San Jose State University

From the SelectedWorks of Magdalini Eirinaki

2016

PRO-Fit: A personalized fitness assistant framework

Saumil Dharia, *San Jose State University*
Vijesh Jain, *San Jose State University*
Jvalant Patel, *San Jose State University*
Jainikkumar Vora, *San Jose State University*
Shaurya Chawla, *San Jose State University*, et al.

PRO-Fit: A personalized fitness assistant framework

Saumil Dharia, Vijesh Jain, Jvalant Patel, Jainikkumar Vora, Shaurya Chawla, Magdalini Eirinaki
Computer Engineering Department
San Jose State University
San Jose, CA, USA

ABSTRACT—The advancements in wearable technology, where embedded accelerometers, gyroscopes and other sensors enable the users to actively monitor their activity have made it easier for individuals to pursue a healthy lifestyle. However, most of the existing applications expect continuous feedback from the end users and fail to engage those who have busy schedules, or are not as committed and self-motivated. In this work, we propose a framework that employs machine learning and recommendation algorithms in order to smartly track and identify user's activity by collecting accelerometer data, synchronizes with the user's calendar, and recommends personalized workout sessions based on the user's and similar users' past activities, their preferences, as well as their physical state and availability.

KEYWORDS—wearable technology, activity tracking, classification, recommendation, personalized assistant

I. INTRODUCTION

The advancements in wearable technology, where embedded accelerometers, gyroscopes, GPS tracking and other sensors enable the users to actively monitor their activity have revolutionized the field, by allowing users to engage and track fitness activities. According to a study screening 200 existing health and fitness apps, the main priority for users is to have an application that makes any physical activity entertaining and rewarding, and motivates them to continue striving for achieving targets they set [1]. In this paper, we present PRO-Fit, a personalized fitness assistant framework. We are motivated by the fact that for busy individuals, the existing interactive models might not be enough to keep them motivated to engage in fitness-related activities. Apart from being a simple activity tracker, PRO-Fit automatically identifies when an individual can exercise, and proactively notifies the user. Instead of simple reminders, the framework automatically generates personalized fitness schedules, depending on the user's location, availability, and preferences.

The proposed framework incorporates two critical modules: an activity classifier, and a ranking and recommendation engine. We use machine learning algorithms on activity data to build predictive models that classify the user's activity into specific types. We build user profiles reflecting their current lifestyle (e.g. sedentary vs. active), age, weight, goals (e.g. time spent exercising each week), and preferences (e.g. favorite fitness activities, level of intensity etc.). This user profile is fed to a hybrid recommendation system that matches the user's profile to available activities, ranked in terms of similarity, but also taking into consideration the geo-location and time availability. For instance, PRO-Fit might recommend a 1-hour yoga class at the University fitness center during lunch time for user A, who's employee at the University, and 20-min jogging at the nearest park for user B, who is a student and has 30 minutes between classes. The rest of the paper is organized as follows: In Section II we review the related work. In Section III we present the results of a user survey that helped us define the requirements for our framework. The architecture and implementation details are discussed in Section IV. In Section V we discuss some preliminary experimental results, and we conclude with our plans for future work in Section VI.

II. RELATED WORK

The existing activity trackers have inbuilt sensors that are able to recognize activity. A lot of research work has been done in this area, by employing machine learning algorithms on past user activity data [2], heart rate data [3], and accelerometer data [4,5, 6] to identify the type of activity, or estimate caloric consumption. In [7], the authors use an on-body chest sensor in coordination with a smartphone to collect the data for the activities performed by the individual, whether static or dynamic. In [8] the authors propose the Digital Fitness Connector (DFC) architecture, which allows the user to monitor physical activity in real-time as well as post-workout. The majority of the proposed approaches employ decision trees and their scalable variations (such as random forests) to perform the activity classification [9], however clustering approaches have also been used to split activity data into categories [10]. While most of existing works focus on improving the activity prediction process, most require extensive user profiling and interaction throughout the day. In our work, we propose a framework that minimizes the need for user input by *proactively* reminding him/her about their goals and generating personalized fitness recommendations based on their daily schedule and activities.

