected in NE and wearable computers provide opportunities to collect such data, though the process is burdensome. Therefore, we propose an intelligent scoring algorithm that limits the number of user annotation requests through the confidence of predictions generated by the tracking algorithm and automatically annotating data with high confidence. We enhance our scoring algorithm by providing improvements in our tracking algorithm by obtaining context data from nearable sensors. Each specific context of a user bounds the set of activities that can likely occur, which in turn improves the tracking algorithm and confidence. Finally, we propose a hierarchical annotation approach, where repeated use allows us to ask for detailed annotations that differentiate fine-grained differences in ways individuals perform activities. We validate our approach in a diet monitoring case study. We vary the number of annotations requested per day to evaluate model accuracy; we improve accuracy in NE by 8% when restricting requests to 20 per day and improve F1-score of activities by 11% with hierarchical annotations, while discussing implementation, accuracy, and power consumption in real-time use.**

***Index Terms*—Human Activity Recognition, Diet Monitoring, User-Generate Content, Machine Learning, Smartwatch.**

I. Introduction

PERSONAL health monitoring and tracking has become more feasible through ubiquitous, wearable sensors, such as smartwatches [1]. Systems built around these devices can track personal activity and query individuals for understanding behavior and health [2, 3]. For example, tracking food intake enables users to maintain a healthier diet. To train tracking algorithms, extensive amounts of data including sensor readings and their corresponding annotations are needed. However, such algorithms do not often generalize well, as they are usually trained on data obtained under well-defined experimental setups that are annotated by a second party [4, 5]. To overcome this issue, many systems require users to annotate data, which can become burdensome [6, 7]. Therefore, a system that collects such data from wearable devices while intelligently requesting annotations is needed.

Annotated datasets are often obtained through well-defined protocols under constrained settings in laboratories [8]. Consequently, the collected data does not necessarily represent that of natural environments (NEs). Therefore, any algorithm trained on these datasets may perform poorly in NEs. In a posture detection study authors found that although the miss-classification rates with data collected in lab were around 9%, this rate increased to 33.3% outside of lab [5]. To solve this issue, data must be collected and annotated in NEs; however, this is challenging and typically requires extensive user interactions with the system.

In NEs, annotations are usually gathered manually by users. Manual logging of activities, such as dietary intake, activities performed, and locations visited is a burdensome process that often sees greatly reduced user compliance and poor quality due to user recall bias [6, 7]. For example, in remote cardiac rehabilitation, patients self-log their exercise and activities and then communicate their notes to their physicians [9]. Unfortunately, this overhead often causes them to drop out of programs [10, 11]. A study in mental wellness reported reduced adherence rates of users when they were asked to answer an average of eight stress related questions a day [7]. In dietary monitoring, adherence reduction has also been observed [12]. Users forget to log food intake, do not want to log for privacy reasons, or simply find the benefits are not outweighed by the burden of annotating their eating patterns. If systems leverage technology that involves additional passive sensing, observations captured can infer logging and reduce the burden of manual logging by automating most of the process, potentially increasing user adherence.

Different approaches to automatic data annotation have been proposed, aiming at asking users for annotations only when

This paper was first submitted for review on August 1, 2019. It was supported, in part, by the National Science Foundation under grant CNS-1734039, EEC-1648451, and National Institutes of Health under grant 1R01EB028106-01.

R. Solis and B. J. Mortazavi are with the Department of Computer Science and Engineering, 3112 TAMU, Texas A&M University, 77843 USA. (email: roger.solis, bobakm@tamu.edu).

A. Akbari is with the Department of Biomedical Engineering, Texas A&M University, College Station, TX, 77843 USA (email: aliakbari@tamu.edu).

R. Jafari is with the Department of Biomedical Engineering, Electrical and Computer Engineering, and Computer Science and Engineering, Texas A&M University, College Station, TX, 77843 USA (email: rjafari@tamu.edu).

needed. Such algorithms are mostly based on a classification model's probability output. For example, semi-supervised approaches, such as self-learning [13] and co-training [14], use the probability returned by a classifier to guide their labeling process, requiring annotations only when the model's output probability is below a threshold. However, these approaches can overfit and may ignore samples for which the classifier's probability output was high, even though the assigned label was incorrect. Automatic labeling approaches, such as active learning (AL), and other approaches that obtain labels directly from users, like experience sampling methods (ESM), increase accuracy at higher rates [15, 16]; however, such approaches do not limit user interactions, likely impacting adherence [17]. We propose a system for data collection in NEs that intelligently prompts users for annotations when required, minimizing user annotation burden and facilitating longitudinal studies.

This paper is an extension of our preliminary study that analyzed the usage of environmental information and a score that drove annotation requests from users [18]. In this paper, we leverage contextual information obtained from the environmental sensors to enhance the capability of the system in predicting activities. To obtain user annotations, we propose a score that considers not only the confidence and capability of the classifier, but also the natural capability of humans in responding to prompts. Moreover, we show how this system can obtain fine-grained contextual and physical information from the user. Knowing fine-grained details about how or in what situation an activity is being performed could provide vital insight about the user's behaviors and life style, as well as improve recognition accuracy for movements more common to that user's daily life. Additionally, fine-grained information can also provide contextual information for various healthcare applications. Therefore, not only is it important to detect if an activity is happening (*e.g.*, eating), but it is also important to understand details about the activity (*e.g.*, eating soup versus eating a sandwich while walking). We also investigate the overhead of running the proposed algorithms on a smartwatch for data collection and personalization in NEs, evaluating power and computational resource consumption in online system utilization. Finally, we discuss a diet monitoring study we conducted in real world scenarios in the presence of other activities of daily living (ADL), allowing for individual user variation. The contributions of this work are as follows:

- Analysis of a proposed scoring algorithm for driving intelligent annotations.
- Analysis of hierarchical models for fine-grained activity recognition.
- Analysis on the impact of running deep learning models on a device with limited computational power.
- A validation of the proposed platform and algorithms in a diet monitoring study.

II. Related Work

A. Data Collection Platforms

Different collection platforms for activity monitoring have been proposed in the literature. A data collection system using a smartwatch, smartphone, and micro-location data provided by Bluetooth low energy (BLE) beacons was proposed [4]. The authors used custom-made beacons, which potentially impacts scalability. Another system aimed at centralizing the data collection for activity recognition in a single device rather than involving multiple devices and sensors [19]. The platform performs activity recognition but is limited by the collection of data from the singular platform source. We aim for flexibility without the overhead of additional sensors.

B. Annotations

Some approaches to annotations in NEs are automatic while others require users to manually provide labels. Manual annotation requires significant interaction from the user, often resulting in boredom and lack of adherence [7, 20], reducing the response rate by as much as 67% [7]. Previous studies on ESM, reported that users lost interest on providing information when the requests were too frequent [21]. A similar lack of adherence was reported in a diet monitoring study as participants left the novelty period [12]. Additionally, cumbersome approaches for requesting labels can result in incorrectly annotated data or lack of annotated data [20]. AL approaches such as co-training [14], en-co-training [22], and democratic co-learning [23] attempt to automate the annotation process based on the classifier's uncertainty by identifying key samples to label [24]. However, most of the approaches do not pay special attention to limiting the number of annotations requested per day [25]. A solution is needed that reduces user burden by limiting user interaction and automating most of the annotation process. This study augments AL and ESM approaches with a score-based algorithm to identify critical samples and limit user interactions.

C. Diet Monitoring

Audio processing, inertial measurement unit (IMU) data processing, and the development of wearable devices are among the most popular approaches followed for eating detection. In one study, the processing of audio signals captured through smartphones was used to predict family eating moments [26]. However, the goal was to detect eating moments of a group of people rather than per individual. Therefore, the approach was only tested within home settings. Accuracy of 80% was achieved in detecting eating moments using wrist-worn inertial sensors [27]. However, this accuracy was achieved in laboratory conditions.

