Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation

Jianmo Ni
University of California, San Diego
jin018@ucsd.edu

Larry Muhlstein
University of California, San Diego
larrymuhlstein@gmail.com

Julian McAuley
University of California, San Diego
jmcauley@ucsd.edu

ABSTRACT

Activity logs collected from wearable devices (e.g. *Apple Watch*, *Fitbit*, etc.) are a promising source of data to facilitate a wide range of applications such as personalized exercise scheduling, workout recommendation, and heart rate anomaly detection. However, such data are heterogeneous, noisy, diverse in scale and resolution, and have complex interdependencies, making them challenging to model. In this paper, we develop context-aware sequential models to capture the personalized and temporal patterns of fitness data. Specifically, we propose *FitRec* – an LSTM-based model that captures two levels of context information: context within a specific activity, and context across a user's activity history. We are specifically interested in (a) estimating a user's heart rate profile for a candidate activity; and (b) predicting and recommending suitable activities on this basis. We evaluate our model on a novel dataset containing over 250 thousand workout records coupled with hundreds of millions of parallel sensor measurements (e.g. heart rate, GPS) and metadata. We demonstrate that the model is able to learn contextual, personalized, and activity-specific dynamics of users' heart rate profiles during exercise. We evaluate the proposed model against baselines on several personalized recommendation tasks, showing the promise of using wearable data for activity modeling and recommendation.

KEYWORDS

Fitness Recommendation; Sequential Modeling; Personalization

ACM Reference Format:
Jianmo Ni, Larry Muhlstein, and Julian McAuley. 2019. Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313643

1 INTRODUCTION

The development of wearable technologies has given people the opportunity to measure and track their activities via mobile devices (e.g. smart watches). Such devices collect various types of data, including health-related measurements (e.g. heart rate) and contextual measurements (e.g. location, altitude, activity type). An example of such data (from *endomondo.com*) is shown in Figure 1. Harnessing this data to model fitness is beneficial for developing safer and more reliable activity-related services [1]. However, this remains a challenging task for a number of reasons: (1) the collected sequential data are often heterogeneous and the interdependencies among different variables are hard to capture; (2) users' activity patterns and health conditions change over time; and (3) the data has high variance across users, and while large datasets can be easily collected, the amount of data associated with each user is limited.

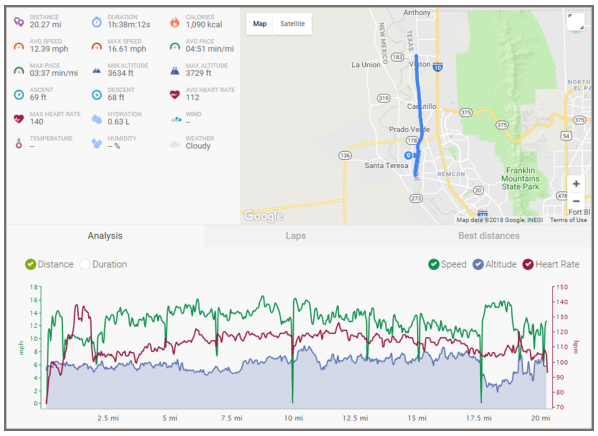


Figure 1: A representative example of our data collected from *endomondo.com*. The figure contains measurements during one workout, including route, speed, altitude and heart rate sequences as well as overall statistics.

Researchers have recently sought to incorporate deep learning techniques to capture the dynamics of sequential data. One of our goals is to adapt such models to capture personalized dynamics in GPS and heart rate data. Recurrent Neural Network (RNN) based models [6, 13] have been proposed to model multivariate sequential data and shown promising performance [5, 20]. Compared with traditional models, RNNs are able to learn complex non-linear relationships between input variables. Furthermore, RNNs are convenient for modeling sequential problems with different input and output structures (e.g. sequential inputs and sequential outputs, or sequential inputs and numerical outputs). This allows us to straightforwardly design model variants for different prediction tasks on our data. More recently, attention mechanisms [3] have further boosted the performance of RNN based models due to their ability to make the model aware of all previous hidden states.

In order to bring state-of-the-art models of sequential data to bear on large datasets of fitness activities, it is important to ask how we can *personalize* models of sequence data to individual users. Personalization is crucial when modeling people's fitness due to the wide variation among individuals. For example, users may differ

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313643

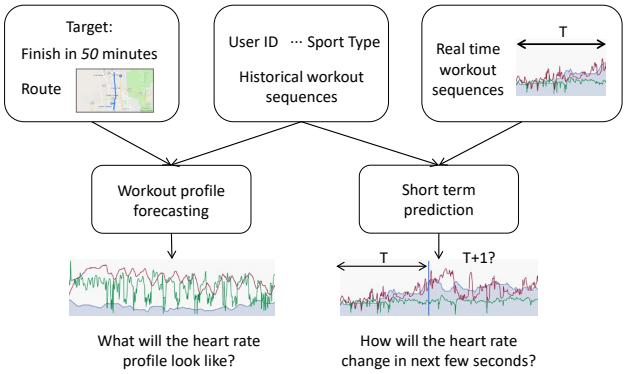


Figure 2: *FitRec* can be applied to two different tasks: workout profile forecasting and short term prediction.

significantly in comfort heart rate zones, adaptation ability for different exercises, etc. Accounting for such differences requires robust modeling techniques to capture the dynamics of both explicit and implicit user-dependent features. An effective model needs to efficiently learn both static and temporal features from historical data.

In this paper, we address these challenges by exploring a large-scale dataset of workout records and building a predictive model leveraging context within and across workouts. Specifically, we propose a Long Short-Term Memory (LSTM) based model *FitRec* that takes as input user attributes and multiple workout measurement sequences. *FitRec* infers static user embeddings based on the attributes and temporal user embeddings from historical workout measurements. By combining these inferred latent representations of our users, *FitRec* can then be applied to (1) Quantitative tasks, such as personalized sequential modeling to predict how workout measurements (e.g. heart rate) will change across a workout, either beforehand (i.e., based on a map of the intended route) or in real time as the user exercises; (2) Qualitative tasks, such as identifying important features that affect workout performance, or identifying clusters of users based on common embedding structures; and (3) Recommendation tasks, such as recommending alternate routes that will achieve a target heart rate profile.

In particular, we evaluate *FitRec* on two quantitative tasks: 'workout profile forecasting' and 'short term prediction' as shown in Figure 2. In the former, we are concerned with predictions such as *estimating a user's likely speed and heart rate profile given the activity they intend to perform* (e.g. cycling a particular GPS route); in the latter we are interested in short-term prediction, e.g. determining during an ongoing workout *how the user's heart rate will change in the next few seconds or minutes*. To apply to both tasks, *FitRec* uses shared embedding layers while adapting a common predictive component: a two-layer stacked LSTM module for workout profile forecasting and an attention-based encoder-decoder module for short term prediction. We also demonstrate the model's capability to provide personalized recommendations that could help users toward their fitness objectives. Workout profile forecasting can be used (for example) to recommend routes that match a user's desired exercise style, while short term prediction might be used to dynamically help a user match their target heart rate. Fitness and activity data have attracted interest in the data mining and recommendation fields because they contain rich contextual information (e.g. spatial and temporal information) about users that might complement existing recommender systems [2]. However, there have been few works studying how these data can be used in real-world recommendation applications.