III. REQUIREMENTS ANALYSIS

In order to collect information about the users' preferences, devices/apps used, activities performed, feature they like the most, features to modify/add to the existing apps, activity schedule, we conducted a survey among more than 200 people[1]. Based on the survey information we identified, among others, the following requirements: a) the motivation for most of the users is goal tracking, b) the favorite feature is to calculate the calories burnt and present the information on a daily basis, c) people try to schedule their fitness activities at least 2-4 times a week, for 1-2 hours each day, d) most people preferred working out alone, e) the activities that were mostly tracked were walking, jogging and cycling, f) people are

---

[1] The survey was conducted online, through different social networks. We received around 200 responses in 3 days. We kept the survey anonymous, except for the age (range between 18 and 45 years). The survey questions can be found at: https://docs.google.com/forms/d/1MDpDs2yiSL8gYqddpAJX0jp4iftPg3tKe9bx3eZlgjo/viewform

willing to plan their fitness schedule during work hours, g) a desirable feature would be a recommendation system for fitness activities, h) a desirable feature would be reminders and fitness session organizer. All the applications mentioned in Section II.B are more focused to track the user's activities, but none of them provides a recommendation system that would help users to choose from activities based on their interest and accomplishment of goals. Based on these observations, we were motivated to design the PRO-Fit personalized fitness assistant framework that acts as a motivator and organizer for fitness activities making it easy for users to create and follow their workout plan and schedule the sessions according to their availability and preference.
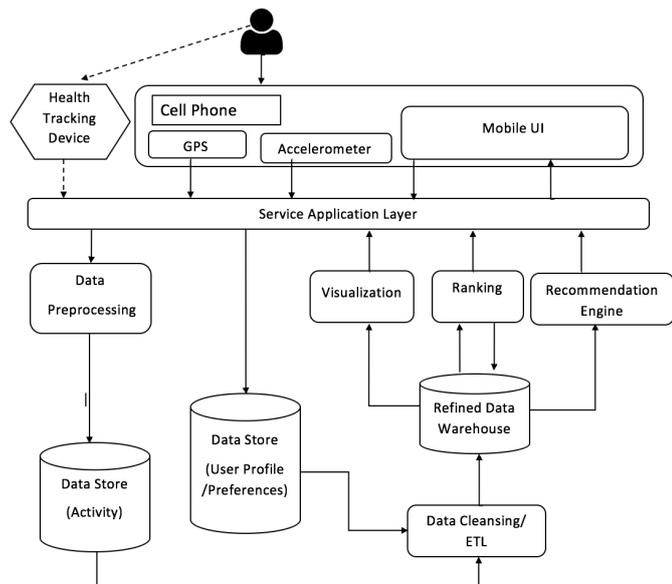

Figure 1. System Architecture Diagram

IV. ARCHITECTURE

The proposed framework architecture is shown in Figure 1. The framework is designed such that it scales for multiple clients. In what follows, we discuss the functionality and the most important design decisions for each module.

A. Health Tracking Device Client

The framework's client is an Android/iOS mobile application, which contains UI screens and interacts with Mobile GPS and accelerometer to get user's activity and location data. The end user interacts with the mobile app to sign up, occasionally give inputs and receive recommendations on fitness sessions. Third party wearable devices expose developer APIs to pull data for the analysis. Such devices can sync with the Application and service layer can pull data from respected data stores of wearable service providers.

B. Application Service Layer

This is the interface for all interactions of the mobile application to the outer world of the system. This layer can be divided into two different parts: a) the *RESTful APIs* to provide business logic as per the system's functional requirements, and b) the *Service Interface* to connect to notification server, which communicates with the specific client/user to recommend some fitness activities/fitness partner at runtime.

C. Calendar management

Integrating multiple calendars to the system is very crucial and requires scalable and flexible design. This is achieved by employing the adapter design pattern. As different calendar service providers are having different data format and APIs, adapters can be created for each service provider. Such design is highly scalable as support for new service provider can be added easily without modifying the whole modular code base.

D. Data Pre-processing and classification

This module focuses on the preprocessing and classification of users' activities and provides information about the ETL process carried out to store the classified and processed data. This process consists of three main parts: data collection, data pre-processing and storage, and data classification.

Data Collection. The accelerometer data is collected from user's mobile device. It provides x, y and z co-ordinates with respect to the surface on which the user is performing the fitness activities along with timestamp.