There are only a few studies that detect eating moments with wrist-worn sensors such as smartwatches in uncontrolled real-world scenarios. IMU data from smartwatches has been used for eating moment detection and F1-score of 0.76 was achieved [17]. However, to train their recognition models, researchers had to collect labeled data by capturing video of each participant in lab settings. Then, they tested their platform in NEs. They asked subjects to hang a smartphone around their necks. The smartphone captured images every 60 seconds and the users were then able to manually label the starting and ending times of the eating moments. This annotation process involved many user interactions as users were required to analyze each image before annotating. Accuracy of 48% was achieved in detecting eating moment with wearable sensors in another study with the ability to distinguish between different types of bites [28].

Some wearables come in the form of necklaces or are placed close to the mouth for detecting chewing events. The EarBit aimed at detecting eating moments by detecting chewing events

in an uncontrolled environment and it achieved F1-score of 0.8 [29].However, video recordings were required to perform labeling. Moreover, this system is not convenient for the users. A similar work built a wearable device composed of a necklace with an embedded piezoelectric sensor that captured vibrations in the users' necks [30]. Whenever users would eat, the sensor would capture the vibrations produced by muscle contractions in the throat. From such vibrations they were able to recognize solids and liquid food swallow events. Similarly, another approach used piezoelectric and accelerometer sensors embedded in eyeglasses for the real-time recognition of eating moments and non-eating moments [31]. However, the development of new wearable sensors should be investigated further on usability and pervasiveness.

III. Methods

We developed a platform and scoring algorithm based on a classifier's uncertainty that drives user annotations to reduce the number of user interactions. To reduce the classifier's uncertainty, we leveraged contextual information from nearable BLE devices. Additionally, considering the health benefits of knowing user activities in detail, we developed a hierarchical approach for detailed activity recognition as a first step towards more detailed questioning for fine-grained annotations. To assess usability for data collection in NEs, we evaluated our algorithms' overhead when running in a smartwatch.

A. Platform for Data Collection

To enable data collection in NEs and to facilitate longitudinal studies, we aimed for a pervasive wearable platform for data collection. Most smartwatches can collect multiple types of sensor data, such as inertial measurement unit (IMU) and heart rate, so we selected this as our data hub. Additionally, most smartwatches are equipped with Bluetooth technologies enabling connection with external sensors, such as chest straps for measuring breathing rate [32] for a wider range of studies. This increases the range of possible sensor data that can be aggregated by the device.

The biggest disadvantage on using a wearable device, such as a smartwatch is the reduced computational power and energy capacity. Such restrictions play an important role as we base our solution in deep learning models, which are effective but are well known for being computationally demanding [33]. Therefore, we performed the training phase of the models on an external computer and then loaded the models on the smartwatch to be used in the testing phase. We then evaluated the computational costs of generating predictions.

The proposed platform is shown in Fig. 1. In this study, we chose to use a Polar M600 watch (an Android-based Wear OS watch) with an IMU sampled at 20Hz and heart rate monitor embedded within sampled at 1Hz, and an Intel Next Unit of Computing (NUC) as the external server component. Additionally, the smartwatch also captured BLE addresses from nearable BLE devices with the aim to collect additional information that can be used to aid the annotation process. Global Positioning System (GPS) data tends to consume a lot of energy and users might feel that their privacy is compromised [34, 35, 36]. BLE beacons, however, can be placed at specific locations as a low energy alternative for

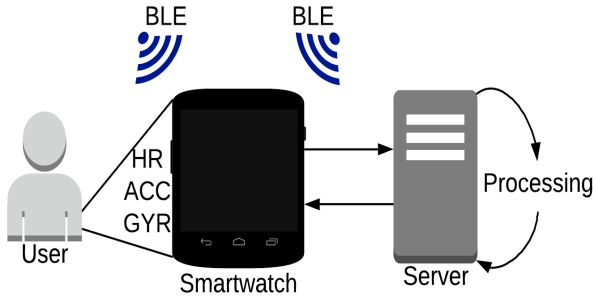


Fig. 1. Our proposed platform composed of a smartwatch for sensor and BLE data collection and an external server for off-line processing and modeling.

location detection [37] without privacy concerns. This platform can continuously collect data for an average of 10-12 hours. We expected users to use the smartwatch during the day to collect and annotate data. At the end of the day, users would plug the watch into the NUC for charging and retrain models in the NUC overnight so that in the morning the newest changes were already on the watch.

B. Utilizing Environment Context for More Accurate Models

Considering the benefits of context-aware approaches, available environmental information should be leveraged to reduce user annotation burden through increased recognition accuracy. Robust models which capture the complex nature of data collected in daily living decrease this uncertainty. For example, if a user is in the car, the

probability of eating is lowered. Here, we used environmental data to cluster user activities, to build context-specific models where the number of activities we aimed to recognize were reduced if the likelihood of some activities in some contexts was extremely low. In fact, when the probability of an activity in a specific context is low, we can remove that from the set of classes that should be detected by the classifier. In other words, contextual information helps reduce the search space of activity classification by ignoring the activities that are not probable in a certain context. To obtain environmental data, we used nearable sensors, namely, addresses broadcasted by nearable BLE devices.

Environment was identified by detecting consistent, repeated BLE addresses. Hence, the usefulness of this contextual information in reducing the search space of activities highly depends on the user's routine and training data of the context. If the user repeatedly performs certain activities in a specific location, then the context will be very effective in reducing the search space. However, if the user visits new places more often, then the context will be unknown and cannot be used for reducing the search space effectively.

Every day the system was exposed to several unique BLE addresses. So, to recognize a repetitive pattern in BLEs, we considered their co-occurrence, with an assumption that in certain places where there are static devices, they tended to be observed with each other in all visits to that place. Therefore, the co-occurrence of addresses meant that they likely belonged to the same location. Subsequently, we stored the BLE observation data in a graph structure. In this graph, every vertex represented a BLE address, and edges represented the number of co-occurrences at different locations. Given a list of BLEs seen at a time frame, we considered all pairs of BLEs and

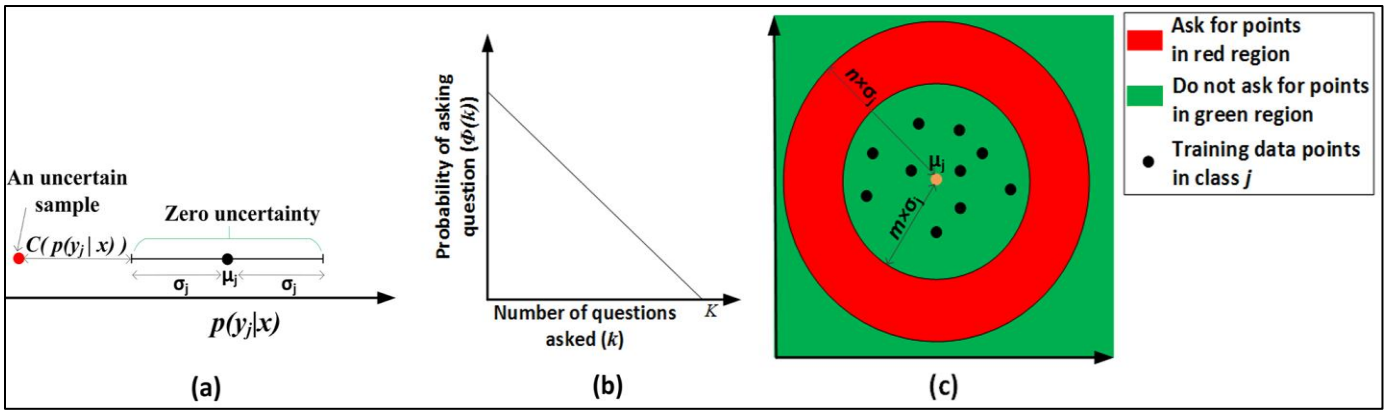


Fig. 2. (a) Data samples with uncertainty that is too low or too high are more likely to request annotation. (b) Probability of asking questions decreases as the system gets closer to the limit of questions. (c) The label is asked for the points that are far from the centroid of training data but not for outliers.

checked whether they were seen together previously and propagated what the location was based on this relationship.