The key contributions in this paper are threefold:

(1) We propose an LSTM-based model *FitRec* that considers context both within and across workouts. *FitRec* infers static user embeddings from user attributes and meanwhile learns temporal embeddings from users' recent workout sequences. *FitRec* can be applied to various tasks by using either a two-layer stacked LSTM module or an attention-based encoder-decoder module. We show the model's performance compared with prior sequential modeling baselines such as Multilayer Perceptrons (MLP) [15] and Dual-stage Attention-based RNNs (DA-RNN) [25].
(2) We demonstrate that *FitRec* can be used on two real-world recommendation tasks: workout route recommendation and short term heart rate prediction. We evaluate performance in terms of standard recommendation metrics (e.g. AUC, F1-score) and show that our model outperforms strong baselines.
(3) We contribute a workout dataset that contains over 250 thousand workout records with multiple types of heterogeneous sequential signals as well as metadata. To the best of our knowledge, we are the first to release a large-scale dataset of this type. Beyond the recommendation problems considered here, such a dataset has significant potential for re-use in research on heart rate data, sequence modeling, personalization, etc.

While there are several works focused on analyzing human activities, we are (to the best of our knowledge) the first to investigate personalized recommendation using sequential modeling methods on fitness-oriented objectives and to conduct extensive experiments on a real-world dataset.

2 RELATED WORK

Our work focuses on leveraging recent advances in RNN-based sequential modeling for fitness prediction and recommendation. We discuss the related work from each relevant area as follows.

Mining Sensor Data. One line of work falls into the category of *pervasive computing*, which studies how to collect and handle data from mobile and wearable devices. The data may include various types of human behavior such as sleep, exercise, surgery, health, etc. [16, 19, 23]. [1] discusses the challenges and recent advances in population-scale pervasive health research. Their work emphasizes the importance of sensor data for uncovering the causal relationships between human activity and health outcomes. [4] provides an overview of data mining research in the healthcare domain and concludes that the capability to build context-aware and personalized decision systems is important for the pervasive sensing market.

Recently, there is a growing trend toward modeling human wellness using sensor data. Farseev et al. collected users' exercise data and proposed a model to combine social network information to predict user wellness trends [9]. Particularly, they focused on BMI

(Body Mass Index) categorization during several periods for each specific user. [7] focused on detecting physical exercise types given limited training data. They extracted features from exercise data (e.g. heart rate, distance) and proposed an AdaBoost-based method to predict the exercise type (e.g walking, aerobics, running, etc.). Both of these works focus on handcrafting features from sensor data, which are then applied to classification problems. Although similar in data modality, our work differs in terms of the prediction tasks we consider, i.e., sequential modeling and sequence prediction.

Context-aware Modeling. Context-aware models have been successfully applied in many fields with abundant contextual information such as recommender systems, social networks, clinical predictions, etc. [8, 24]. Like these applications, fitness and exercise data naturally has heterogeneous input structure in the form of sequential measurements and contextual information. In [18] a recurrent model was proposed with a context-aware layer for predicting a patient's Blood Pressure (BP). The model considers contextual input (e.g. age, gender, BMI) which is encoded via one-hot vectors or numerical values. Experiments showed that adding this contextual information to the model improves BP prediction accuracy. Unlike their approach, we use attribute embedding and contextual embedding modules consisting of LSTM layers to learn from entire historical sequences. Furthermore, their blood pressure measurement data does not consist of measurements within each activity, which differs from the data we consider.

RNN based sequential modeling. In recent years, Recurrent Neural Networks (RNNs) have shown remarkable effectiveness in modeling sequential data, such as text, audio, and clinical data [20], among others. [17] introduced an LSTM-based model to address traditional multivariate time series prediction problems. They combined Convolutional Neural Networks (CNN) with a recurrent skip layer to capture both long- and short-term dependencies in time series data. The model achieves significant improvements over LinearSVMs and 'vanilla' RNNs. [25] proposed a dual-stage attention model to solve the problem of exogenous time series prediction. They designed an input level attention mechanism along with temporal attention to capture the relations between input variables. Similarly, we incorporate attention mechanisms in sequential data to explore the relations between contextual variables in exercise data.

Personalized Recommendation. Several works on personalized recommendation have tried to incorporate content (e.g. item attributes) and context (e.g. user clicks, purchases) to augment the model in straightforward and typically 'static' scenarios (e.g. e-commerce). Recently, researchers from both industry and academia have sought to develop systems for personalized fitness recommendation. [21] focused on personalized running route recommendation, by considering a variety of targets such as user preferences, goals, and environment. [9] combined workout and social network information from the same user in order to conduct wellness prediction. However, neither of these works focus on modeling sequential fitness data (e.g. heart rate sequences). Though [9] released a public dataset, unlike ours it does not provide complete sequential traces for each workout, but rather includes hand-crafted statistics extracted from each sequence.

While prior work has considered fitness and activity modeling from a variety of perspectives, we believe there is a research gap in terms of studying low-level sequential information (e.g. heart rate, pace) as a means of learning more fine-grained models of users and activities. This motivates us to model and publish a large-scale workout dataset with heterogeneous sequential and attribute data, and to conduct exploratory experiments on fitness recommendation tasks. We expect that our work will facilitate future study of both fitness and personalized heart rate profiling.

Table 1: Notation

Notation	Description
$\mathcal{T}_{train}$, $\mathcal{T}_{test}$	training set and test set for two prediction tasks
$\mathbf{X}, \mathbf{Z} \in \mathbb{R}^{N \times T}$	contextual sequences of the current and the most recent workout
$\mathbf{y}, \mathbf{y}' \in \mathbb{R}^{T}$	target sequence of the current and the most recent workout
$\mathbf{a}$	attributes associated with the workout
T	length of input and target sequences
L	the number of sampled data points L
N	number of contextual sequences and each sequence is associated with a variable.
m	number of attributes
$\|a_i\|$	number of distinct values of the ith attribute
D_1	dimension of attribute embeddings
D_2	dimension of contextual embeddings
$\mathrm{E}_{a_i} \in \mathbb{R}^{\|a_i\| \times D_1}$	embedding matrix of the ith attribute
$\mathbf{e}_{a_i} \in \mathbb{R}^{D_1}$	attribute embedding learned from the ith attribute
$\mathbf{e}_t \in \mathbb{R}^{D_2}$	contextual embedding learned from historical workout sequences

3 APPROACH

3.1 Notation and Tasks

Assume we are given N sequences $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_T) \in \mathbb{R}^{N \times T}$ and one target sequence $\mathbf{y} = (y_1, \ldots, y_T) \in \mathbb{R}^T$, where $\mathbf{x}_t = (x_t^1, x_t^2, \ldots, x_t^N) \in \mathbb{R}^N$, $y_t \in \mathbb{R}$, and $t \in [1, T]$. Here $\mathbf{X}$ can be considered as a combination of various contextual sequences (e.g. distance, altitude) for the current workout and $\mathbf{y}$ is a target measurement sequence (e.g. heart rate, speed). Further we assume there are N historical contextual sequences $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_T) \in \mathbb{R}^{N \times T}$ and one historical target sequence $\mathbf{y}' = (y'_1, \ldots, y'_T) \in \mathbb{R}^T$, where $\mathbf{Z}$ and $\mathbf{y}'$ are from the most recent workout. For the dataset used in this work, the number of sampled data points L for each sequence of the workouts is equal. Therefore, we also assume the same length T for all sequences in our model. Note that this is not a requirement of our model, but merely a characteristic of the data we collect, in which all sequences are quantized to have the same number of samples regardless of the original workout duration. The model can be adapted to various sequence lengths by a simple padding strategy. Each workout is also associated with multiple attributes $\mathbf{a} = (a_1, \ldots, a_m)$ (e.g. userID, sport type, gender). Our notation is summarized in Table 1.

