Collecting and Analyzing Millions of mHealth Data Streams

Tom Quisel, Luca Foschini, Alessio Signorini
Evidation Health, Inc.
Santa Barbara, CA
{tquisel,luca,alessio}@evidation.com

David C. Kale
USC Information Sciences Institute
Marina del Rey, CA
kale@isi.edu

ABSTRACT

Players across the health ecosystem are initiating studies of thousands, even millions, of participants to gather diverse types of data, including biomedical, behavioral, and lifestyle in order to advance medical research. These efforts to collect multi-modal data sets on large cohorts coincide with the rise of broad activity and behavior tracking across industries, particularly in healthcare and the growing field of mobile health (mHealth). Government and pharmaceutical sponsored, as well as patient-driven group studies in this arena leverage the ability of mobile technology to continuously track behaviors and environmental factors with minimal participant burden. However, the adoption of mHealth has been constrained by the lack of robust solutions for large-scale data collection in free-living conditions and concerns around data quality.

In this work, we describe the infrastructure Evidation Health has developed to collect mHealth data from millions of users through hundreds of different mobile devices and apps. Additionally, we provide evidence of the utility of the data for inferring individual traits pertaining to health, wellness, and behavior. To this end, we introduce and evaluate deep neural network models that achieve high prediction performance without requiring any feature engineering when trained directly on the densely sampled multivariate mHealth time series data.

We believe that the present work substantiates both the feasibility and the utility of creating a very large mHealth research cohort, as envisioned by the many large cohort studies currently underway across therapeutic areas and conditions.

KEYWORDS

mHealth, wearables, time series, neural networks

1 INTRODUCTION

Major advancements in the prevention and treatment of diseases have stemmed from a better understanding of the factors contributing to health and disease in individual patients. Under this premise *precision medicine* has become the main idea by which "treatment and prevention can be maximized by taking into account individual variability in genes, environment, and lifestyle" [17]. In recent years, the advent and rapid adoption of *mobile health* (mHealth) enabled by wearable technologies have made continuous monitoring of environment and lifestyle ("life logging" [25]) a concrete possibility. It is estimated that 69% of the U.S. population keeps track of their weight, diet, or exercise routine, and 20% of such life loggers report leveraging technology such as digital health devices and apps to perform self monitoring [16].

mHealth data holds great promise for characterizing the environmental and lifestyle factors driving health outcomes outside traditional points of care. However, due to the novelty of the mHealth domain, the data that has been collected and published today largely comprises populations too small and with too short a duration to effectively capture the weak signals linking behavior to longer term health outcomes [15]. Much of the data is gathered in modestly sized trials [3, 39], often testing condition-specific hypotheses in controlled settings that are hard to generalize to a large-scale online population of healthy subjects.

Overcoming such limitations have driven recent efforts to scale research towards recruiting massively large patient cohorts, including some that enroll hundreds of thousands of participants, virtually without requiring visits at a care center. The Precision Medicine Initiative Cohort Program (PMI-CP) [17], Million Veteran Program [12], My Research Legacy [2], and Project Baseline [44] represent the culmination of such efforts across various health players in the United States. In the case of PMI-CP, the goal is to create and manage a national, large-scale research participant group of 1 Million patients or more. Data that participants consent for research use will include demographics, self reported measures (PROs), electronic health records, claims data, physician notes, lab tests and other biospecimen, and mHealth data (See Group et al. [17], Table 5.1.)

However, challenges with the configuration of technology around the collection and use of mHealth data have been identified. For example, while the PMI-CP Working Group has recommended that mHealth data should not be required for enrollment, they also state in their final report that "despite limited standardization of mHealth technologies, particularly commercial sensors, and a number of competing technologies, early acquisition of such data will enable exploration of use cases and facilitate building the infrastructure to handle the scale and collection of such data." ([17], Section 5.B). Perhaps even more concerning, skepticism around the *value* of the collected data has been expressed in the clinical community, with specific concerns around biases and accuracy [6].

1.1 Contribution

In the first part of the paper we describe the technology infrastructure developed at Evidation Health to perform data collection and normalization while honoring user preferences on sharing and collection modalities. We address challenges that are specific to this data collection approach, data reliability, lack of continuous

KDD'17, August 13–17, 2017, Halifax, NS, Canada.

ISBN 978-1-4503-4887-4/17/08…$15.00
DOI: http://dx.doi.org/10.1145/3097983.3098201