We clustered the activities based on their locations and their inherent similarity so that we could model similar activities together. Each specific context was associated with a certain set of activities. Therefore, we learned contexts that maximized the likelihood of the observation of those specific activities. We learned unknown contexts and the distribution of each user's specific activity likelihood in each of these contexts to improve event recognition. This was determined in an expectation maximization (EM) approach where the mixture components were context-specific activity recognition models learned on annotations identified by the user, and the activity detection improved under that reduced search space. This was achieved by defining the log-likelihood of observing activities given contexts as:

$$log\ P(ACTIVITY|X,\theta) = \sum_{n=1}^{N} log\ P(activity_i|x_i,\theta) = \sum_{n=1}^{N} log \sum_{c=1}^{N_c} P(activity_i, c_i = c|x_i,\theta) \quad (1)$$

where θ are the mixture component parameters, N is the number of data samples, and c is the number of expected clusters (contexts) to which each data point belongs.

In this unsupervised formulation, clusters did not necessarily contain environmental information. To inject this information into the latent variables, the E-step of the EM algorithm was regularized with the Kullback-Leibler (KL) divergence [38]. The KL divergence was used as a measure of similarity between two distributions to penalize the posteriors that were less similar to the context distribution collected from user annotations. Therefore, the E-step in EM algorithm was changed as:

$$q^{k+1} = \underset{\theta}{argmax}\ E_{q^{k+1}}\ log\ P(X,C|\theta) - \lambda KL(q|context) \quad (2)$$

which promoted posterior distributions more similar to environmental distribution in order to inject environmental meaning into them. The M-step remained as standard in EM to improve activity recognition tasks [39].

To implement our context-aware system, two separate neural networks were used. One network was dedicated to environment estimation. The other network performed user activity recognition given sensor readings and the location estimated by the first network. Each iteration had two substeps: environment estimator training and environment-aware classifier training. In the first substep, the environment estimator was optimized, and the environment-aware classifier was fixed. Therefore, in the first substep the environment estimator network optimized the location distribution given the environment-specific model. In the second substep, the environment-aware classifier network was optimized given the distribution over locations, which was the output of the environment estimator network.

The number of different environments was considered a hyperparameter that represented the number of mixture models. In our supervised approach, we chose the number of environments to be the number of places for which each user had annotated data. In (2), λ was the penalty parameter used to constrain the posterior distribution over environments. This varied the impact of environment on the event recognition, allowing activities that do not belong to certain environments to still be detected. We performed a grid search on λ between $[0-10]$ in increments of 1 to evaluate the improvement in event recognition. We limited ourselves to 10 as we observed that the largest impact came when λ changes from zero to one, and we observed that the impact lessened with each subsequent increase, likely due to our coarse-grained definition of location.

C. *Intelligent Annotation*

With an accurate, environment-aware classifier, we then focused on intelligent annotations to enhance the use of wearable sensors in NEs. To increase the accuracy through a better training dataset, the system needed to have label information for critical and novel samples. These novel observations were created by variations in the way an activity could be performed, or an event could happen. This challenge became more significant when algorithms were trained on the data of certain users but used by a new user. Thus, the system needed to update over time. To address this issue, the system must have access to the labels for those novel observations for retraining. This information was obtained by asking the user for data annotations. Those annotations were then used to improve accuracy and generalizability of the system through retraining.

We developed a score for identifying the importance of a sample by taking into account the following parameters: 1) the classifier's uncertainty $C\big(p(y|X)\big)$, 2) the number of questions

that have already been asked in a day $\phi(k)$, 3) the confusion matrix γ, and 4) the similarity of the input data to the previously observed data $D(X)$. The proposed score S is calculated as:

$$S = \alpha_1 . C\big(p(y|X)\big) + \alpha_2 . \phi(k) + \alpha_3 . \gamma + \alpha_4 . D(X) \quad (3)$$

where α_{1-4} are hyperparameters that determine the importance of each score parameter on the final decision. We performed a grid search to choose these hyperparameters by changing the values in the range of 0 to 1 with steps of 0.25, and we found the best set of parameters given our training data to achieve the highest accuracy in cross validation performance.

The intuition behind each parameter in (3) are as follows. Very low confidence, or very high confidence indicated that the predicted label could be wrong because of noise or overfitting. We calculated the mean and standard deviation (SD) of the output of the classifier $p(y|X)$; the probability of each activity to be correct, for all classes over all training data. For class j, μ_j^y is the mean and σ_j^y the SD of the probabilities given by the classifier over the whole training data. For a new sample, the value of $C\big(p(y|X)\big)$ was calculated as:

$$C\big(p(y|X)\big) = \begin{cases} 0, & if \left|p(y|X) - \mu_j^y\right| < \sigma_j^y \\ \left|p(y|X) - \left(\mu_j^y + \sigma_j^y\right)\right|, & otherwise \end{cases} \quad (4)$$

illustrated in Fig. 2-a, where $c = argmax\ p(y|X)$.

To reduce the user burden, the number of questions that could be asked per day was restricted to K. If the system had already asked the user several times within a day, then asking more questions should be done less frequently and only when really needed. The probability of asking questions is calculated as:

$$\phi(k) = \frac{K - k}{K} \quad (5)$$

where k is the number of questions already asked. When k is large, the probability of asking questions is reduced as depicted in Fig. 2-b.

Classifier's uncertainty cannot always be trusted [40], so it should not be the only deciding factor. For classes which the system usually predicts correctly, fewer user annotations were needed but some might still be necessary. However, for classes on which the system makes mistakes, there was a need to request user annotations more frequently. This is calculated as:

$$\gamma = \frac{TP_c + FN_C}{TP_c} \quad (6)$$

where TP_c is the true positive and FN_C is the false negative for the class c, which is the decision of the classifier $c = argmax\ p(y|X)$.

Finally, if the input data was not like the data of any detected class c, the sample could potentially be an unseen data type, and a prompt for user annotation would occur. On the other hand, that data could be an outlier, not requiring user prompting. To address this complication, as depicted in Fig. 2-c, the method analyzed the Euclidean distance between the current sample X and the mean point of the training data that belongs to the same class μ_j^X as:

$$D(X) = \begin{cases} 0\,, & if\ \left||X - \mu_c^X|\right| < m * \sigma_c^X, or \\ & \left|\left|X - \mu_{neighbor}^X\right|\right| > n * \sigma_{neighbor}^X \\ \dfrac{\left||X - \mu_c^X|\right|}{max\left(\left|\left|X - \mu_j^X\right|\right|\right)}\,, & otherwise \end{cases} \quad (7)$$

where μ_c^X is the mean of all training data that belong to class c, σ_c^X is their SD, and $c = argmax\ p(y|X)$ is the decision of the classifier. The closest mean class point to the sample X is $\mu_{neighbor}^X$ and its SD is $\sigma_{neighbor}^X$. The maximum distance from the input X to the mean of any class is given by $max\left(\left|\left|X - \mu_j^X\right|\right|\right)$ and is used to normalize the value of $D(X)$. Here, m and n are hyper parameters that are chosen empirically. In our study m = 1 and $n = 5$. In (7), for a sample that is close to the other samples of the same class, the probability of asking is then reduced. Similarly, the probability is reduced for a sample that is farther than all class boundaries, as it was a candidate of being an outlier new activity.

After calculating S, it was compared to a constant threshold parameter Th. The instances for which the score was higher than the threshold were nominated for user annotation. However, a single windowed prediction may not be accurate. Additionally, most ADLs, such as eating, occur as longer, cyclical, repetitive motions. Thus, S was calculated for a series of consecutive windows. If S was higher than Th for N times within the last T windows, the system asked the user for that specific time interval. In this work, T was set to 10 (30 seconds) and N was set to 5.