We consider two types of sequential prediction problems: workout profile forecasting and short term prediction.

3.1.1 Workout profile forecasting. Given the contextual sequences $\mathbf{X}$, attributes $\mathbf{a}$ for a candidate workout, historical sequences $\mathbf{Z}$ and $\mathbf{y}'$, as well as the total time of the workout, predict the entire target

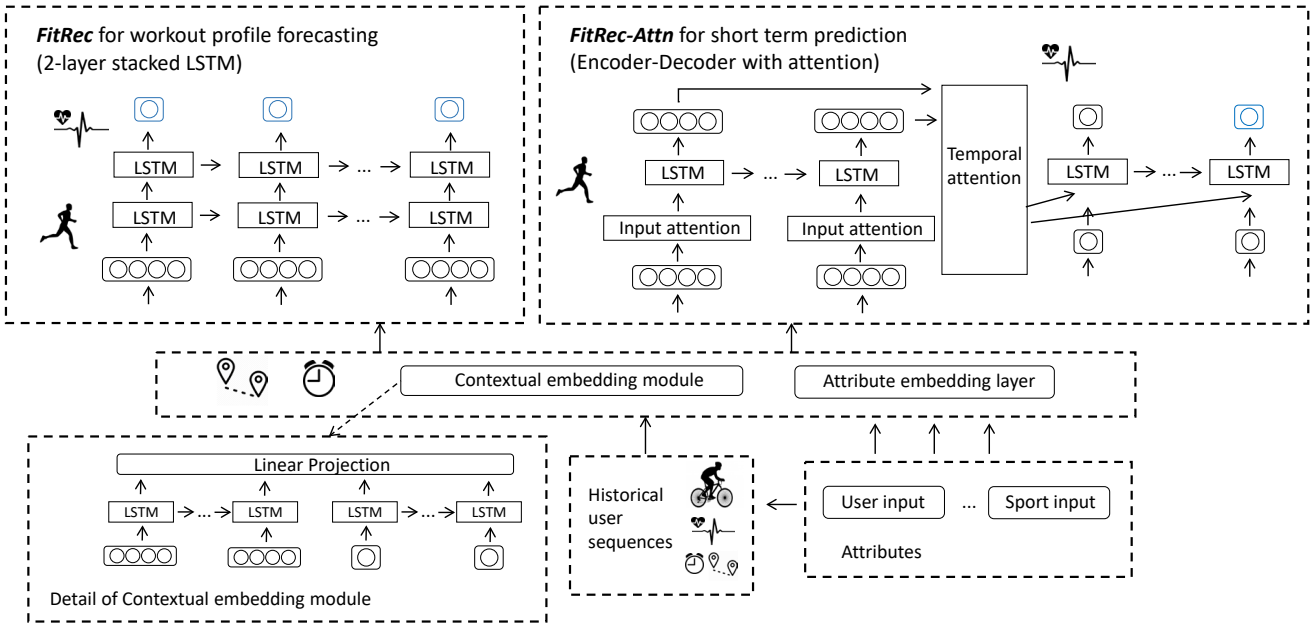


Figure 3: Model structure for workout profile forecasting (*FitRec*) and short term prediction (*FitRec-Attn*). *FitRec* contains a 2-layer stacked LSTM and *FitRec-Attn* has an encoder-decoder module with dual-stage attention. The final outputs are colored in blue.

sequence $\mathbf{y} = (y_1, \ldots, y_T)$. Notably, the input and target sequences correspond to the whole workout and T equals the total length L of the workout. This prediction task might correspond to a scenario where a user specifies a route as well as their desired total time (i.e. how much time they plan to spend on the workout), and the system estimates their likely heart rate profile.[1]

3.1.2 Short term prediction. Given the contextual sequences $\mathbf{X}$, attributes $\mathbf{a}$ for a candidate workout, historical sequences $\mathbf{Z}$ and $\mathbf{y}'$, as well as the previous target sequence $(y_1, \ldots, y_{T-1})$, predict the target output y_T at time step T. Here the input and target sequences correspond to a fixed window of one workout (i.e. part of the whole workout) and T equals the window size (i.e. T is smaller than L). This task is essentially a form of auto-regressive modeling that allows us to predict short term heart rate dynamics during an activity.

3.2 Relation Between the Two Tasks

Our two tasks differ in terms of their given inputs, prediction outputs, and application scenarios as shown in Figure 2.

Workout profile forecasting aims to predict a user's workout performance (including speed and heart rate) given an expected workout time and route. The goal is to predict the overall trends and characteristics of each measurement (e.g. max/min/avg heart rate) before the user starts the workout. As a result, the model can help the user find workout plans that meet their expectations on performance, or could be used to find alternate workout routes that are similar to their previous routes in terms of how the user will respond to them. This model can be adapted to applications such as training schedule recommendation, alternative route recommendation, etc.

On the other hand, short term prediction focuses on forecasting sequential measurements during an ongoing workout. This is similar to traditional problems of time series prediction with exogenous variables [11, 25] but differs in that fitness data has measurements both within and across activities which need to be modelled separately. Short term prediction is useful in scenarios like anomaly detection and real-time decision making (e.g. advising a user that they should slow down in the next minute to avoid exceeding their desired maximum heart rate).

3.3 Model Structure

As shown in Figure 3, we propose two models: *FitRec* for workout profile forecasting and *FitRec-Attn* for short term prediction.

Both models share the same basic structure consisting of a contextual embedding module and an attribute embedding layer. These layers encode user attributes and historical information into an embedded representation that captures their latent individual attributes (e.g. their cardiovascular fitness, endurance, etc.) and can facilitate personalized prediction. Following the embedding layers, the two models have different predictive components. As shown on the top left of Figure 3, *FitRec* consists of a 2-layer stacked LSTM which simultaneously outputs the prediction of the target variable at all time steps. *FitRec-Attn* includes an encoder-decoder network with an attention mechanism that outputs the prediction at each time step one by one as shown on the top right of Figure 3. We describe the details for each component as follows.

[1] We also experimented with a variant where the total time was *not* provided as input, but found there to be excessive variance between (e.g.) running/sprinting/jogging styles such that heart rates could not be reliably estimated without first knowing a rough intended speed.

3.3.1 LSTM to process sequences. In both models, we use Long Short-Term Memory (LSTM) [13] to process the input and target sequences. Given the input sequence $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T)$, an LSTM learns an update function

$$\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{x}_t), \tag{1}$$

where $\mathbf{h}_t$, $\mathbf{h}_{t-1}$ are the hidden states of the LSTM at time step t and $t-1$, respectively, and $\mathbf{x}_t$ is the input at time step t. Different from traditional RNNs, the LSTM cell introduces several gates and memory cells. We follow the LSTM structure from [25, 28]:

$$\begin{aligned} \begin{pmatrix} \mathbf{i} \\ \mathbf{f} \\ \mathbf{o} \\ \mathbf{g} \end{pmatrix} &= \begin{pmatrix} \text{sigmoid} \\ \text{sigmoid} \\ \text{sigmoid} \\ \tanh \end{pmatrix} \mathbf{W} \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \\ \mathbf{s}_t &= \mathbf{s}_{t-1} \odot \mathbf{f}_t + \mathbf{i}_t \odot \mathbf{g}_t \\ \mathbf{h}_t &= \tanh(\mathbf{s}_t) \odot \mathbf{o}_t, \end{aligned} \tag{2}$$

where $\mathbf{s}$ is the cell state; $\mathbf{g}$ is the activation function applied on the input; $\mathbf{i}$, $\mathbf{o}$ and $\mathbf{f}$ are three sigmoid gates ('input,' 'output,' and 'forget,' respectively); and $\mathbf{h}$ is the hidden state of the cell. W is the learned weight matrix.