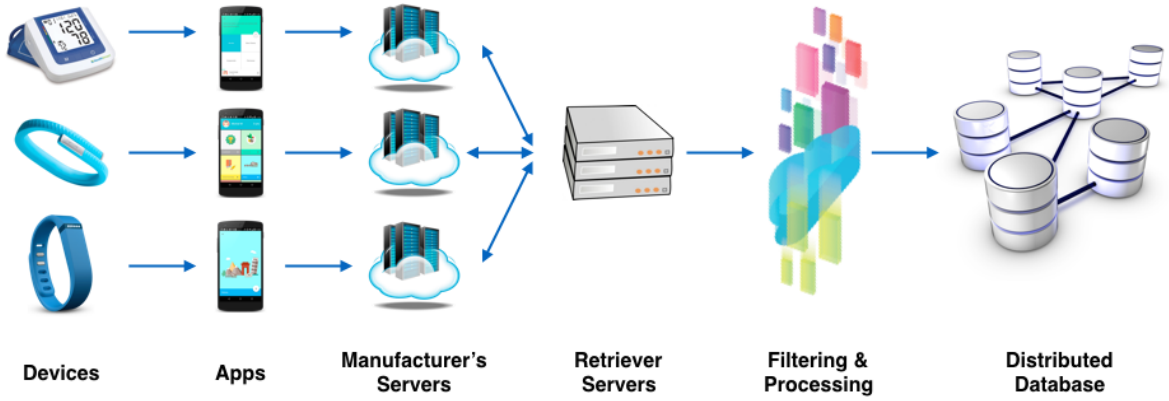

Figure 1: End-to-end data flow from the end user to the collection platform data storage.

connectivity, and data sanitization. The presented infrastructure is able to ingest data from millions of users on hundreds of different devices and apps with minute-level granularity in real time, and scales horizontally. We also describe some key learnings that hopefully can benefit the community and institutions deciding to embark on similar efforts.

In the second part of this work, we show how to leverage the collected data to perform large observational cohort studies. By framing our investigation as a set of supervised learning tasks, we show that mHealth data collected in a free-living environment has significant power to predict traits associated with wellness (BMI, weekly alcohol intake), lifestyle (food diary habits and scale utilization) and even behavioral traits (propensity to comply to an intervention, activity on a fitness-related social network). We have further extended our findings since the preparation of this manuscript by considering clinically-relevant traits such as self-reported chronic conditions [35, 36].

From a methods perspective, we show that deep neural network models (both convolutional and recurrent) achieve the best overall performance, outperforming models based on hand-engineered features. This indicates that mHealth data is fertile ground for deep learning techniques, which have the potential to bring the same ground-breaking advances that representation learning and unsupervised feature discovery have brought to other domains characterized by dense, continuous signals measured over time. Feature learning is especially valuable in a novel field such as mHealth, where feature engineering and a principled choice of generative model story are not nearly as developed as they are for clinical problems and data. Finally, we show that deep neural networks methods have the potential to learn relevant features *transferable* across data sets via a modified version of *multi-task learning* [5, 37]. This finding can prove beneficial in improving learning on different, not necessarily overlapping sub-cohorts of the same collection cohort considered for different analyses.

2 DATA ARCHITECTURE

The architecture we now describe has been deployed in production environments by Evidation Health to collect data from mHealth apps and devices from million of users over the the course of the last five years. The main challenge we had to address in developing the architecture was to cope with the fragmentation of the device and app space. Numerous commercial hardware manufacturers entered the mHealth space in recent years. The most well-known are Fitbit, Garmin, and recently Apple, but less-known ones, such as Fitbug, Striiv or Misfit, have been quickly catching up with more affordable devices offering similar functions. Manufacturers frequently have dozens (if not hundreds, as in Garmin's case) of devices models and continuously update them with improved sensors, new features, and software updates. For example, devices such as blood pressure monitors generally return only a few fixed quantities (e.g., systolic and diastolic pressure, heart rate) whereas smart watches and wristbands track a variety of parameters that range from steps taken and hours slept, to heart rate variability. As of today, the platform is capable of ingesting more than 40 types of activities (e.g., running, biking, medications taken and glucometer readings), each able to provide more than 50 different quantities (e.g., duration, distance, quantity, measurement) tracked through hundreds of devices and apps.

2.1 Data Collection

A diagram of the data collection flow is reported in Figure 1. Raw data is collected from the user either passively (e.g., via accelerometers, GPS) or actively by self report from the user. Collection happens either through a custom device (e.g., a wristband) or the mobile phone of the user. In the case of an external device data producer, a companion application is also involved. Device applications are generally lightweight and do not usually perform any intensive computation. Their purposes are mainly: (1) collect the data from the device, (2) visualize it for the user (e.g., dashboard of steps taken, miles run, etc.) and (3) transmit it to a central server under the control of the app developer or device manufacturer for storage and further processing.