The maximum number of annotations requests a user could get in a day is indicated by K. Assuming each day of data collection to be approximately 12 hours, the number of annotations K was set empirically to 20. This was determined since a user would likely not annotate data more than 1 to 2 times every hour. The impact of K on classifier accuracy will be discussed further in the experimental evaluation. The value for Th on which the algorithm determined whether to ask for an annotation was heuristically set to 0.4, as most scores fell within the range (0.2, 0.5). Higher values discouraged the algorithm from requesting annotations, causing fewer annotations, which would cause a reduction in the classifier's accuracy. On the other hand, lowering the threshold caused the algorithm to request annotations on unimportant samples, limiting accuracy improvements.

D. Activity Recognition Model

We discussed the importance of context-awareness on obtaining more accurate models for reducing uncertainty. In addition, personalization through intelligent annotation not only offered increased model accuracy, but it also enhanced user annotation experience. In this section, we explain the details of the activity recognition models proposed in this study.

We evaluated convolutional neural networks (CNN) and long short-term memory (LSTM) networks as both are capable of automatic feature extraction [41]. Training such networks is known to be computationally demanding. However, as this process was completed in an external device in our proposed platform, it seemed reasonable to consider both approaches, and compare impact of inferences on smartwatch performance. We

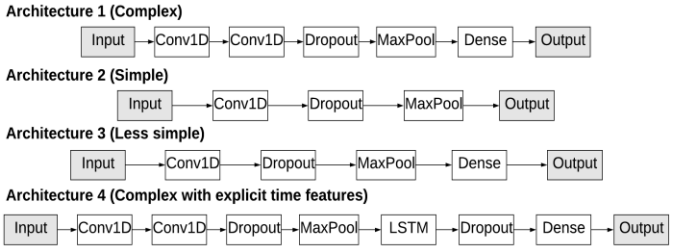


Fig. 3. Architecture 1 was designed to extract low and high-level features. Architecture 2 and 3 were designed as an alternative to Architectures 2 and 3 with aims at reducing the model's parameters. Architecture 4 was designed with aims at explicitly extracting time related features.

tested four different architectures for activity recognition, illustrated in Fig. 3.

Architecture 1 is one of the most commonly used in CNN approaches. The first two convolutional layers extracted low and high-level features respectively, and then the dropout layer reduced the model's likelihood of overfitting. The max-pool layer reduced the model parameter complexity, which is of particular importance as the number of parameters correlates to the computational power needed for running inferences. Architectures 2 and 3 were proposed as low-complexity alternatives to Architecture 1. Architecture 4 was like Architecture 1, with explicit extraction of time features through the LSTM layer.

To select the best architecture, we trained each of them using the raw sensor data and evaluated their performance. To evaluate how the models would behave in future data, the training set was composed of the first weeks of collected data (varying from 2-3 weeks), and the test set was the last week of data. The architecture selection was accomplished by training each of the architectures using the data collected in the diet study. To feed the data to the models we used a sliding window of 6 seconds with 50% overlap on the 6 sensor channels from accelerometer and gyroscope. Thomaz et al. found that the optimal window for the detection of eating moments was 6 seconds [17], so we used this timing for our extraction. Each window of data was then assigned its label and the window of data along with its label was fed into each model. For each architecture, we evaluated different batch sizes and epoch numbers, and selected the architecture that showed best performance. For hyperparameter tuning we followed a grid-search approach.

1) Fine-grained Annotations

Knowing fine-grained details about how or in what situation an activity was performed could provide vital insight about user data. By fine-grained annotations/activities we mean various versions of an activity that can be performed in different situations/under different contexts. For example, in eating activity, different variations are eating while sitting and eating while walking; or it could be eating soup versus sandwich. We call the fine-grained labels subclasses of a parent activity. Although those subclasses are distinguishable due to their inherent differences, they share several similarities that makes it difficult to distinguish them as shown in Fig. 4. Moreover, given their similarities, they can fall into a bigger parent class. Formally, for input signal, X, which is the vector of sensor

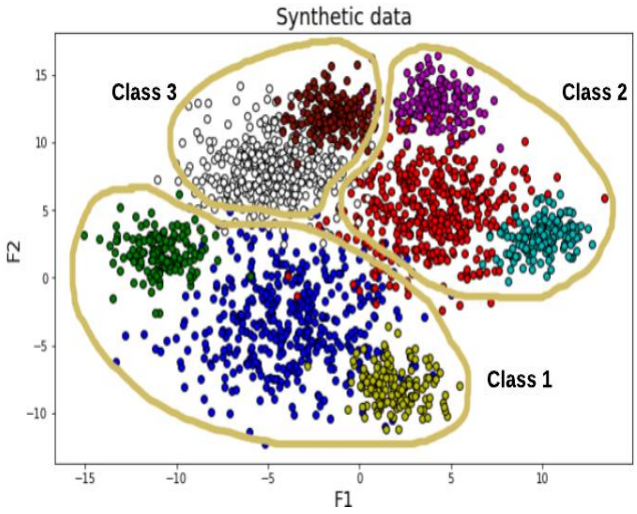


Fig. 4. Synthetic data. There are three main classes: class 1 has 3 subclasses, class 2 has 3 classes, and class 3 has 2 classes.

readings, we assume there is a parent label $y \in \{P^1, P^2, \dots, P^L\}$ available, where P^i is i^{th} parent class (i.e., activity) and L is the total number of activity classes. Moreover, we assume there could be a subclass annotation available for this sample as $y' \in \{S_1^1, \dots, S_{K_1}^1, S_1^2, \dots, S_{K_2}^2, \dots, S_1^L, \dots, S_{K_L}^L\}$ where S_j^i is j^{th} subclass of i^{th} parent class, and K_i is the total number of subclasses of the i^{th} parent class. Depending on the availability of the fine-grained (i.e., subclass) labels and regardless of the specific type of activity, the proposed hierarchical model can improve detection of fine-grained activities by first detecting the parent classes. Later in this section, we try to show the use case of this model with a synthetic dataset in addition to our dietary dataset as well as two publicly available datasets of activity recognition with various parent-subclass labels. It should be noted that the range of the granularity that the system can handle depends on the distinguishability of the motion signals of the activities. For example, consider the subclasses of eating activity. In this case, for subclasses including eating while sitting and eating in rush, for example during walking, the movement patterns are distinguishable given the eating has been detected as the parent class. Thus, after successful detection of the parent class, the system can detect those sub-classes well even among a large set of various activities. On the other hand, detecting fine-grained subclasses such as eating soup versus eating pasta would be very challenging by using merely hand motion signals since those sub-class have very similar motion patterns.

In our dietary monitoring case study, eating detection is essentially detecting the movement of the hand, which is augmented by the contextual information about users' environment. In eating detection, eating versus no eating would be the parent classes and eating in rush or eating while sitting would be subclasses. Detecting such subclasses directly from the sensor data was challenging due to sparsity of labels and person-to-person variation. Moreover, the patterns of the signals in the subclasses could be very similar to each other as in certain cases they were a composition of multiple activities. For instance, eating in a rush could be a composition of eating and walking activities, which makes its movement pattern very similar to the walking. To overcome this challenge, we augmented our system with a hierarchical classification and

annotation approach. The classification started by detecting the parent class from user's motion data. During the personalization phase, after detecting the parent label, if confident, we asked the user more specific labeling questions. Understanding the parent activity was critical to ask the user right questions, if needed. In the testing phase the output of the parent class detection, the primary classifier, was leveraged to augment the model for detecting the subclasses. The output of the primary classifier (i.e., parent class detection), along with the motion signals were fed as inputs to the secondary classifier to detect the subclasses more accurately. These probabilities were used in the scoring algorithm, and for our case study we limited the hierarchical labels to two levels.