In the following sections, we use the notation 'LSTM' to represent the complete update function of an LSTM cell and ignore the cell state $\mathbf{s}$ for simplicity.

3.3.2 Context within and across activities. To address the difficulties in building personalized models, we consider learning attribute embeddings and contextual embeddings associated with the current workout. Specifically, for each attribute input (e.g. user, sport type, gender), we use attribute embedding layers to obtain latent representations of these attributes:

$$\mathbf{e}_{a_i} = \mathrm{E}_{a_i}(a_i), \forall i \in [1, m] \tag{3}$$

where $\mathbf{e}_{a_i}$ is the embedding for attribute a_i.

To extract contextual features from historical activity measurements, we use two LSTMs to process the historical sequences $\mathbf{Z}$ and $\mathbf{y}'$ respectively, then concatenate them and feed them into a linear projection layer to obtain a contextual embedding:

$$\begin{aligned} \mathbf{h}_{1,t} &= \mathrm{LSTM}_1(\mathbf{z}_t, \mathbf{h}_{1,t-1}), \\ \mathbf{h}_{2,t} &= \mathrm{LSTM}_2(y'_t, \mathbf{h}_{2,t-1}), \\ \mathbf{e}_t &= \mathbf{W}_e[\mathbf{h}_{1,t}; \mathbf{h}_{2,t}] + \mathbf{b}_e, \end{aligned} \tag{4}$$

where $\mathbf{h}_{1,t}$ and $\mathbf{h}_{2,t}$ are the hidden states of the two LSTMs, $\mathbf{e}_t$ is the projected contextual embedding, and $\mathbf{W}_e$ and $\mathbf{b}_e$ are trained parameters.

After obtaining these embeddings, they are repeated at each time step and concatenated with the contextual embedding and the input variables:

$$\mathbf{u}_t = [\mathbf{x}_t; \mathbf{e}_t; \mathbf{e}_{a_1}; \ldots; \mathbf{e}_{a_m}], \tag{5}$$

where $\mathbf{u}_t$ is the new concatenated contextual sequence input at time step t. The sequences $\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_T) \in \mathbb{R}^{K \times T}$ are then fed into the cascaded predictive module for different tasks. Here K is the dimension of each $\mathbf{u}_t$.

3.3.3 FitRec. We use a 2-layer LSTM model with a projection layer on the output for the workout profile forecasting problem, which we call *FitRec*:

$$\begin{aligned} \mathbf{h}_t &= \mathrm{LSTM}(\mathbf{u}_t, \mathbf{h}_{t-1}), \\ \hat{y}_t &= \mathrm{SELU}(\mathbf{W}_{NAT}\mathbf{h}_t + \mathbf{b}_{NAT}), \end{aligned} \tag{6}$$

where $\mathbf{h}_t$ is the hidden state of the LSTM's second layer, SELU is the scaled exponential linear unit activation function and $\mathbf{W}_{NAT}$ and $\mathbf{b}_{NAT}$ are learned parameters. The MLP output layer's dimension is set as the sequence length T. As shown in Figure 3, *FitRec* will output the prediction for the whole workout (all T time steps) simultaneously.

3.3.4 FitRec-Attn. Inspired by the success of the dual-stage attention RNN in modeling sequential data [25], we build *FitRec-Attn* – a dual-stage attention encoder-decoder model on top of the attribute and contextual embedding module for short term prediction.

Given the concatenated contextual sequences $\mathbf{u}$, we process them using an encoder LSTM with input attention. Input attention helps to find the importance of each input dimension among the concatenated contextual sequences. Consider $\mathbf{u}^k = (u_1^k, \ldots, u_T^k)$ for the kth input dimension of $\mathbf{u}$. An attention score s_t^k is calculated for $\mathbf{u}^k$ and the previous encoder hidden state $\mathbf{h}_{t-1}$, which represents the importance of that dimension at time step t. The weight is used to scale the original input and build a new input sequence $\tilde{\mathbf{u}}_t = (\alpha_t^1 u_t^1, \ldots, \alpha_t^K u_t^K)$:

$$\begin{aligned} s_t^k &= \mathbf{v}_s^\top(\mathbf{W}_s[\mathbf{h}_{t-1}; \mathbf{u}^k] + \mathbf{b}_s)\ 1 \le k \le K, \\ \alpha_t^k &= \frac{\exp(s_t^k)}{\sum_{j=1}^K \exp(s_t^j)}, \end{aligned} \tag{7}$$

where α_t^k is the normalized attention score, and $\mathbf{v}_s$, $\mathbf{W}_s$ and $\mathbf{b}_s$ are learned parameters. The newly computed $\tilde{\mathbf{u}}_t$ is used to update the encoder LSTM_e:

$$\mathbf{h}_t = \mathrm{LSTM}_e(\mathbf{h}_{t-1}, \tilde{\mathbf{u}}_t). \tag{8}$$

Following the encoder, a decoder LSTM with temporal attention is used to predict the target y_t. Specifically, the temporal attention score is calculated for each decoder hidden state $\mathbf{d}_t$ and all encoder hidden states, which represents how much the encoder hidden state is related to the decoder. Then the attention scores are used as weights to produce a context vector $\mathbf{c}_t$:

$$\begin{aligned} l_t^i &= \mathbf{v}_l^\top(\mathbf{W}_l[\mathbf{d}_{t-1}; \mathbf{h}_i] + \mathbf{b}_l)\ 1 \le i \le T, \\ \beta_t^i &= \frac{\exp(l_t^i)}{\sum_{j=1}^T \exp(l_t^j)}, \quad \mathbf{c}_t = \sum_{i=1}^T \beta_t^i \mathbf{h}^i \end{aligned} \tag{9}$$

where l_t^i and β_t^i are the attention score and normalized attention score for the t^{th} decoder hidden state and the i^{th} encoder hidden state. $\mathbf{v}_l$, $\mathbf{W}_l$ and $\mathbf{b}_l$ are learned parameters.

Once the context vector is obtained, it is concatenated with the previous target variable y_{t-1} and used to update the decoder LSTM:

$$\begin{aligned} \tilde{y}_{t-1} &= \tilde{\mathbf{w}}^\top[y_{t-1}; \mathbf{c}_{t-1}] + \tilde{b}, \\ \mathbf{d}_t &= \mathrm{LSTM}_d(\mathbf{d}_{t-1}, \tilde{y}_{t-1}), \end{aligned} \tag{10}$$

where $\tilde{y}_{t-1}$ is a linear transformation of the concatenated context variable and target variable at time step t (as in [25]).

Given the previous target sequence $(y_1, \ldots, y_{T-1})$, each y_{t-1} is processed sequentially and $\mathbf{c}_t$ and $\mathbf{d}_t$ are updated. Finally, we obtain the context vector and hidden state $\mathbf{c}_T$ and $\mathbf{d}_T$ at the last time step T. These two are fed into a linear output layer to predict the target variable $\hat{y_T}$ as shown in Figure 3:

$$\hat{y_T} = \mathbf{W}_{AT}[\mathbf{d}_T; \mathbf{c}_T] + \mathbf{b}_{AT}, \tag{11}$$

where $\mathbf{W}_{AT}$ and $\mathbf{b}_{AT}$ are learned parameters.