Data pre-processing and storage. The activity data needs to be first preprocessed to identify features within a specific time window that will be used as input to the classification process. The time windows are parameters of the system and can be different for each user and each activity. Figure 2 details the process of defining these windows for a particular interval.
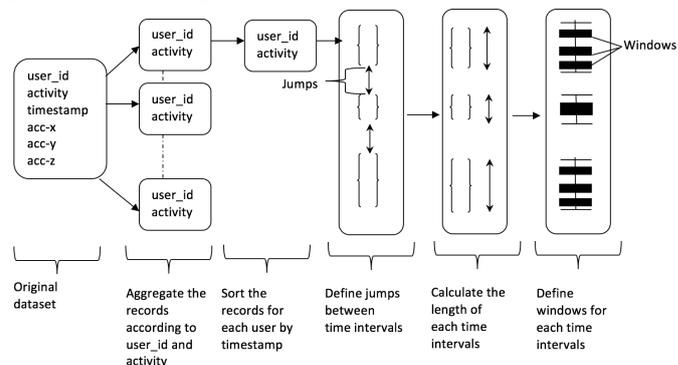

Figure 2. Data processing architecture

In the first step, the records in the original data store are grouped by user id and activity. After grouping is done, the records are sorted by timestamp in ascending order. The next step is to identify the *jumps* that are durations when no data is collected. The jump time interval is a parameter of the framework that needs to be determined experimentally. In our framework, this jump time interval is set to 5 minutes. Finally, we determine the length of each interval (*time window)* for which we want to identify the user activity. We experimented with intervals of 15, 10 and 5 seconds.

The amount data that needs to be collected is very big as it is generated every 50 milliseconds. We use Apache Kafka to stream the data in real time, while Cassandra is the best selection for storing time-series data as it has the capacity to handle real time requests. The data stored in Cassandra is ingested by feature calculation mechanism and then used by the application's classifier algorithm to classify the user's activity in near real time. The decoupling between data generation and classifier via Kafka would provide high scalability with increasing users.

Data classification. This is the core component of this module. In the current implementation, the user activities are classified in one of the following categories: *walking, jogging, running,*

climbing stairs, descending stairs and *cycling*. The classifier needs to be designed to classify each category with high accuracy, since some activities present very similar characteristics (for example, running and jogging). In order to identify user activity for a particular time interval we follow an approach inspired by the work of [11, 12], and identify the following features:

- *Average acceleration (*calculated for each axis).
- *Standard deviation* (calculated for each axis).
- *Average absolute difference* (the average of the difference between the value of each input sample records and mean of the total input sample records, calculated for each axis).
- *Average resultant acceleration* (the average of the square root of sum the squares of values of each axis).
- *Time between peaks* (the time in milliseconds between the peaks in the sine wave for each axis).

The features are calculated for a particular window size and then the user's activity is classified based on the features calculated for each window. The feature "time between peaks" is useful to find repetitive patterns such as walking and jogging. We have experimentally evaluated various algorithms and present the most interesting findings in Section V.

E. Data Stores

The framework includes multiple types of data stores on the basis of the type of data. *User activity data store* stores user's physical activity data. This data can be used in machine learning to train the model for better prediction and recommendations. *User profile data store* stores data related to the user profile, their connections, preferences and settings. *Data Warehouse* will take data from a data cleansing/ETL module will convert raw data to a standardized schema that will be useful for data analytics.

F. Visualization

This module is responsible for generating graphs or trends that would provide valuable insights to users. The main components of this module are described here. Due to huge amount of user activity data, we use Kafka to stream activity data from the application database for ETL processing Spark job since it promises high throughput [13]. To provide near real time analytics, we decided to use Apache Spark that provides a streaming library that can take input from Kafka, process it and output to any data source. One of the important tasks in the ETL is mapping of activity with goal and session based on its start time and end time that is performed in Spark. The output of Spark job will be store in a warehouse. Data Visualization module fetches data from the data warehouse and displays them in various graphic visualizations.

G. Fitness Session Recommendation Engine

This component uses user profile and activity data to recommend fitness sessions to a particular user on the basis of different parameters. This module takes following attributes as an input:

- *Type of Activity*: Physical activity like jogging, running etc.
- *Session*: Time period when the user performs a specific fitness activity.
- *Calendar Event*: A time period on a specific day.
- *Calendar*: Calendar for scheduled fitness activities.