To validate the hierarchical approach, we performed two experiments: one on synthetic data and one on a publicly available data. The synthetic dataset in Fig. 4 had two features and was composed of 3 parent classes, each with 2 or 3 subclasses. The goal was to understand if the hierarchical classification could improve the performance compared to the case of classifying the subclasses directly. We defined a simple network architecture with 2 hidden layers of 16 and 8 neurons respectively with *relu* activation as the primary classifier for detecting the parent class. The output of this network used *softmax* activation function that generated three values between zero and one corresponding to the probability of each of three parent classes. The secondary classifier, which was the one to detect subclasses, had the same architecture as the primary classifier. However, in addition to two features, the three probabilities generated by the primary classifier were fed as input to this classifier. We performed a 5-fold stratified cross-validation with balanced classes. The testing set was composed of 1-fold and the training set was composed of the remaining folds.

In addition to the synthetic data, we tested this approach with a publicly available dataset of human activity recognition called Actitracker that contained real-world ADL data captured by a smartphone from multiple subjects [42]. We used the data from the first 10 subjects. The labeled data included sitting, standing, walking, jogging, walking downstairs, and walking upstairs. We created three parent classes out of this data: an ambulatory class containing walking and jogging as subclasses; a sedentary class containing sitting and standing as subclasses; and a stair climbing class containing walking upstairs and walking downstairs as subclasses. For this dataset, the primary classifier contained three convolutional layers for automated feature extraction followed by two dense layers for combining the features and mapping them to the parent classes. The secondary classifier had the same architecture, aside from accepting the primary classifier output as extra inputs.

Finally, we tested the hierarchical approach in the data obtained during our study to improve the detection of detailed activities through the knowledge provided by the parent activities. We followed the same approach as for the Actitracker experiments, but with Architecture 1 from Fig. 3.

2) Deep Learning in Smartwatches

One of the main concerns about running deep learning models in smartwatches is the computational overload. We proposed to perform the training process offline. However, we still needed to evaluate the computational impact of running real time model inferences. We wanted to evaluate the CPU usage and energy impact of running our best model with different hyperparameter configurations. Running an inference involved a series of mathematical operations using the model's parameters. The parameters and number of operations depended on the hyperparameter configurations of the models, such as the number of filters in a convolutional layer or the number of neurons in a hidden layer. Three hyperparameter settings were evaluated: low, medium, and a high number of parameters. The CPU and energy impact were monitored and recorded through Android Studio's monitoring tool. Additionally, we evaluated the number of floating-point operations per second (FLOPS) that each model requires and the average time it took to the model to finish an inference.

IV. Experimental Evaluation

A. Case Study

In this work, we performed a diet monitoring study for which our main goal was to improve eating detection through intelligent annotation. The study was divided in two parts. In the first part of the study, only eating activities were collected. Eating activity was collected by asking participants to annotate every time they eat. For the second part, *eating*, *sitting*, *walking*, and *exercising* activities were collected by asking participants to annotate data every 15 minutes. A total of 10 participants collected data for 4 weeks each. However, some participants had difficulties annotating data on a regular basis, resulting in annotation noise. Therefore, only data from 6 participants was used in the first part of the study, and from 5 in the second part. This study was reviewed and approved by the Texas A&M University Institutional Review Board (IRB#2018-0998D).

B. Environment Utilization and Intelligent Annotations

We evaluated the performance of our activity recognition framework in the detection of eating moments. As mentioned in Section III-B, an important component of the proposed framework was the utilization of environmental information to reduce the need for user annotations. Our models achieved an overall accuracy of 76% when detecting eating moments without using environmental information or annotations. The overall accuracy, however, increased to 82% when adding environmental information, specifically location leveraged from BLEs. Over the course of the study, the participants visited an average of 9 different locations when eating, including restaurants, office, and home.

To investigate the impact of the opportunistic collection of labels using our proposed score, the data for each user was divided into three parts: training, feedback, and testing sets. The training data was used to train the environment-aware model and feedback data was used to obtain annotations. The threshold was empirically chosen to $Th = 0.4$. Based on this score, the system requested users for the label and then used that label to retrain the model. Our models achieved an overall accuracy of 84% when using environmental information and user annotations, resulting in an improvement of 2% from the environment-only approach, and about 8% from the standard model. The number of annotations was limited to 20 per day. This improvement was gained as the model was modified to learn the correct labels of previously uncertain samples.

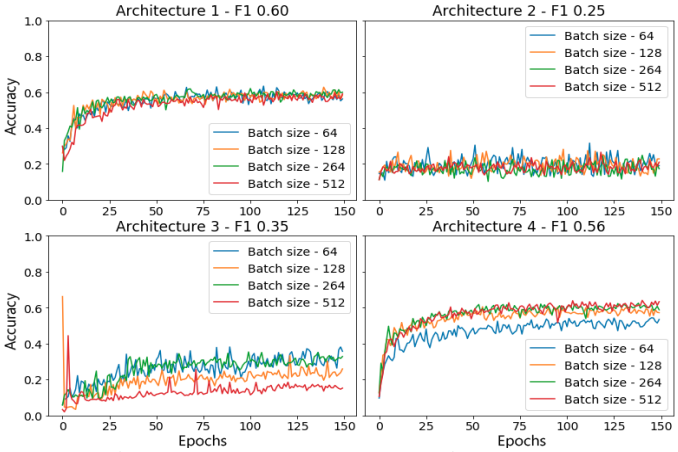


Fig.5. The figure shows the accuracy results and f1 score (micro-averaged) for each architecture. Architectures 1 and 4 perform better as they are more complex and therefore extract better features out of the raw sensor data.

C. *Automatic Feature Extraction*

We also investigated the effect of the neural network architecture on activity recognition. The goal was to choose the best one to abstract information from the raw data. Fig. shows the results when training each architecture, as explained in Section III-D, with different batch sizes. As expected, Architectures 1 and 4 performed the best as they were designed to extract complex features at the expense of having more parameters. In our experiments, architecture 4 is slightly better than architecture 1 although we expect it to perform significantly better due to explicit modeling of temporal characteristics. Architecture 4 has more parameters compared to the architecture 1 while the dataset is not that huge and overfitting could be one reason behind this. We expect to see more significant improvement in architecture 4 compared to 1 if the size of the training dataset increases. In addition, with pure CNN (architecture 1), a number of time characteristics are still extracted by processing windows of 6 seconds of data. Therefore, although architecture 1 does not have an explicit temporal modeling component, CNN can to some extent compensate for this by implicitly modeling certain temporal characteristics of the signals.

Their counterparts, Architectures 2 and 3, did not manage to learn through the training process despite being simpler. Architecture 1 had fewer parameters with similar performance to Architecture 2, and therefore it was chosen for our platform. Finally, a grid-search approach for hyperparameter tuning was followed. Considering the most important hyperparameters from each layer in Architecture 1, we list the ones covered in Table 1 and tested all the possible combinations resulting in a total of 144 models being evaluated. Fig. 6 shows the best 5 hyperparameter configurations. It should be noted that Based on our experiments, although increasing the layers from architecture 1 to 2 boosts the accuracy, increasing that beyond architecture 1 does not significantly improve the accuracy. Moreover, increasing the number of layers or neurons per layer will make the model more complex to be implemented on the smartwatch, or any other wearable computer, with limited computational capabilities. For all the trainings, Adam optimizer with learning rate of 1 was utilized.

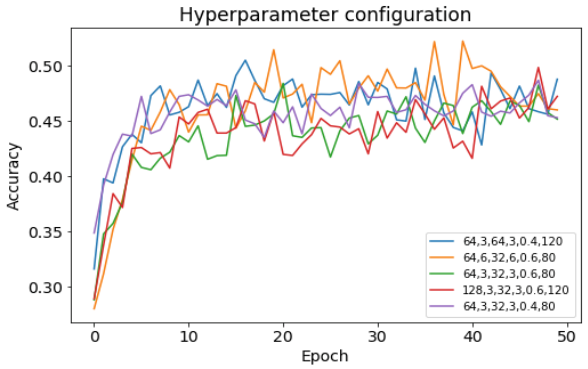


Fig. 6. The best 5 configurations are shown. In the legend are the values for each of the parameters as follows: Number of filters ConvLayer 1, Number of filters ConvLayer 2, Kernel size ConvLayer 1, Kernel size ConvLayer 2, Dropout rate DropoutLayer, Number of neurons DenseLayer.