3.4 Objective Function

For both workout profile forecasting and short term prediction, we use the mean squared error as our loss function:

$$\mathcal{L} = \frac{1}{|N_{train}|} \sum_{\mathbf{y} \in \mathcal{T}_{train}} \sum_{t=1}^{L} (\hat{y_t} - y_t)^2, \tag{12}$$

where $|N_{train}|$ is the number of total time steps for all workouts in the training set, and L is the total length of each workout.

4 EXPERIMENTS

In this section, we first introduce the *Endomondo* dataset, a new real-world dataset for empirical study. Then we conduct experiments on this dataset and show the performance of our models for the two prediction tasks.

4.1 The Endomondo Dataset

We collect data from *endomondo.com*. The data includes both measurements (i.e., heart rate, timestamps, distance, speed) and contextual data (i.e., longitude, latitude, altitude, gender, sport, user identity). The original data has a high rate of missing measurements for distance and speed, presumably due to the fact that these fields are only available from certain devices. Hence we further introduce two derived sequences: (1) **derived distance**: Calculate the distance between two data points given their latitudes and longitudes via the Haversine formula [26]; (2) **derived speed**: Divide the derived distance and the time interval between two data points. In our experiments, we use the derived speed instead of the original speed (which we found to be unreliable due to missing data or erroneous magnitude values).

Table 2 lists the variables we use in this work ('Seq.' denotes whether the variable is sequential or not; 'Der.' denotes that a variable is derived). Our data includes several other variables such as weather, hydration, humidity, calories, cadence, etc., though these variables are much sparser than the variables listed (some are only available for certain sport types, or from certain wearables); we leave the exploration of these data for future work.

We discard abnormal data points based on simple rules (e.g. overly large magnitude, mismatching timestamps, abrupt changes in GPS coordinates).[2] We further keep users with at least 10 workout records to make sure there is enough data to evaluate our personalization strategies. Table 3 summarizes our dataset. The original dataset contains 253,020 workout records for 1,104 users.[3] After filtering, there are 167,373 workout records for 965 users. On average each user has over 175 workout records that cover an average span of around 2 years. This makes the dataset promising for the purpose of studying long-term temporal variation among users.

[2]Which may occur due to sensor error, or more simply due to a user forgetting to disable the device after returning to their car (for example).

[3]We collected a contiguous block of workouts, after which we collected the complete workout histories for all users represented in that block.

Table 2: Description of variables

	Variable	Seq.	Unit
Measure-ment	Heart Rate	✓	Beat per Minute (BPM)
	Timestamp	✓	Unix Timestamp
	Distance	✓	Mile
	Speed	✓	Mile per Hour (MPH)
Context-ual	UserID	✗	/
	Sport	✗	/
	Gender	✗	Male, Female
	Altitude	✓	Meter
	Longitude	✓	Degree
	Latitude	✓	Degree
Derived	Der. Distance	✓	Kilometer (KM)
	Der. Speed	✓	Kilometer per Hour (KMPH)

Table 3: Endomondo dataset statistics

Statistic	Original	Filtered	Re-sampled
# of workouts	253,020	167,373	102,343
# of users	1,104	956	887
Avg. length (hours)	5.998	1.486	1.059
Avg. span (days)	/	766	733

The original sequences as collected from *Endomondo* each contain $L = 500$ data points which are not necessarily sampled in fixed-width intervals. The sampling interval may vary from seconds to minutes. Rather than considering the problem of imputing missing values across long intervals, we simply discard workouts with sampling intervals greater than 1 minute. Furthermore, the workout data are truncated to 450 time points (out of an original 500) to account for the fact that many of the workouts tend to complete before the full 500 time-steps (e.g. during a user's cooldown).

For the problem of short term prediction, it is important to have a version of the dataset with a fixed sampling interval (i.e., so that our predictions always correspond to a fixed amount of seconds in the future). To this end, we produce a second version of the dataset that is re-sampled into fixed intervals using cubic spline interpolation followed by resampling in 10-second intervals. We discard workouts in cases where resampling generates out-of-bounds values. After resampling, we further discard those workouts with fewer than 300 samples (i.e., 300×10 seconds). Finally, we maintain a subset of 102,343 workout records as shown in Table 3.

Our data are highly multi-modally distributed, as the relationship between variables for one user differs significantly from that of other users, and the structure of the data for one sport differs significantly from that of other sports (e.g. bicycling uphill vs. downhill results in quite different heart rate patterns than running uphill vs. downhill). Our experiments show that the proposed models are able to learn these complex relations among variables.

Table 4: Parameter settings in our experiments

Model	*FitRec*	*FitRec-Attn*
Hidden Size	64	64
Attribute embed. size	5	5
Contextual embed. size	64	64
Dropout	0.2	0.1
L2 Regularizer	0.01	0.002
Batch Size	512	5120
Learning Rate	0.001	0.005

4.2 Training Procedure

We sort each user's workouts in chronological order based on the first timestamp of each workout. Then we split and aggregate users' first 80% of workouts for training, the next 10% for validation and the final 10% for testing. This is to make sure the contextual information of a workout in the test set does not appear in the training set. During training, the target data are not scaled while the input data are scaled by representing the data points in terms of their z-scores.

We used Keras[4] and Pytorch[5] to implement our models. Our models are trained using Adam [14]. We set the size of the LSTM hidden state, attribute embedding and contextual embedding as 64, 5, and 64, respectively. Attribute embeddings are randomly initialized and learned from scratch. We add dropout before and after each LSTM layer and an ℓ_2 regularizer on attribute embeddings and contextual embeddings.

The best hyper-parameters are obtained via grid search on the validation set. The learning rate is chosen from {0.001,0.005,0.01}, the dropout rate from {0.1,0.2,0.3}, and our ℓ_2 regularizer from {0.002, 0.005,0.01, 0.02}. The detailed parameter settings are as shown in Table 4. Training is terminated after 50 epochs. The model that achieved the best evaluation score on the validation set was run on the test data to obtain the final performance score. To train the models with contextual embeddings, we found it more stable to use a pre-training technique. Specifically, we first initialize the model parameters using the best model without contexutal embeddings, then continue training until convergence. All code and data are available publicly.[6]

4.3 Evaluation Metrics

We choose two popular metrics in sequential modeling: the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) to evaluate performance on our two prediction tasks:

$$\text{RMSE} = \sqrt{\frac{1}{|N_{test}|} \sum_{\mathbf{y} \in \mathcal{T}_{test}} \sum_{t=1}^{L} (y_t - \hat{y_t})^2} \tag{13}$$

$$\text{MAE} = \frac{1}{|N_{test}|} \sum_{\mathbf{y} \in \mathcal{T}_{test}} \sum_{t=1}^{L} |y_t - \hat{y_t}|, \tag{14}$$

where $|N_{test}|$ is the number of total time steps of all workouts on the test set $\mathcal{T}_{test}$. Note that in our setting the two metrics have quite different semantics. For example, a model which correctly identified heart rate 'trends' but was shifted in time or scale might be considered an acceptable model; such predictions would incur a high MSE but a more moderate MAE.

[4]https://keras.io/
[5]https://pytorch.org/
[6]https://github.com/nijianmo/fit-rec

Table 5: Workout profile forecasting on speed and heart rate. *: significantly better than the second best score ($p < 0.05$).