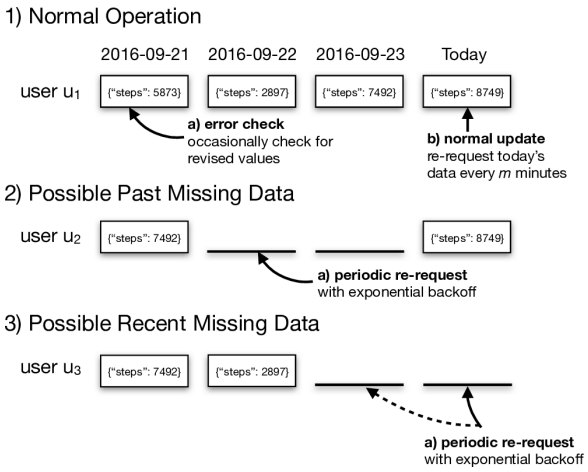

Figure 2: Strategies for handling unreliable data streams

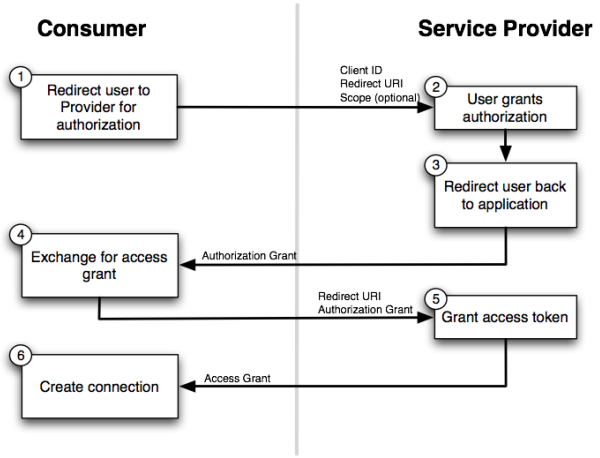

Figure 3: Example of authorization flow using OAUTHv2

While often overlooked, the design and quality of the companion application is of crucial importance for any data gathering platform. Data must be transferred from the device to the application (e.g., using Bluetooth), and then from the application to the central servers before any third party data gathering platform can retrieve it. If connectivity issues or software bugs in the application affect this process, the local user's view and what a third party may have collected and visualized will mismatch. Addressing these issues requires both a transparent user interface to avoid user confusion and a mechanism for keeping the platform's view of the data as up to date as possible while minimizing requests to the central servers. Figure 2 depicts the platform data synchronization strategy. Typical operations involve normal updating (1b), a strategy that repeatedly requests current data every m minutes. m is set adaptively for each user and device pair, and it is increased in case the manufacturer's global or per-user download rate limits are close to being reached, or if the user views their data less frequently on the platform. An error checking strategy (1a) is also employed, which uses excess capacity within the rate limits to double check old values for silent revisions. Past missing data is challenging to detect, as it is sometimes explicitly signaled during API requests with error responses, while in other cases it is implicitly signaled by 0 values. In these cases, a periodic re-request strategy (2a) is employed, which repeatedly retries data fetches with exponential backoff. Missing data can also occur contiguously at the most-recent end of the time series, in which case it may depend on the user having stopped to use the app/device, or the user having temporarily lost connectivity and the missing days to eventually be filled in. The recent missing data strategy (3a) repeatedly re-requests recent missing days with a bias towards the most recent days. The bias allows the algorithm to quickly detect the common case of a temporary stop. The strategy also performs exponential backoff, ensuring that limited resources are not spent re-requesting data in the case of a permanent stop. Finally, if new data is eventually found, all recent missing days are re-requested to handle the temporary loss-of-connectivity case, and any remaining missing days are handed off to the past missing data strategy (2a).

2.2 Authorization

Independently from how the data is accessed and formatted, a data gathering platform needs proper authorization to access user data. Although some outdated health applications require users to share their usernames and passwords with the third party platform, thereby exposing the users to risk with insecure connections, most health application favors some implementation of the OAUTH standard [18] to delegate authorization privileges. OAUTH provides a token with a set expiration date and restricted privileges to the third party platform from which it obtains limited access to the user's data. In the most common implementation of this standard, the developer redirects users to the manufacturer's website, where they are asked to login and authorize the third party to obtain the data; once that is confirmed, the user is redirected back to a predefined URL on the developer's site with special code as a parameter, which the developer's site can exchange for an access token contacting the central server (see Figure 3).

Often times, device and apps APIs implement custom versions of the OAUTH standard. Furthermore, the OAUTH standard alone introduces some confusion by providing two major versions that are mutually incompatible. Manufacturers usually incrementally move towards a standard OAUTH flow, and then to OAUTHv2 [19], each time invalidating access tokens. Additionally, OAUTH access tokens can expire, be revoked or get out of sync, causing major problems to the data gathering process that in some cases can only be resolved asking the user to reconnect their device or application with the platform. We have observed that most of APIs expire access tokens at least once a year, and on average after a month of inactivity. Over a the course of five years, for the top largest device manufacturer Evidation supports, we have witnessed at least two major service disruptions involving tokens being revoked for more than 10% of all connected devices on our platform, requiring users to reconnect their device.

2.3 Scalability

Processing steps such as de-duplication, filtering and parsing involve mostly CPU-bounded operations on each user's data streams

considered in isolation. This enables horizontal scaling of the data ingestion process described, allowing for elastic deployments that can increase the number of data processing units when needed. However, to further reduce computing resources and unnecessarily overloading the database we employ several optimization strategies, some of which are listed below.

Response Signatures. Some APIs allow to request for multiple days of data, while others may include large amounts of data (e.g., minute level data) in their replies, thus significantly increasing the size of the responses. While compression algorithms such as GZIP can help reduce data transfer times, decompressing the response will use CPU resources. Furthermore, the response format (e.g., JSON) needs to be interpreted, activities need to be parsed, and data need to be compared (and updated) in the database. Responses that are empty or contain data already fetched waste computational resources. To avoid this, we keep around per-user signatures of the responses returned for each request and abort decompressing and parsing if the responses match the previous one. For most services this reduces the CPU utilization by approximately 40%.