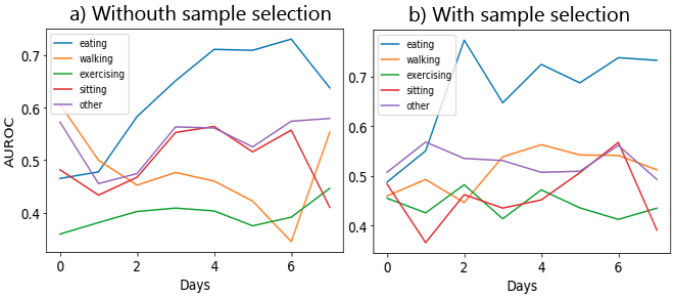


Fig. 7. AUROC scores per activity. Day 0 correspond to the baseline model (no annotations). Each day indicates the number of days used for retraining. a) shows the results obtained when retraining with all the samples available within a day whereas b) shows the results when retraining with critical samples

D. *Accomplishing Personalization*

After defining the architecture to be used, we wanted to evaluate the capabilities for personalization to each user. Using the data obtained in our diet monitoring study, we progressively trained models by adding data in a simulated daily basis. In a first approach, we assumed that all the data from the annotations set would be available and we used it for retraining. Fig.7-a shows the *AUROC* values for each of the activities of interest. For most activities there was no consistent improvement on the *AUROC* values as we added data from more days. We observed that for *eating*, there was a clear improvement when we added data from days 1 through 6, however a drop appeared after the data for day 7 was added, likely because of the presence of noise or some confounding label. Similarly, for *sitting* there was a decrease when the data from the last day was added. Prediction of *walking* appeared to worsen as more data was added, because of the combination of activities, such as eating while walking, and the user choosing to label eating. As no good AUROC improvements were found, we did not further evaluate F1 scores at optimal thresholds.

TABLE 1
HYPERPARAMETERS FOR GRID SEARCH

Hyperparameter	**Values**
Num of filters ConvLayer 1	128, 64, 32
Num of filters ConvLayer 2	3,6
Kernel size ConvLayer 1	128, 64, 32
Kernel size ConvLayer 2	3,6
Dropout rate DropoutLayer	0.4, 0.6
Num of neurons DenseLayer	80, 120

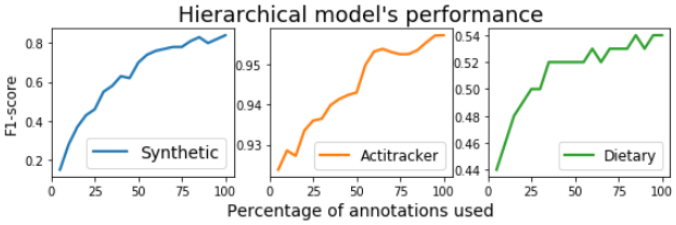


Fig. 8. Performance improvement per dataset as more annotations are used.

Additionally, we attempted to recreate this example by selecting the most important labels, to evaluate the impact of the scoring algorithm on the incremental learning. After training the baseline model, we computed the average probability output of our baseline model for each given class on the training set. When a new sample arrived, its probability was estimated by the model and we checked if the probability was within 1 standard deviation (SD) from the baseline average. If the data was within 1 SD, we assumed the new sample was not very different from what we had seen so far, therefore, it might have not been worth using for retraining. On the other hand, if the probability output was outside 1 SD from the average, we assumed that this new sample was considerably different from what we had seen so far, making it worth annotating and using for retraining, similar to how the label asking score decided to request labels. The results are shown in Fig. 7-b. The selective solution stabilized some activities. However, it did not work for all of them, suggesting that intelligent selection is possible but still needs further analysis to determine if the confounding hierarchical labels continue to degrade performance.

E. Fine-grained Annotations

We evaluated our hierarchical model's effectiveness in predicting subclasses (fine-grained activities), by comparing it to a simple model trained directly to predict the subclasses that impacted the prior models, as the hierarchical model was the culminating algorithm of using context-aware recognition, intelligent (hierarchical) labeling, and deep learning optimizations. We compared the models when trained in a best-case scenario, assuming all annotations were available. In the synthetic dataset, there was an overall improvement of 1.6%, in the Actitracker dataset the improvement was of 1.8%, and in our dietary dataset the improvement was of 11.0%. The amount of improvement in the hierarchical model compared to the case of using a simple model trained directly to predict the fine-grained labels (i.e., subclasses) is associated with the complexity of those subclasses, more specifically the similarity between their motion patterns. In our dietary dataset, the subclasses include the same activity (eating) performed in different situations that makes it difficult to detect them directly with a single model. In this case, using the hierarchical model results in significant improvement. However, in other datasets, since the data of the subclasses are inherently distinguishable, the improvement gained by the hierarchical model is marginal in these experiments.

We also evaluated the relationship between the number of annotations and the model's F1 score by analyzing the F1 score changes as more annotations become available. Therefore, we trained our model by progressively adding 5% of the total training data in a total of 20 iterations, capturing the F1 score per iteration for each dataset as shown in Fig. 8, evaluating the

TABLE 2
ENERGY CONSUMPTION AND ESTIMATIONS

Config.	Energy usage	MFLOPS	Running time (µs)	Estimated duration (Hours)
Baseline	Light	-	-	10 -12
1	Light	375.53	3772	7.14 - 8.5
2	Light	442.18	8950	3.8 - 4.6
3	Light	811.93	17552	3.3 – 4

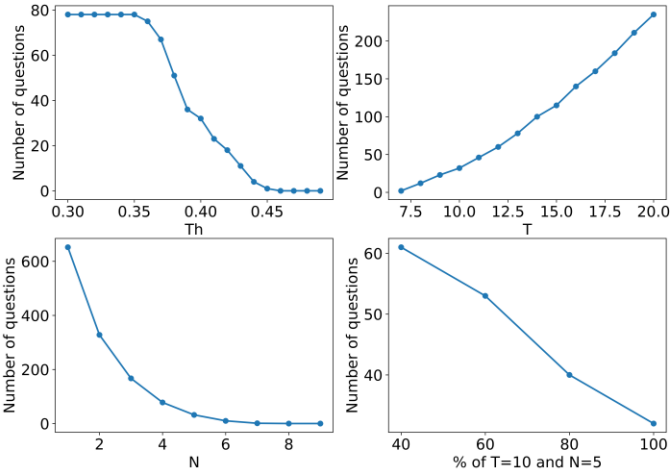


Fig. 9. Effect of threshold (TH) and parameters T and N on the number of annotation requests

impact of providing more labels through different scoring algorithm thresholds and through longer-term usage of a system by adherent users, either of which could result in additional labels for algorithm re-training. The results suggest that our hierarchical approach detected subclasses more accurately than a model trained directly on only the subclasses. The results indicate that it is possible to get better insights of user's detailed activities by using additional information hierarchically rather than trying to identify such sparse activities directly. Moreover, we chose the threshold parameter Th to limit the number of the annotation requests to be 20 per day in order to avoid bothering the user too much while achieving a reasonable personalization accuracy. We assume that these annotation are collected to improve model personalization accuracy, and with the smart annotation, we try to get the labels for the most important samples to ensure this. Fig. 8 shows the effect of the number of annotation requests on the ultimate performance of the personalization. It should be noted that changing the threshold Th explained in Section III-C highly impacts the number of questions as shown in Fig. 9. With larger threshold the number of questions asked from the user will be decreased due to increased restriction in selecting the samples, and it might be the case that the system does not ask all K possible questions during a day. Therefore, based on Fig.8, increasing the threshold can negatively affect system's performance. On the other hand, decreasing the threshold leads to asking more questions which can become burdensome for the end users. In fact, the threshold can be used to control the rate of asking questions from the users, for instance, based on their preferences or their context (location, time, etc.). Based on this figure, we limited the number of requests to 20 as a good number to balance between the accuracy and user's burden.