	Speed (KMPH)		Heart rate (BPM)	
Model	RMSE	MAE	RMSE	MAE
Global mean (train)	11.997	8.532	24.302	18.185
User mean (train)	10.690	6.418	20.117	15.090
MLP	8.998	4.411	18.256	13.918
FitRec (U)	7.144	2.472	18.784	14.142
FitRec (U/S)	7.073	**2.381**	17.325	13.012
FitRec (U/S/G)	7.328	2.396	18.207	13.831
FitRec (U/S/C)	**7.025***	2.384	**17.051***	**12.847***

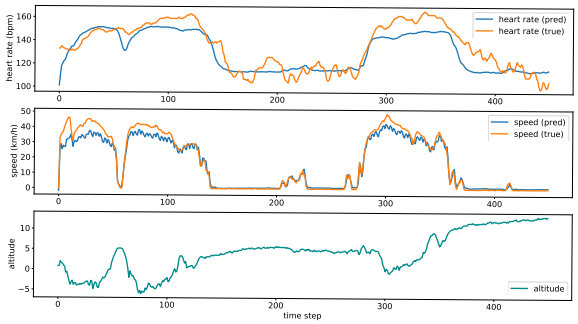
4.4 Results: Workout profile forecasting

We compare *FitRec* to two baselines and also conduct ablation tests. Our baselines are:

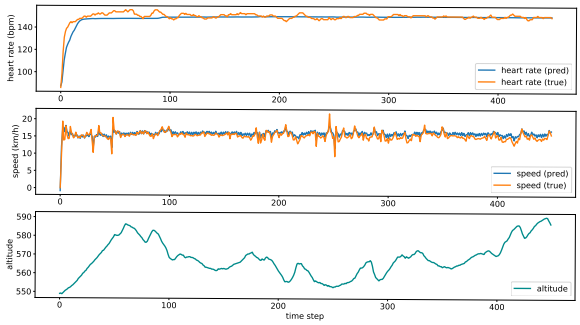
- **Global mean**. Predict the target variable at each time step as the global mean of the training set.
- **User mean**. For each workout from a user u, predict the target variable at each time step as the mean value of all u's sequences in the training set. This serves as a naïve personalized baseline.
- **MLP**. A variant of a traditional Multilayer Perceptron that concatenates the learned attribute embeddings, contextual embeddings and contextual inputs together as its input. The MLP predicts the target variable at each time step simultaneously. The dimension of the MLP's output layer is the total length of the sequence. This baseline is used to demonstrate that the LSTM component of *FitRec* is better at modeling these contextual inputs compared to an MLP.
- **FitRec** (U). *FitRec* model with user embedding only.
- **FitRec** (U/S). *FitRec* model with user embedding and sport embedding.
- **FitRec** (U/S/G). *FitRec* model with user embedding, sport embedding and gender embedding. This model is used to measure whether the gender embedding improves *FitRec*.
- **FitRec** (U/S/C). *FitRec* model with user embedding, sport embedding and contextual embedding.

We use *FitRec* to predict two types of workout profile: (1) speed and (2) heart rate, given the expected workout time and a specified route. In particular, the route is given as a distance sequence and an altitude sequence along the complete workout.

As shown in Table 5, *FitRec* (U/S/C) with all embeddings outperforms each of the traditional baselines. Specifically, it decreases the RMSE of the best baseline (MLP) by 21.93% and 6.6% on speed prediction and heart rate prediction, respectively. The user mean baseline achieves lower RMSE and MAE than the global mean baseline, demonstrating the need for personalized models for this task. *FitRec* (U) with user embeddings achieves further improvements over the user mean baseline, which shows the efficacy of the learned embeddings in capturing individual patterns of users.



(a) Biking



(b) Running

Figure 4: Examples of predicted heart rate for different sports, altitude has much greater influence on heart rate and speed change when biking as compared to running (for example).

Furthermore, our ablation tests exhibit improvements due to the attribute embeddings and the contextual embedding. *FitRec* (U/S/C) outperforms *FitRec* (U) and *FitRec* (U/S), except for the MAE on speed prediction. This may be because speed is sensitive to sport type, and the previous workout that we use to learn the contextual embedding has a different sport type from the current workout; in such cases the contextual embedding may not be informative. Missing data issues aside, this model could, in the future, be straightforwardly extended to consider other attributes such as weather, humidity, etc.

It is also surprising to find out that an auxiliary gender embedding does not help improve the prediction accuracy of our model: *FitRec* (U/S/G) performs worse than *FitRec* (U/S). There are a few possible reasons. One is that our data includes 863 males, 80 females and 13 unknown users. Such an imbalance in data might make it difficult to learn patterns across different genders. Another possibility is that the user embedding might already encode the user's gender information. As such, an auxiliary gender embedding does not bring any further benefit to the model.

Figure 4 gives examples of the predicted heart rate and speed for biking and running instances. In both cases the predicted sequences closely match the trend of the ground-truth during the workout. Notably, *FitRec* is able to learn the semantics of different sport types. For example, entering a downhill portion of a bike ride causes a user's heart rate to drop substantially, whereas running downhill does not result in a heart rate reduction.

Table 6: Short term heart rate prediction. *: significantly better than the second best score ($p < 0.05$).

Model	Heart rate (BPM) RMSE	MAE
Windowed MLP	3.035	1.919
Seq2Seq+A	2.807	1.720
DA-RNN	2.816	1.733
FitRec-Attn (U)	2.795	1.705
FitRec-Attn (U/C)	**2.783***	**1.695***

4.5 Results: Short term prediction

For short term prediction, we consider three baselines for auto-regressive modeling and conduct ablation tests to compare against the performance of *FitRec-Attn*.

The baselines are:

- **Windowed MLP** [10]. A conventional time series model that considers a sliding window of the target variable's previous values, and predicts the target variable at the current time step.
- **Seq2Seq+A** [6]. A competitive encoder-decoder baseline that is widely used in sequential modeling. This model includes temporal attention between the encoder and decoder.
- **DA-RNN** [25]. A state-of-the-art attention based RNN model for auto-regressive time series modeling with exogenous variables. The model includes an input attention mechanism that assigns weights to each input variable as well as temporal attention.
- **FitRec-Attn** (U). *FitRec-Attn* model with user embedding only.
- **FitRec-Attn** (U/C). *FitRec-Attn* model with user embedding and contextual embedding.

Because our ablation tests showed that the gender embeddings did not improve results for short term prediction, we did not include them in this section.

We use *FitRec-Attn* to predict the heart rate at each time step t given the speed and altitude sequences, the previous $T-1$ heart rate values, as well as the attribute embeddings and contextual embedding. We choose $T = 10$ in the experiments. We also experimented with $T = 5, 15, 20$ and found that values greater than 10 yield marginal improvement. When implementing Seq2Seq+A and DA-RNN, we only use speed and altitude sequences since they do not include the embedding modules for user and contextual embeddings. The baseline windowed MLP only considers the previous $T-1$ heart rate values as in traditional autoregressive models.

As shown in Table 6, *FitRec-Attn* (U/C) achieves the best performance compared with the other baselines. When only considering speed and altitude as inputs, attention-based Seq2Seq outperforms the more sophisticated DA-RNN. This may be because that the number of input variables is relatively small (2 in this case), leaving the input attention of DA-RNN as but a source of extraneous complexity. On the other hand, by incorporating user embeddings and contextual embeddings, *FitRec-Attn* (U/C) improves results on both metrics compared to DA-RNN. This demonstrates the effectiveness of input attention when the input variable is high dimensional.