- *Availability*: Time periods when a specific user has no calendar event.
- *Blocked Time Slots*: When user doesn't want fitness session.
- *Goal*: Time period, during which a user targets to perform certain activities/sessions to achieve a specific goal given in terms of "calories burnt" or "distance walked" etc.
- *User Strength*: A user's capacity to perform specific activity for a specific time period. Calculated using BMI (Body Mass Index).

For session recommendation, system performs two activities: 1. Generate a session for a user based on his strength, goal and activity preference. 2. Generate list of calendar events for the sessions generated in step 1 based on user's availability. Below is the outline of how above two functions would work

Session generation. Time duration of daily fitness Session can be calculated from given *Goal* (calories to burn and time duration to accomplish the target), user's fitness *Activity preference* and user's BMI (Body Mass Index).

Recommendation process. Considering user's *Availability* and *Blocked Time Slots* by syncing all user's calendar events, the framework generates a personalized list of recommended fitness Sessions as a Calendar Event in the fitness Calendar**,** as follows:

1. Upon sign up, the user enters all the personal information (e.g. Calories, height, weight, activity preference etc.)
2. After registration, PRO-Fit retrieves all the contacts and calendars event from the user's device into the application.
3. PRO-Fit inputs the user profile in a hybrid recommender system and generates a list of recommended fitness sessions. Recommendations are based on user's preferences (using content-based recommendations), as well as these of similar users (using item-based collaborative filtering).
4. User can reschedule session anytime.PRO-Fit will smartly scan for all the other available slots in the calendar and schedule that session.
5. A periodic sync of calendars, as well as updates on the user profile (including new activity recommendations) will be supported by the system.

When a session is just about to start, the app will send a push notification on the user's device before a certain amount of time to give a gentle reminder.

H. Social Ranking Recommendation Engine

This module integrates with the user's social network, and ranks and recommends users or friends as "fitness buddies" to the user. Fitness challenges, goal tracking and regularity in fitness activities are decision factors to decide relative ranking. This module will be developed as part of future work.

V. EXPERIMENTAL EVALUATION

Identifying user activities based on accelerometer data can be defined as a classification problem. We experimentally evaluated various classification algorithms to identify the most appropriate ones for this task. Since there is no single platform that provides all libraries, we the *Decision Tree, Random Forest*, and *Multinomial Logistic Regression* implementations of Spark's MLlib, the *Extremely Randomized Trees, AdaBoost, K-NN, Naïve Bayes* and *SVM (multi-class)* implementations of Python's Scikit-learn, and also implemented the *Gradient*

Boosted Trees algorithm, mainly to analyze the effect of this ensemble method on creating accurate model.

Raw input labeled data Statistics	
Number of inputs records	2,980,764
Class Distribution	
Walking	1,255,923 (42.1%)
Jogging	438,871 (14.7%)
Stairs	57,425 (1.9%)
Sitting	663,706 (22.3%)
Standing	288,873 (9.7%)
Lying Down	275,967 (9.3%)

Table 1 Raw input labeled data statistics

For our experiments, we used the dataset collected by Kwapisz et al. [11], publicly available through the WISDM lab [14], including activity data collected from android devices. The data was collected at an interval of 50ms, which means it contains 20 samples per second. It contains user information along with their tri-axial axis information and timestamp when it was collected. Each axis represents the relative movement of the user with respect to the surface where the user activity was performed The statistics of the data set are included in Table 1.

We first calculated the feature set as described in section IV. Once the feature set is calculated, these features are used as input to the classifier algorithms. The experiments were carried out for sampling rate of 300, 200 and 100 records of test data to measure the accuracy for each algorithm. We used two validation methodologies, namely holdout set validation (with a 60/40 split for training/test data respectively), and 10-fold cross validation. Figures 3 and 4 show the comparison of different algorithms for different window sizes (we omit the rest due to space constraints). We observe that as we decrease the window size the number of samples increases and we get and better accuracy for our classifiers. In terms of accuracy, multiple algorithms perform well, with the decision tree-based ones consistently giving better results for all window sizes.