Another factor that affects the number of questions asked per day from the user is T and N as described in Section III-C. To

decide whether asking the user to provide annotations, we look at the score generated by intelligent annotation module over the last T windows instead of only one window. In this study, we chose *T=10* to look over *30* seconds of data to make that decision since it is an appropriate interval for eating activity (the main focus of this paper), which has longer cyclical hand motions. Within those 10 windows, if the score is above the threshold for at least $N = 5$ times, we consider that as an uncertain or unknown period and then ask the user for annotation. Fig. 9 shows the effect of changing these parameters on the number of questions asked. Increasing values of T and N at the same time (bottom right diagram in Fig. 9) leads to asking less questions since the system becomes more restrictive, and decreasing them simultaneously would lead to asking more, which can consequently increase false positives especially when looking at activities like eating that have longer period of motion compared to other activities such as walking. Therefore, as increasing these values decreases the number of annotation queries, it could negatively affect the performance of the system as Fig. 8 illustrates. Furthermore, per Fig. 9, keeping $N = 5$ constant and increasing T leads to increased questions while keeping $T = 10$ constant and increasing N reduces the number of annotation requests.

F. Deep Learning in Smartwatches

We discussed the benefits in terms of accuracy and model parameters complexity that Architecture 1 has versus Architectures 2, 3, and 4 in Section IV-C. To evaluate the computing impact of Architecture 1 in the smartwatch, we designed an experiment to evaluate the CPU and energy usage. We tested 3 different hyperparameter configurations for Architecture 1: configuration 1 included 32 filters in the convolutional layers and 50 neurons in the hidden layers; configuration 2 included 64 filters and 80 neurons; configuration 3 included 100 filters and 100 neurons. We evaluated versus a baseline without running any inference, only collecting IMU, BLEs, and HR data. We used Android Studio's tool Android Profiler that gives estimates on energy consumption. Table 2 shows the results of this analysis.

Android Profiler categorized the energy consumption as *light* regardless of the configuration chosen. Additionally, Table 2 shows the MFLOPS for running each configuration as well as the time it takes to run an inference. As FLOPS are related to CPU and energy usage, we observe that the hyperparameter configuration correlates to the power impact. Configuration 1 provided a good balance between accuracy and power impact. However, this parameter configuration will depend on the complexity of activities that the model aims to predict. In other words, different projects might require different configurations and therefore, might result in different power impact. Long-term battery life projections are shown in Table 2. Using a MediaTek M2601 (the Polar M600's ARM Dual Core 1.2 GHz Cortex A7 CPU) runs for up to 12 hours with no inferences. Table 2 shows further estimated durations for each of the model configurations running on the smartwatch.

With aims at providing a tool capable of accomplishing smart annotations in real time in a smartwatch device, we also evaluated the implications of running our scoring approach in the smartwatch. Computing the score S took about 67.29 MFLOPS when using Architecture 1 (Fig. 3), making inferences, and computing uncertainty $C\big(p(y|X)\big)$. Considering that some processing was done offline, the scoring approach took about 28 milliseconds, including the inference run by the model, suitable for a real time implementation.

We can further improve the efficiency and maximize the total time we can continuously collect data while running inferences. We ran inferences continuously, but we could instead run them opportunistically, such as every time IMU significantly changed. Given that some activities are stationary, the readings for both the accelerometer and gyroscope might remain constant. We can simply not run an inference if we are confident enough the activity and context have not changed, which is likely as the sensor data has not changed. We could also opportunistically turn BLE collection on/off. BLE addresses were collected with the purpose of providing contextual information but such contextual information might not always be needed. BLE data collection could be actively turned on only if the model's output probability was low, or if the probability for two or more classes were close to each other and the context provided by BLEs could serve as a tie breaker.

V. Conclusion

In this paper, we introduced a wearable data collection platform that enables the smart collection and annotation of user data. We introduced a score driven by the model's predictive behavior composed of the model's confusion matrix, a measure for uncertainty, a metric of similarity, and a limit on the number of requests. We also proposed an approach to utilize environmental information through nearable BLE devices. Our score along with the environment utilization resulted in a gain of 8% in detection accuracy of eating moments. Additionally, we evaluated different neural network architectures for accurate prediction of activities. We also proposed a hierarchical model for more accurate prediction of fine-grained activities, which sets up a path for more detailed questioning and better user tracking. We evaluated the hierarchical model in 3 different datasets. The results suggested that our proposed model predicts fine-grained activities more accurately. Finally, we provided an online evaluation of running deep learning models in our platform as well as computing our proposed score for real time intelligent annotation through the smartwatch, presenting energy impact as well as FLOPS. We validated our work through a diet monitoring study on which our goal was to achieve accurate detection of eating moments for better nutrition management.

References

[1] E. Jovanov, "Preliminary analysis of the use of smartwatches for longitudinal health monitoring," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, 2015.

[2] V. Ahanathapillai, J. D. Amor, Z. Goodwin and C. J. James, "Preliminary study on activity monitoring using an android smart-watch," *Healthcare technology letters*, vol. 2, pp. 34-39, 2015.

[3] E. Årsand, M. Muzny, M. Bradway, J. Muzik and G. Hartvigsen, "Performance of the first combined smartwatch and smartphone diabetes diary application study," *Journal of diabetes science and technology*, vol. 9, pp. 556-563, 2015.

[4] A. Filippoupolitis, W. Oliff, B. Takand and G. Loukas, "Location-enhanced activity recognition in indoor environments using off the shelf smart watch technology and BLE beacons," *Sensors*, vol. 17, p. 1230, 2017.

[5] F. Foerster, M. Smeja and J. Fahrenberg, "Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring," *Computers in Human Behavior*, vol. 15, pp. 571-583, 1999.

[6] L. E. Burke, J. Wang and M. A. Sevick, "Self-monitoring in weight loss: a systematic review of the literature," *Journal of the American Dietetic Association*, vol. 111, pp. 92-102, 2011.

[7] R. Wang, F. Chen, Z. Chen, T. Li, G. Harari, S. Tignor, X. Zhou, D. Ben-Zeev and A. T. Campbell, "StudentLife: assessing mental health, academic performance and behavioral trends of college students using smartphones," in *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing*, 2014.

[8] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *International Conference on Pervasive Computing*, 2004.

[9] H. M. Dalal, A. Zawada, K. Jolly, T. Moxham and R. S. Taylor, "Home based versus centre based cardiac rehabilitation: Cochrane systematic review and meta-analysis," *Bmj*, vol. 340, p. b5631, 2010.

[10] J. C. Bollen, S. G. Dean, R. J. Siegert, T. E. Howe and V. A. Goodwin, "A systematic review of measures of self-reported adherence to unsupervised home-based rehabilitation exercise programmes, and their psychometric properties," *BMJ open*, vol. 4, p. e005044, 2014.

[11] M. Varnfield, M. Karunanithi, C.-K. Lee, E. Honeyman, D. Arnold, H. Ding, C. Smith and D. L. Walters, "Smartphone-based home care model improved use of cardiac rehabilitation in postmyocardial infarction patients: results from a randomised controlled trial," *Heart*, pp. heartjnl--2014, 2014.

[12] N. D. Peterson, K. R. Middleton, L. M. Nackers, K. E. Medina, V. A. Milsom and M. G. Perri, "Dietary self-monitoring and long-term success with weight management," *Obesity*, vol. 22, pp. 1962-1967, 2014.

[13] L.-J. Li and L. Fei-Fei, "Optimol: automatic online picture collection via incremental model learning," *International journal of computer vision*, vol. 88, pp. 147-168, 2010.

[14] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, 1998.

[15] A. Kapoor and E. Horvitz, "Experience sampling for building predictive user models: a comparative study," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008.

[16] B. Longstaff, S. Reddy and D. Estrin, "Improving activity classification for health applications on mobile devices using active and semi-supervised learning," in *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, 2010.

[17] E. Thomaz, I. Essa and G. D. Abowd, "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.