Table 7: Route recommendation performance

	AUC	Hit@10	NDCG
User mean (train)	0.500	0.100	/
MLP	0.506	0.106	0.212
FitRec (U)	0.639	0.214	0.262
FitRec (U/S)	0.643	0.234	0.266
FitRec (U/S/C)	**0.673**	**0.272**	**0.284**

5 PERSONALIZED RECOMMENDATION

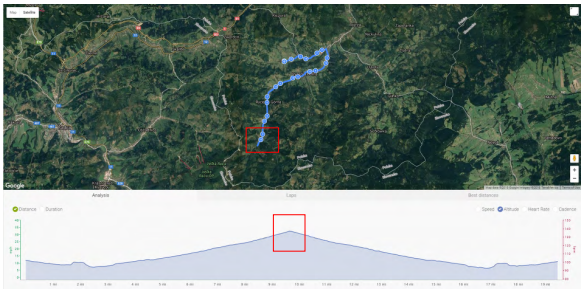
So far we have shown the effectiveness of our model at generating 'black box' heart rate predictions given contextual inputs. In this section we illustrate the ability of *FitRec* to provide personalized recommendations in practice. We consider the following two recommendation tasks:

- Workout route recommendation: Given expected workout criteria (e.g. an expected workout time, an idiosyncratic heart rate or speed curve), we suggest routes (i.e. historical routes from all users) that will match the user's expectation. This might be applied if a user has particular heart rate targets in mind, or simply wishes to identify alternate routes that are 'similar to' (in terms of heart rate profile) a route a user has taken previously. Such a system could be used (for example) by a user who is traveling but wants to maintain their regular exercise routine by identifying qualitatively similar routes.
- Short term heart rate prediction: Predict whether a user's heart rate will exceed some threshold if they continue at the current pace during their workout. This might be used to build a personalized assistant that guides a user's behavior (e.g. speed up or slow down) during a workout.

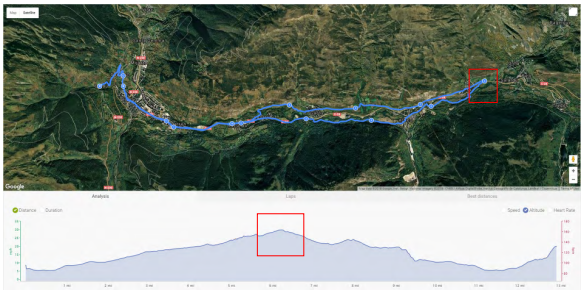
5.1 Route Recommendation

5.1.1 Evaluation Methodology. We evaluate our route recommendation task in terms of ranking performance. To do so, we first use *FitRec* to generate a ranked list of candidate routes, given a user's expected workout criteria. Then, for each workout in the test set (positive sample), we consider its route (i.e. distance and altitude sequences) as the positive route, and randomly sample another 100 workouts as negative routes. That is, given target workout criteria (heart rate etc.) as input, we are evaluating the model's ability to correctly identify the real workout that generated these criteria by assigning it a higher score than randomly chosen alternatives. Given these positive and negative routes, *FitRec* generates a predicted heart rate sequence for each of them. The similarity between the predicted sequences and the ground-truth heart rate sequence is calculated using fast dynamic time warping (fastDTW) [27]. We use DTW to measure similarity rather than the ℓ_2 loss because it is less sensitive to temporal shifting.

After sorting these similarity scores in descending order, we obtain the ranked list of all routes, and are interested in the ranked position of the positive route amongst the negative routes. Here we consider three recommendation metrics that are commonly used in ranking-based recommendation [12]: AUC, Hit Rate@K (Hit@K) and Normalized Discounted Cumulative Gain (NDCG). AUC accounts for the overall ranking of the positive route, Hit@K measures whether the positive route appears in the top K of the list, and the NDCG uses a graded relevance criterion to evaluate the overall quality of the ranking.



(a) Map of the ground-truth route.



(b) Map of the recommended alternate route.

Figure 5: Examples of a recommended route. The model recommends an alternate route that is similar to the ground-truth (i.e., both routes are comparable hill-climbs).

5.1.2 Performance Analysis. Table 7 shows the performance of route recommendation on the test set. We find that baseline models exhibit poor performance on personalized route modeling: (1) The user mean baseline does not take into account the route (i.e., distance, altitude) information; and (2) the MLP model considers routes as input during learning, but fails to rank the positive routes higher than other routes (its AUC of 0.506 is close to that of the User Mean baseline). On the other hand, *FitRec* with user embeddings is able to recognize the difference between routes and outperforms baseline models in all metrics. After adding the attribute embeddings and contextual embedding, the model's performance is further improved.

Figure 5 presents an example of a recommended route given by our model (i.e., the route ranked first on the ranked list, other than the ground truth) along with the ground-truth route. As the route map shows (marked in blue),[7] both the ground truth route and the recommended route are 'hill climbs' of approximately the same length and difficulty (the red bounding box in each route map is the region with the greatest altitude during the workout). This shows the effectiveness of the model to recommend alternate but semantically similar routes.

[7] Snapshot from *endomondo.com*

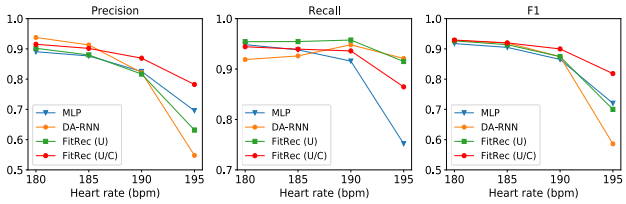


Figure 6: Performance of short-term excessive heart rate prediction

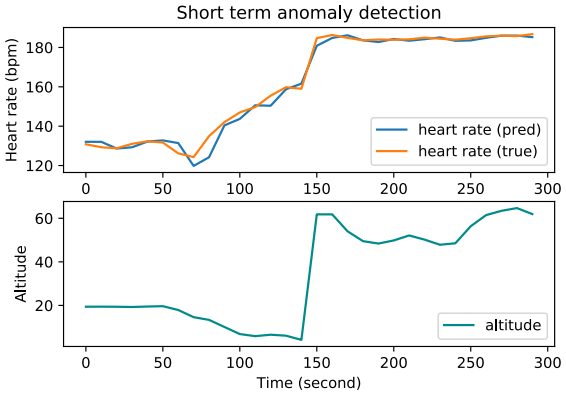


Figure 7: Example of short term excessive heart rate prediction.

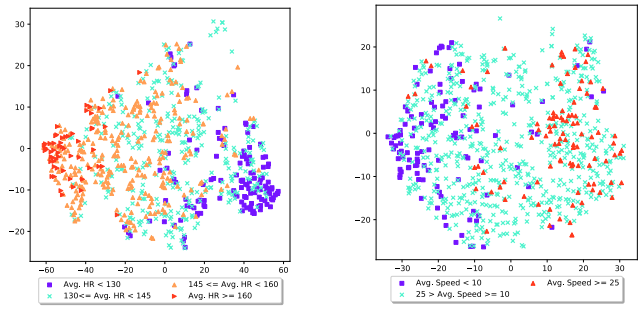
5.2 Short term heart rate prediction

The goal of this task is to predict whether a user's heart rate will exceed some level if the user continues their workout at the current speed. We assume there is a threshold value predefined by each user that she/he considers to be a desirable maximum heart rate. We consider four different threshold values: 180, 185, 190, 195 and compute three metrics to demonstrate the effectiveness of our model to anticipate rapid heart rates in test sequences.