Figure 3. Comparison of all algorithms - window size of 10 seconds

VI. CONCLUSIONS

In this paper we presented PRO-Fit, a novel framework for personalized fitness recommendations. Apart from presenting details on the system architecture and module design, we experimentally evaluated several machine learning algorithms concluding that decision tree-based algorithms are the best choice for classifying activity data. In the next phase of this project we plan to focus on incorporating the social network of the user, by designing recommendation algorithms to suggest "fitness buddies" based on similar users and proximity.
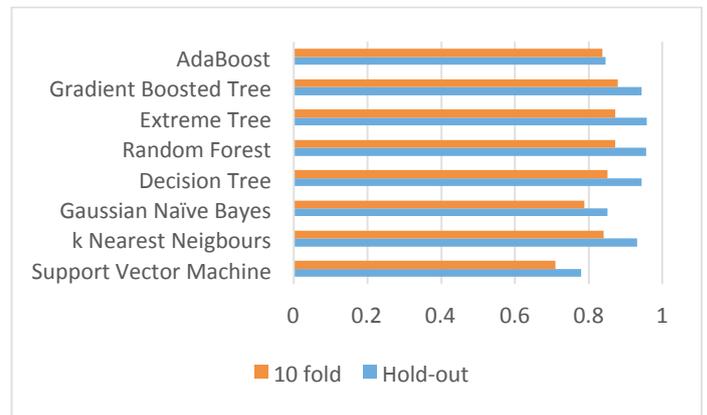

Figure 4. Comparison of all algorithms - window size of 5 seconds

REFERENCES

[1] "MobiHealthNews: Half of mobile health app users are using fitness apps" [Online]. Available at http://mobihealthnews.com/30199/half-of-mobile-health-app-users-are-using-fitness-apps/

[2] Jin-Hyuk Hong; Ramos, J.; Dey, A.K., "Toward Personalized Activity Recognition Systems With a Semipopulation Approach," in IEEE Trans. on Human-Machine Systems, 46(1), pp.101-112, Feb. 2016

[3] Bajpai, A.; Jilla, V.; Tiwari, V.N.; Venkatesan, S.M.; Narayanan, R., "Quantifiable fitness tracking using wearable devices," in 37th IEEE Intl. Conf. of the Engineering in Medicine and Biology Society (EMBC), Aug. 2015

[4] Ayu, M.A.; Mantoro, T.; Matin, A.F.A.; Basamh, S.S.O., "Recognizing user activity based on accelerometer data from a mobile phone," in 2011 IEEE Symposium on Computers & Informatics (ISCI), March 2011

[5] Anjum, A.; Ilyas, M.U., "Activity recognition using smartphone sensors," in 2013 IEEE Consumer Communications and Networking Conference (CCNC), Jan. 2013

[6] Ghose, S.; Barua, J.J., "A systematic approach with data mining for analyzing physical activity for an activity recognition system," in Intl. Conf. on Advances in Electrical Engineering (ICAEE), Dec. 2013

[7] Guiry, J.J.; Van de Ven, P.; Nelson, J., "Classification techniques for smartphone based activity detection," in 11th IEEE Intl. Conf. on Cybernetic Intelligent Systems (CIS), Aug. 2012

[8] Gupta, N.; Jilla, S., "Digital Fitness Connector: Smart Wearable System," in 1st Intl. Conf. on Informatics and Computational Intelligence (ICI), Dec. 2011

[9] Uddin, M.T.; Uddiny, M.A., "Human activity recognition from wearable sensors using extremely randomized trees," in 2015 Intl. Conf. on Electrical Engineering and Information Communication Technology (ICEEICT), May 2015

[10] Yingying Li; Yimin Zhang, "Application of data mining techniques in sports training," in 5th Intl. Conf. on Biomedical Engineering and Informatics (BMEI), Oct. 2012

[11] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," ACM SIGKDD Explorations Newsletter, pp. 74–74, 2011.

[12] M.-K. Suh, A. Nahapetian, J. Woodbridge, M. Rofouei, and M. Sarrafzadeh, "Machine Learning-Based Adaptive Wireless Interval Training Guidance System," Mobile Networks and Applications Mobile Netw Appl, pp. 163–177, 2011

[13] J. Kreps, "Benchmarking Apache Kafka: 2 Million Writes Per Second (On Three Cheap Machines)," [Online]. Available at: https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines. [Accessed: Jul-2015].

[14] Lockhart, J., Weiss, G., Xue, J., Gallagher, S., Grosner, A., & Pulickal, T. (n.d.). "Design considerations for the WISDM smart phone-based sensor mining architecture." In 5th Intl. Workshop on Knowledge Discovery from Sensor Data - SensorKDD '11