Data Signatures. In some cases the response of the server contains some identifier or timestamp that invalidates response signatures, thus requires decompressing and parsing the content. However, when the response is parsed, it may be still be the case that the data returned still matches that stored in the database. In a high concurrency environment limiting the number of hits to the database is important, especially when done to check for changes in the data or to update the data. To improve this data path we keep around a signature of the values parsed for each activity and we confirm that it changed before going to the database. Moreover, whenever we need to store the update in the database we take advantage of the UPSERT command offered by recent databases requiring only 1 hit for both INSERTs and UPDATEs. These optimization removed about 60% of database hits.

Data Notifications. To minimize the number of unnecessary pull requests most modern service APIs allow clients to subscribe to notifications triggered by updates in user's data. These notifications are generally sent with "best-effort" (i.e., may not be sent for all updates) but their intent is to allow clients to optimize data requests to the API. This functionality was appreciated in past years when syncing a device was more inconvenient and required a physical connection to a computer via some sort of cable, but today's devices and apps often keep very frequent communication with their servers (e.g., during a user physical activity) which in turn generates floods of notifications. Processing these notifications require computing power both in the front-end (e.g., receive the request, parse the message, schedule the appropriate action) and the back-end (e.g., fetch new data, parse response, update database). During certain activities notifications can be received every minute, generating lots of work for often small and barely significant updates. To optimize this part of the work-flow we employ algorithms that detect surges in notifications related to a user and make sure that no more than one every 12 minutes is executed. During time of intense activity this helps us cut about 80% of traffic.

Collectively, the optimizations described significantly reduces the fraction of data fetches that result in a database update. However, it is also important to limit requests to data producer servers, which enforce rate limiting policies that vary significantly across data providers. The ingestion system must honor these limitations and correctly handle the error messages, throttling the number of requests when necessary.

Data providers generally enforce two kinds of rate limiting: a per-user quota (maximum number of requests for a given user per hour) and a global quota (maximum number of request per hour across all users). To convey the best user experience, data must continuously be polled from a data provider to provide users with the most up to date view of their activity. Given the global quota constraint, as the population serviced scales, it becomes paramount to be able prioritize more frequent refreshes for users that are more active. We implemented such a prioritization by doubling the delay between consecutive updates for the same user whenever the value returned from a poll to the data provider (e.g., the number of steps taken) is unchanged, up to the maximum delay of one day. Such a policy has allowed the servicing of hundreds of thousands of daily active users, consistently guaranteeing a maximum latency - from the time the data is first made available from the device, to the time when it reported back to the user (e.g., in a dashboard) - of less than an hour.

2.4 Parsing and Normalization

Once the data has been retrieved from the data producer's servers it needs to be parsed and normalized. While parsing ad-hoc binary representations of the data can only be done through proprietary libraries, in most cases data is encoded using some form of XML or JSON, which can be efficiently decoded using standard libraries.

The lack of a standardized data schema in the industry means that every app and device manufacturer uses a proprietary nested structure to represent their data. This makes normalization to a common data schema a challenging task. The schema we have chosen is event-based, where each event corresponds to a data point made available by the data producer. While discussing the common data schema adopted is beyond of the scope of this work, it is important to consider how it can help enforce event uniqueness. Event duplication can occur when a data provider decides to republish an event that had been already transmitted in the past without modification. De-duplication must also be enforced across different devices measuring the same quantity for the same user. Typical examples are runners who wear wristbands (thus recording steps count) but also report single training sessions through runner-specific apps, resulting in double counting of steps.

Quantity normalization is also essential. Data producers may return quantities in different units of measurement, which may vary depending on the user's settings and current location. The same applies to timestamps, which may be communicated with or without timezone qualifiers. Some data providers always return the timestamp in user local time and provide the timezone in the user profile (causing issues if the profile is not fetched at reasonable intervals), others return UTC timestamps and actual timezones, and a few just UTC timestamps. Timezones are often returned following ISO-8601 standards (e.g., -HH:MM) but it is not uncommon to see abbreviations (e.g., EST) or deltas expressed as minutes, seconds or even milliseconds of difference with respect to UTC (e.g., -21600000). Handling all possible formats, variations and exceptions

requires a great deal of ad-hoc code, especially when dealing with databases (e.g., Postgres) which do not maintain timezone information associated with timestamps. In our experience, storing the timestamp in user local time (forced to UTC in the database) and the timezone in a separated field is advantageous both during data storage/retrieval and analysis.

2.5 Datastore and Processing

Similar to other large-scale event-stream use cases, we use a lambda architecture [31] to detect anomalies in real time and allow for more elaborate analyses. The real-time component of the lambda architecture is based on Elasticsearch, a distributed search engine that uses a schema-free format and allows efficient queries on raw mHealth data. Data is also converted to a wide-column relational format and stored in compressed columnar format on HDFS, where it is made available to for analysis via the Apache Spark computing engine, or queried in SQL via Presto.

3 ANALYSIS

3.1 Goal of the Analysis

In this section, we assess the utility of the data collection described in Section 2 by evaluating the prediction performance on classification tasks drawn from commonly used health questionnaires that capture lifestyle and environmental traits. An illustrative example for the specific case of cardiovascular disease is the Harvard Healthy Heart Score survey [20], which calculates a Cardiovascular Lifestyle Risk Score (CLRS) based on lifestyle habits such as smoking, physical activity, and diet. It is not surprising that, despite inaccuracies [6], tracker data can help shed light on outcomes determined by behavior. In the case of the Harvard CLRS questionnaire, an example question such as "During the past year, what was your average time per week spent on walking (slower than 3 miles per hour)?" can be answered immediately using the step count reported by a pedometer. On the other hand, for other questions of the questionnaire that cannot be directly inferred from tracker summary statistics, e.g., weekly alcohol consumption, it is less obvious whether and to what extent mHealth tracker data can be used to infer an answer. In this analysis we study the performances of models in inferring the answer to such questions as a proxy to the broader utility of the collected data.