[18] R. Solis, A. Pakbin, A. Akbari, B. J. Mortazavi and R. Jafari, "A human-centered wearable sensing platform with intelligent automated data annotation capabilities," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, 2019.

[19] T. Choudhury, G. Borriello, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea, D. Wyatt, S. Consolvo, D. Haehnel and others, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Computing*, pp. 32-41, 2008.

[20] E. M. Tapia, S. S. Intille and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *International conference on pervasive computing*, 2004.

[21] M. Fuller-Tyszkiewicz, H. Skouteris, B. Richardson, J. Blore, M. Holmes and J. Mills, "Does the burden of the experience sampling method undermine data quality in state body image research?," *Body Image*, vol. 10, pp. 607-613, 2013.

[22] D. Guan, W. Yuan, Y.-K. Lee, A. Gavrilov and S. Lee, "Activity recognition based on semi-supervised learning," in *Embedded and Real-Time Computing Systems and Applications, 2007. RTCSA 2007. 13th IEEE International Conference on*, 2007.

[23] Y. Zhou and S. Goldman, "Democratic co-learning," in *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on*, 2004.

[24] A. Mannini and S. Intille, "Classifier Personalization for Activity Recognition using Wrist Accelerometers," *IEEE journal of biomedical and health informatics*, 2018.

[25] L. Chan, V. D. Swain, C. Kelley, K. Barbaro, G. D. Abowd and L. Wilcox, "Students' Experiences with Ecological Momentary Assessment Tools to Report on Emotional Well-being," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, p. 3, 2018.

[26] C. Bi, G. Xing, T. Hao, J. Huh, W. Peng and M. Ma, "Familylog: A mobile system for monitoring family mealtime activities," in *Pervasive Computing and Communications (PerCom), 2017 IEEE International Conference on*, 2017.

[27] Y. Dong, A. Hoover, J. Scisco and E. Muth, "A new method for measuring meal intake in humans via automated wrist motion tracking," *Applied psychophysiology and biofeedback*, vol. 37, no. 3, pp. 205-215, 2012.

[28] S. Eskandari, "Bite Detection and Differentiation Using Templates of Wrist Motion," Master's Thesis, Clemson University, Clemson, SC, USA, 2013.

[29] A. Bedri, R. Li, M. Haynes, R. P. Kosaraju, I. Grover, T. Prioleau, M. Y. Beh, M. Goel, T. Starner and G. Abowd, "EarBit: using wearable sensors to detect eating episodes in unconstrained environments," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, p. 37, 2017.

[30] H. Kalantarian, N. Alshurafa, T. Le and M. Sarrafzadeh, "Monitoring eating habits using a piezoelectric sensor-based necklace," *Computers in biology and medicine*, vol. 58, pp. 46-55, 2015.

[31] M. Farooq and E. Sazonov, "Real time monitoring and recognition of eating and physical activity with a wearable device connected to the eyeglass," in *Sensing Technology (ICST), 2017 Eleventh International Conference on*, 2017.

[32] J. A. Johnstone, P. A. Ford, G. Hughes, T. Watson and A. T. Garrett, "Bioharnessâ„¢ multivariable monitoring device: part. II: reliability," *Journal of sports science & medicine*, vol. 11, p. 409, 2012.

[33] A. Guzhva, S. Dolenko and I. Persiantsev, "Multifold acceleration of neural network computations using GPU," in *International Conference on Artificial Neural Networks*, 2009.

[34] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, 2010.

[35] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox and A. Schmidt, "Micro-blog: sharing and querying content through mobile phones and social participation," in *Proceedings of the 6th international conference on Mobile systems, applications, and services*, 2008.

[36] P. Klasnja, S. Consolvo, T. Choudhury, R. Beckwith and J. Hightower, "Exploring privacy concerns about personal sensing," in *International Conference on Pervasive Computing*, 2009.

[37] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE journal on Selected Areas in Communications*, vol. 33, pp. 2418-2428, 2015.

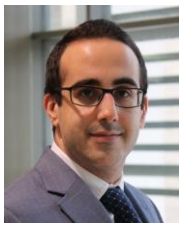
[38] Ganchev, Kuzman, B. Taskar and J. Gama, "Expectation maximization and posterior constraints," in *Advances in neural information processing systems*, 2008.

[39] Dempster, A. P. a. Laird, N. M. a. Rubin and D. B, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, pp. 1-22, 1977.

[40] Gal, Y. a. Ghahramani and Zoubin, "Dropout as a Bayesian approximation," *arXiv preprint arXiv:1506.02157*, 2015.

[41] H. Gammulle, S. Denman, S. Sridharan and C. Fookes, "Two stream lstm: A deep fusion framework for human action recognition," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[42] J. W. a. W. Lockhart, G. M. a. Xue, J. C. a. Gallagher, S. T. a. Grosner, A. B. a. Pulickal and T. T, "Design considerations for the WISDM smart phone-based sensor mining architecture," in *Proceedings of the Fifth International Workshop on Knowledge Discovery from Sensor Data*, 2011.



Ali Akbari is a Ph.D. student in Biomedical Engineering at Texas A&M University. He received his B.Sc. and M.Sc. degrees in Electrical and Bio-electrical Engineering from Sharif University of Technology, Tehran, Iran in 2013 and 2015 respectively. His research interests include artificial intelligence, machine learning, deep learning, signal processing, and embedded systems for applications in remote, smart and connected health.

Roger Solis was an M.S. student in the department of Computer Science & Engineering at Texas A&M University, earning his degree in the summer of 2019. Prior to that, he received his B.S. in Computer Science from the Universidad Autonoma de Yucatan and was awarded a scholarship from the Consejo Nacional de Ciencia y Tecnologia to study wearable sensors, signal processing algorithms, and machine learning techniques for activity recognition. He now works as a software engineer at Venafi.

Bobak J. Mortazavi received his B.S. degree in electrical engineering and computer science and his B.A. degree in applied mathematics from the University of California, Berkeley, in 2007 and his Ph.D. degree in computer science from the University of California, Los Angeles, in 2014. He is currently an assistant professor of Computer Science & Engineering at Texas A&M University, College Station. He previously conducted postdoctoral research at the Yale School of Medicine in the Yale Center for Outcomes Research and Evaluation, New Haven, Connecticut, on machine learning for cardiovascular care and clinical outcomes, and currently holds an Adjunct position at Yale University to continue this research. His research interests lie at the intersection of personal sensing, machine learning analytics, and clinical outcomes research for applications in smart and connected health.

Roozbeh Jafari (SM'12) is an associate professor in Biomedical Engineering, Computer Science and Engineering and Electrical and Computer Engineering at Texas A&M University. He received his Ph.D. in Computer Science from UCLA and completed a postdoctoral fellowship at UC-Berkeley. His research interest lies in the area of wearable computer design and signal processing. He has raised more than $55M for research with $6.9M directed towards his lab. His research has been funded by the NSF, NIH, DoD (TATRC), AFRL, AFOSR, DARPA, SRC and industry (Texas Instruments, Tektronix, Samsung & Telecom Italia). He has published over 170 papers in refereed journals and conferences. He has served as the general chair and technical program committee chair for several flagship conferences in the area of Wearable Computers. Dr. Jafari is the recipient of the NSF CAREER award (2012), IEEE Real-Time & Embedded Technology & Applications Symposium best paper award (2011), Andrew P. Sage best transactions paper award (2014), ACM Transactions on Embedded Computing Systems best paper award (2019), and the outstanding engineering contribution award from the College of Engineering at Texas A&M (2019). He has been named Texas A&M Presidential Fellow (2019). He serves on the editorial board for the IEEE Transactions on Biomedical Circuits and Systems, IEEE Sensors Journal, IEEE Internet of Things Journal, IEEE Journal of Biomedical and Health Informatics, IEEE Open Journal of Engineering in Medicine and Biology and ACM Transactions on Computing for Healthcare. He serves on scientific panels for funding agencies frequently and is presently serving as a standing member of the NIH Biomedical Computing and Health Informatics study section.