5.2.1 Evaluation Methodology. For each workout we construct two binary sequences $(b_1, \ldots, b_L)$ and $(b_1^*, \ldots, b_L^*)$ (of predictions), where b_t / b_t^* equals 1 if the groundtruth / predicted (respectively) heart rate at timestep t exceeds a pre-defined threshold (zero otherwise). We then calculate Precision, Recall, and Macro-F1 scores based on these two binary arrays to evaluate the performance of predicting whether the speed will exceed the threshold.

5.2.2 Performance Analysis. Figure 6 demonstrates the performance of *FitRec-Attn* when given different threshold values. As we can see, *FitRec-Attn* (U/C) achieves a better F1-score compared to other baseline models for all threshold values. Specifically, the improvement of *FitRec-Attn* increases as the threshold value increases.

Figure 7 shows an example of short term heart rate prediction. Note that for this task, the model takes as input the observed heart rate values for the previous $T-1$ time steps; therefore the prediction is more accurate than in our previous experiment. In the figure we see an abrupt change in altitude, following which the model successfully predicts the heart rate will reach a high value (i.e. over 185 BPM).



(a) User embedding learned for heart rate prediction.

(b) User embedding learned for speed prediction.

Figure 8: 2d t-SNE of the learned user embedding, and its relationship to user segments in terms of heart rate and speed.

5.3 What have the user embeddings learned?

To qualitatively explain what has been captured by the learned user embeddings, we use t-SNE [22] to project them into low-dimensional vectors. Figures 8a and 8b show the 2d t-SNE of the user embeddings learned for the tasks of predicting heart rate and speed, respectively.

For heart rate, we calculate the average heart rate of each user based on all of their workouts. We classify them into four categories by threshold values: < 130, < 145, < 160, and ≥ 160 (BPM). Similarly for speed, we calculate the average speed of each user for all of their workouts. We classify them into three categories by thresholds: < 10, < 25, and ≥ 25 (KM per Hour). As shown in Figures 8a and 8b, users that fall into the same class (i.e., with similar average heart rate and average speed) seem to have closer user embeddings.

6 CONCLUSIONS

We present *FitRec* – a system that addresses sequential prediction problems in fitness and exercise data and can serve as a building block for developing personalized fitness recommendation applications. We achieve this by learning embedded representations from auxiliary information such as user identity, sport type and historical workout sequences, each of which is incorporated into an LSTM based sequential modeling framework. We verify that the model is capable of providing high quality prediction on tasks centered around estimating users' heart rate sequences, such as workout profile forecasting and short term prediction. We conduct extensive experiments on a new real-world dataset and demonstrate that the model outperforms other baselines without auxiliary embedded representations consistently on both tasks. Furthermore, the model shows promising performance on recommendation tasks such as recommending alternate workout routes and short-term excessive heart rate prediction. Visualizations also show that the learned embedded representations are useful for identifying distinct user segments. In the future, we are interested in applying *FitRec* to recommendation tasks with more complex scenarios (i.e. safety-aware route recommendation) and incorporating other auxiliary data (i.e. interactive recommendation).

REFERENCES

[1] Tim Althoff. 2017. Population-Scale Pervasive Health. *IEEE Pervasive Computing* 16 (2017), 75–79.

[2] Tim Althoff, Rok Sosic, Jennifer L. Hicks, Abby C. King, Scott L. Delp, and Jure Leskovec. 2017. Large-scale physical activity data reveal worldwide activity inequality. In *Nature*.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

[4] Hadi Banaee, Mobyen Uddin Ahmed, and Amy Loutfi. 2013. Data Mining for Wearable Sensors in Health Monitoring Systems: A Review of Recent Trends and Challenges. In *Sensors*.

[5] Zhengping Che, Xinran He, Ke Xu, and Yan Liu. 2017. DECADE: A Deep Metric Learning Model for Multivariate Time Series. In *NIPS-ML4H*.

[6] Kyunghyun Cho, Bart van Merrienboer, ÃĞaglar GülÃğehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*.

[7] Alok Kumar Chowdhury, Aleksandr Farseev, Prithwi Raj Chakraborty, Dian Tjondronegoro, and Vinod Chandran. 2017. Automatic classification of physical exercises from wearable sensors using small dataset from non-laboratory settings. *2017 IEEE Life Sciences Conference (LSC)* (2017), 111–114.

[8] Miguel Ramos de Araujo, Pedro Manuel Pinto Ribeiro, and Christos Faloutsos. 2017. TensorCast: Forecasting with Context Using Coupled Tensors (Best Paper Award). In *ICDM*.

[9] Aleksandr Farseev and Tat-Seng Chua. 2017. TweetFit: Fusing Multiple Social Media and Sensor Data for Wellness Profile Learning. In *AAAI*.

[10] Ray J. Frank, Neil Davey, and Stephen P. Hunt. 2001. Time Series Prediction and Neural Networks. *Journal of Intelligent and Robotic Systems* 31 (2001), 91–103.

[11] Tian Guo, Tao Lin, and Yao Lu. 2018. An interpretable LSTM neural network for autoregressive exogenous model. *CoRR* abs/1804.05251 (2018).

[12] Xiangnan He, Tao Chen, Min-Yen Kan, and Xiao Chen. 2015. TriRank: Review-aware Explainable Recommendation by Modeling Aspects. In *CIKM*.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation* 9 8 (1997), 1735–80.

[14] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[15] Timo Koskela, Mikko Lehtokangas, Jukka Saarinen, and Kimmo Kaski. 1996. Time Series Prediction with Multilayer Perceptron, FIR and Elman Neural Networks. In *World Congress on Neural Networks*.

[16] Takeshi Kurashima, Tim Althoff, and Jure Leskovec. 2018. Modeling Interdependent and Periodic Real-World Action Sequences. *WWW* 2018 (2018), 803–812.

[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *SIGIR*.

[18] Xiaohan Li, Shu Wu, and Liang Wang. 2017. Blood Pressure Prediction via Recurrent Models with Contextual Layer. In *WWW*.

[19] Zhiyuan Lin, Tim Althoff, and Jure Leskovec. 2018. I'll Be Back: On the Multiple Lives of Users of a Mobile Activity Tracking Application. *WWW* 2018 (2018), 1501–1511.

[20] Zachary Chase Lipton. 2015. A Critical Review of Recurrent Neural Networks for Sequence Learning. *CoRR* abs/1506.00019 (2015).

[21] Benedikt Loepp and JÃijrgen Ziegler. 2018. Recommending Running Routes: Framework and Demonstrator. In *Workshop on Recommendation in Complex Scenarios*.

[22] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.

[23] Emma Pierson, Tim Althoff, and Jure Leskovec. 2018. Modeling Individual Cyclic Variation in Human Behavior. In *WWW*.

[24] Soujanya Poria, Erik Cambria, Devamanyu Hazarika, Navonil Majumder, Amir Zadeh, and Louis-Philippe Morency. 2017. Multi-level Multiple Attentions for Contextual Multimodal Sentiment Analysis. In *ICDM*.

[25] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. In *IJCAI*.

[26] C Carl Robusto. 1957. The cosine-haversine formula. *The American Mathematical Monthly* 64, 1 (1957), 38–40.

[27] Stan Salvador and Philip Chan. 2007. Toward accurate dynamic time warping in linear time and space. *Intell. Data Anal.* 11 (2007), 561–580.

[28] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent Neural Network Regularization. *CoRR* abs/1409.2329 (2014).