3.2 Data Sets

The data sets considered for this analysis take advantage of the heterogeneity of the population and the types of data collected, zeroing in on a variety of different cohorts with different behaviors and kinds of data tracked. Each data set consists of a cohort of users for which the label associated with the specific prediction task (i.e., the target variable) was available. All data included in the datasets was shared by users of a commercial reward-based wellness platform powered by the the described data collection architecture in 2015 and 2016. Across all tasks, the only predictor used for each individual consisted of the individual's time series of historical daily step counts, sleep duration, and interactions with a connected scale (a binary indicator whose value is 1 if the user weighed themselves through a connected scale, and 0 otherwise). The value of the recorded weight was not used. We considered a history of 147 days across all individuals and prediction tasks. The data sets utilized in the analysis, each of which associated with a different prediction task, are described below.

Uptake. This data set consists of 1,996 users who took part in an IRB-approved study designed to increase the level of physical activity through small monetary incentives. Over a two-week intervention period, the groups were offered the same average incentives for physical activity. We considered a subset of the users in the experimental arms (the control group did not undergo the intervention) that have a history of measurements of at least 147 days. We assigned a positive label to users whose *future* median daily step count during the intervention period showed an uptake of more than 2,000 steps/day[1] compared to the median pre-intervention. This resulted in 22% positive labels.

BMI. This data set consists of the 1,978 users who have shared their BMI measurements (weight reported by a connected scale, height self-reported). We assigned a positive label to users with *future* BMI (as reported in the first half of 2016, six months after the end of the 147-day time series used as predictor) higher than a chosen clinically relevant threshold [45], which resulted in 44% positive labels.

Alcohol. This data set consists of 815 users that agreed to participate in a one-click survey answering the lifestyle question "On average, do you have more than one drink per week?" inspired by the Healthy Heart Survey [20]. We assigned a positive label to users who answered the question positively, which resulted in 33% positive labels.

Friends. This data set consists of all 16,862 users of the rewards platform who reported at least one walk, sleep, or weighing event in the first half of 2015. We assigned a positive label to any user who had at least one friend on a social network associated with the tracking device they used in the second half of 2015, resulting in 80% positive labels.

FoodLogger. This data set consists of all 16,862 users of the rewards platform who reported at least one walk, sleep, or weighing event in the first half of 2015. We gave a positive label to any user who logged at least one meal in the second half of 2015, resulting in 55% positive labels.

ScaleUsage. This data set consists of all 16,862 users of the rewards platform who reported at least one walk, sleep, or weighing event in the first half of 2015. We gave a positive label to any user who weighed themselves in the second half of 2015, resulting in 71% positive labels.

4 METHODS

The clinical research community has historically utilized modeling techniques with a few hand-picked predictor variables, with the goal of obtaining models that are easier to interpret. Recent developments in the health informatics community have seen widespread adoption of modern machine learning methods, including neural networks, due to their high performance on complex inputs such as the multivariate time series considered here [29].

We pose our predictive modeling problems as *sequence classification* tasks. Given multivariate time series $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T]$ of

[1]This level of increase in activity, if sustained in the long run, can yield significant health benefits [34]

tracked behavioral data for T days for a user, we estimate the conditional probability $p(y \mid X)$ of the target y (e.g., whether user's future BMI is above a certain threshold). In all data sets considered, $\boldsymbol{x}_t \in \mathbb{R}^5$ includes two real-valued variables (daily tracked step counts and hours slept) and three binary utilization variables, indicating whether the user logged her weight, steps, and sleep that day.

We apply two different neural network architectures to classifying behavioral sequences: temporal convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Both CNNs and RNNs are able to capture correlations across different variables and over time. While RNNs can in principle be applied to sequences of different lengths, we do not make use of this property here. All neural nets in this paper use sigmoid outputs, so that $p(y \mid X) = 1/(1 + e^{-\boldsymbol{w}^\top \boldsymbol{a}})$ where $\boldsymbol{a} = \boldsymbol{f}(X)$ is a transformation of user's time series performed by the rest of the neural net. The final output layer can be viewed as a logistic regression and the rest of the model as a feature extractor learned from data, rather than designed by hand. All neural nets are trained using gradient descent with back-propagation to minimize the negative log likelihood of the true label y: $\text{loss}(y, \hat{y}) = -(y \log \hat{y} + (1-y)\log(1-\hat{y}))$ where $\hat{y} = p(y \mid X)$. All neural nets were implemented and trained using the open source Keras package [9].

4.1 CNN Architecture

The temporal convolution neural net architecture considered is shown in Figure 4. Each individual sequence $X_u^{(k)} = [x_1^{(k)}, \ldots, x_T^{(k)}]_u$ (e.g., step counts) is fed separately to a two-stage univariate feature extractor, where each stage consists of a 1D temporal convolution followed by a non-linear activation function and a pooling operation. In contrast to Zheng et al. [46], we use a hyperbolic tangent non-linearity (tanh) and max-pooling (vs. a sigmoid and average pooling) and dropout of probability 0.5 before each pooling layer. These changes significantly improved the performance of the networks during cross validation.

The output of the feature extraction layers is flattened and fed to a standard fully connected multilayer perceptron (MLP) with one hidden layer [28]. The hidden layer uses a rectified linear activation function and dropout of probability 0.5 before the final sigmoid output.

4.2 RNN Architecture

The recurrent neural network (RNN) we consider is a simplified variant of the long short-term memory (LSTM) network [22], with gated recurrent unit (GRU) hidden layers [8]. In the GRU RNN, the persistent hidden state $\boldsymbol{s}_t^{(\ell)}$ in layer ℓ at time step t is a linear interpolation of the previous state $\boldsymbol{s}_{t-1}^{(\ell)}$ and a transient activation $\boldsymbol{h}_t^{(\ell)}$, where the interpolation weight is also a learned function of the inputs and previous state. The GRU is fully specified by the following feed-forward equations:

$$
\begin{aligned}
\boldsymbol{z}_t^{(\ell)} &= \sigma(U_z^{(\ell)} \boldsymbol{s}_t^{(\ell-1)} + W_z^{(\ell)} \boldsymbol{s}_{t-1}^{(\ell)} + b_z) \\
\boldsymbol{r}_t^{(\ell)} &= \sigma(U_r^{(\ell)} \boldsymbol{s}_t^{(\ell-1)} + W_r^{(\ell)} \boldsymbol{s}_{t-1}^{(\ell)} + b_r) \\
\boldsymbol{h}_t^{(\ell)} &= \phi(U_h^{(\ell)} \boldsymbol{s}_t^{(\ell-1)} + W_h^{(\ell)} (\boldsymbol{r} \odot \boldsymbol{s}_{t-1}^{(\ell)}) + b_h) \\
\boldsymbol{s}_t^{(\ell)} &= (1 - \boldsymbol{z}) \odot \boldsymbol{s}_{t-1}^{(\ell)} + \boldsymbol{z} \odot \boldsymbol{h}_t^{(\ell)}
\end{aligned}
$$

where $\odot$ is the element-wise product of two vectors and σ and ϕ are element-wise sigmoid and tanh functions, respectively. For layer $\ell = 1$, the input is the observation, i.e., $\boldsymbol{s}_t^{(0)} = \boldsymbol{x}_t$. $\boldsymbol{z}_t^{(\ell)}$ is the *update gate* that controls the interpolation between previous state and transient activation, while the *reset gate* $\boldsymbol{r}_t^{(\ell)}$ controls the influence of the previous state on $\boldsymbol{h}_t^{(\ell)}$.

Predictions $\hat{y}_t$ are generated at each time step by feeding the top-most hidden layer $\boldsymbol{s}_t^{(\ell)}$ into a fully connected sigmoid output. During training, we apply the *target replication* strategy proposed by Lipton et al. [29], which has been shown to improve the performance of RNNs in classifying long time series of the type analyzed here. The training loss for a single example is a weighted average of the loss for the prediction at each time step $\hat{y}_t$. We use a simplified version of the target replication loss that corresponds to setting $\alpha = 1$:

$$
\text{loss}\left(y, \{\hat{y}_t\}_{t=1}^T\right) = \frac{1}{T} \sum_{t=1}^{T} \text{loss}(y, \hat{y}_t)
$$

4.3 Multi-task Training with Fragmented Data Sets

Multi-task training can improve performance on individual tasks, especially in the absence of large labeled data sets and when the tasks are related [5, 29]. It is straightforward to train a single neural net to solve C different predictive tasks simultaneously: we add a separate output (with its own output weights $\boldsymbol{w}_c$) for each task c. The training loss for a single example with label vector $\boldsymbol{y} = [y_1, \ldots, y_C]$ is the average over the individual task losses $\text{loss}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = (1/C) \sum \text{loss}(y_c, \hat{y}_c)$.

In typical multi-task training, we have a single data set of N training examples with a full observed $\boldsymbol{y}$ for each example. Our setting differs in that our data is *fragmented*: we have C distinct data sets, one each for task, with the same set of features but only one task label y available per data set. We can still use such data to train a neural net in multi-task fashion by treating the unobserved labels as missing. We augment our data with a second set of "labels" $\boldsymbol{\delta}$, where $\delta_c = 1$ only if y_c is observed and $\delta_c = 0$ otherwise. The modified per-task training loss then becomes $\text{loss}(\boldsymbol{y}, \hat{\boldsymbol{y}}, \boldsymbol{\delta}) = (1/C) \sum \delta_c \text{ loss}(y_c, \hat{y}_c)$. In other words, if y_c is unobserved, then task c does not contribute to the loss for that example.

In the above objective function, larger data sets will contribute more heavily to the overall loss, biasing the learning toward those tasks, which may be undesirable. To combat this, we multiply each task-specific loss by a weight inversely proportional to the size of its respective training set. Specifically, for C task-specific data sets with N_c training examples each, we use the following modified loss function, where $N = \sum_c N_c$ is the total number of users included

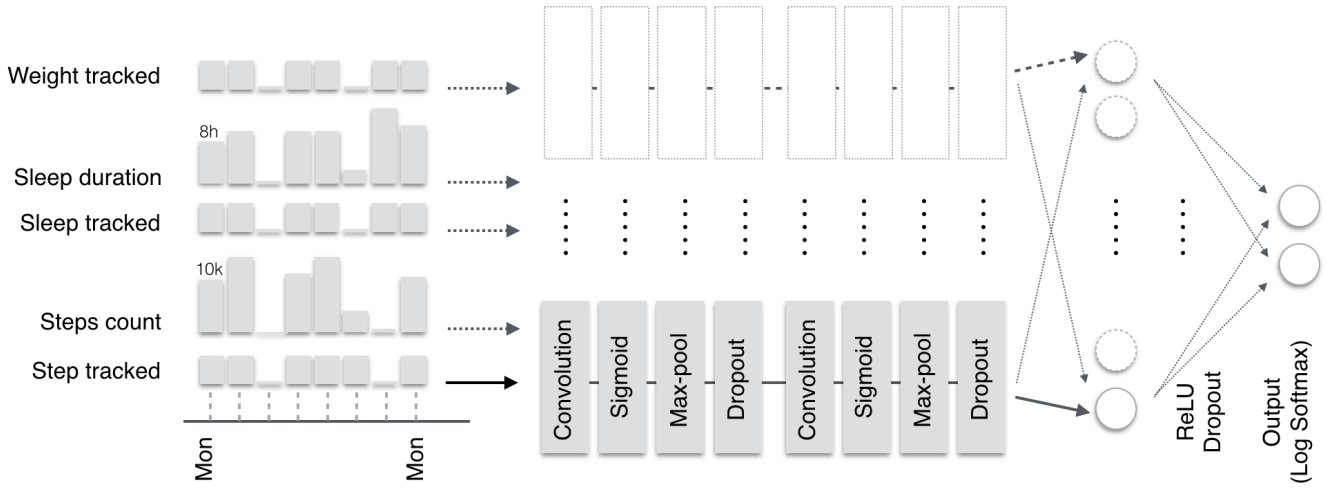

Figure 4: The temporal CNN architecture, for a single classification task. For each user, each time series is fed through the convolutional layer separately. The output layer, the only one label-specific, is a Log SoftMax classifier.

in the dataset, counted with their label multiplicity:

$$\text{loss}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{C} \sum_c \delta_c \, \frac{N}{N_c} \text{loss}(y_c, \hat{y}_c)$$

4.4 Baseline Classifiers

To quantify the effectiveness of the neural net models, we developed a set of baseline classifiers using traditional machine learning approaches. To verify that the extra complexity of the RNN and CNN models is justified, we devised a simple L2-regularized logistic regression model that uses the same multi-dimensional 147 day history of step counts, sleep duration values, and scale interactions as its input. We chose logistic regression as a baseline because it can be thought of as a neural network with zero hidden layers. The L2 regularizer ($\|\boldsymbol{w}\|_2^2$, the squared magnitude of the regression weight vector) penalizes large weights and reduces overfitting, making L2-regularized logistic regression more robust to large numbers of features and fewer training examples.

We further included a Random Forest (RF) and an L2 logistic regression classifier that use hand-engineered features in the baseline classifier set. This allows us to compare the neural net models that are based on raw data and and learn their own feature representations to traditional classifiers that are given hand-engineered features and can skip the feature-representation-learning step. We chose the hand-engineered logistic regression classifier to explore the effect on performance of switching from raw features to hand-engineered features while holding the classifier constant. The RF classifier provides an example of a powerful, ensemble-based machine learning algorithm. Hyperparameters for the classifiers were chosen using cross-validation and random search. 27 hand-generated features were used, 9 for each of the three time series. The features included statistical features that capture central tendencies, variability, and trends in the time series: mean, standard deviation, quantiles, and slope, among others.

5 EXPERIMENTAL RESULTS

We performed a series of binary classification experiments using the data sets and labels described in Section 3.2, which all consist of 147 days of 5 time series: step counts and utilization, sleep duration and utilization, and scale utilization. Our goal is to predict binary labels representing user traits related to wellness (ScaleUsage, FoodLogger), lifestyle (BMI, Alcohol) and behavior (Uptake, Friends). Since the preparation of this manuscript we have further expanded our investigation on other data sets collected through the same infrastructure described in Section 2 to predict additional clinically relevant traits [35, 36, 42].

Our final CNN architecture includes two convolutional layers of 8 and 4 filters with kernels of width 7 and 5, respectively. Both filters use step size of length 2 and are followed by a max pooling with width 2 and step size 2. The final fully connected hidden layer has 300 nodes, fed into C sigmoid outputs one per task. (In Figure 4, a single-task version of the architecture with 2 output nodes is shown.) Our final RNN architecture uses two layers of 32 GRUs each. We use dropout with probability 0.25 after each layer, L_2 weight decay of 1e-6 on all weights, and target replication. Both architectures were chosen based on validation performance and trained using the Adam variant of SGD with parameters of $\alpha = 0.005$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ [24].

We measured classifier performance using area under the receiver operating characteristic (ROC) curve, averaged across four folds. AUC, which is optimized by ranking positive examples ahead of negative examples, is an appropriate metric of success for the intended application of targeting interventions. For example, the predicted propensity of individuals to increase their physical activity as a result of a digital intervention (Uptake data set) can be used to improve triaging of interventions. More sophisticated, higher-cost interventions like in-person coaching can be targeted to individuals identified as less inclined to improve, while simpler

Table 1: AUC scores for the raw data and feature-based classifiers ± the standard error of the mean across cross-validation folds.

	RNN Raw	CNN Raw	Random Forest Features	Logistic L2 Raw	Logistic L2 Features
Overall	0.698 ± 0.005	0.695 ± 0.002	0.693 ± 0.003	0.669 ± 0.004	0.677 ± 0.002
Alcohol	0.526 ± 0.016	0.532 ± 0.014	0.569 ± 0.019	0.521 ± 0.028	0.572 ± 0.017
BMI	0.615 ± 0.014	0.642 ± 0.005	0.659 ± 0.016	0.637 ± 0.009	0.660 ± 0.017
Uptake	0.699 ± 0.008	0.689 ± 0.006	0.610 ± 0.014	0.686 ± 0.007	0.588 ± 0.012
Friend	0.739 ± 0.004	0.715 ± 0.005	0.738 ± 0.006	0.673 ± 0.007	0.695 ± 0.006
Food	0.757 ± 0.004	0.746 ± 0.004	0.744 ± 0.005	0.668 ± 0.004	0.717 ± 0.002
Weight	0.851 ± 0.003	0.848 ± 0.002	0.839 ± 0.003	0.826 ± 0.002	0.833 ± 0.002

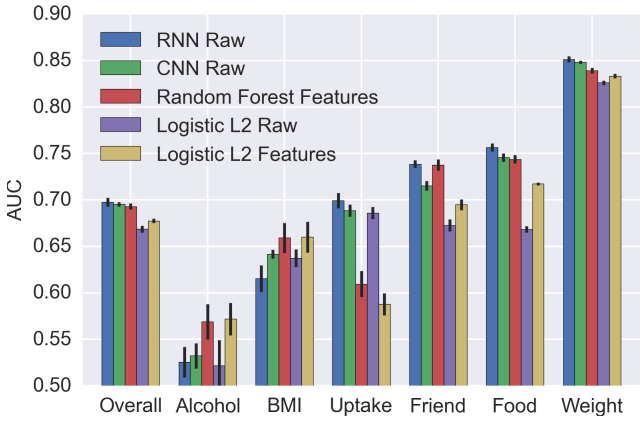

Figure 5: Plot of single-task AUC achieved with each method on each data set and overall (average of per-dataset AUCs, each weighted by the size of the data set). Error bars the standard error of the mean across cross validation folds.

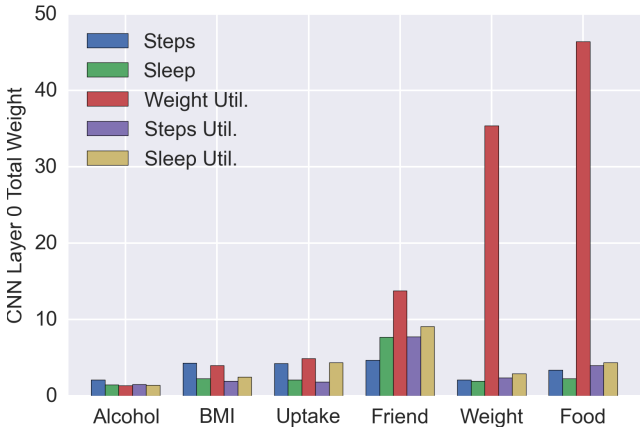

Figure 6: Sum of the absolute values of the first layer weights in the CNN model for each of the 5 input streams across each task.

and more cost-effective strategies, such as email reminder, are be sufficient for those with a higher propensity to change.

5.1 Classification Results

The per-task classification performance is summarized in Figure 5. The RNN model performs best overall, followed very closely by the CNN, with respective average AUCs of 0.698 ± 0.005 and 0.695 ± 0.002. Both are comparable to the RF with an AUC of 0.693 ± 0.003 and they outperform the logistic baselines by a large margin. The neural net models demonstrate comparable ability to model the temporal dependencies in these reasonably long behavioral biomarker time series and to predict health-related outcomes and traits. Their respective AUCs are within the margin of error for most tasks, except Friends. This leaves open the question of whether CNNs or RNNs are better suited to modeling these types of time series.

Perhaps more interesting are the tasks where the neural nets are outperformed by other models. On BMI, the features-based classifiers outperform the raw-data classifiers, indicating that not having to learn the hand-engineered features is an advantage for this data set. On Uptake, simple logistic regression using the raw time series as inputs is far superior to models using hand-engineered features and competitive with the neural nets. It is not surprising that daily step count is highly predictive of Uptake, which is itself defined as a function of future activity and steps. It seems likely that the neural nets are also able to discover this relationship and that little additional signal remains.

In Figure 6 we plot the sum of the absolute values of the first layer weights for each of the 5 input streams and for each of the tasks. The scale utilization signal appears to have high power in the ScaleUsage and FoodLogger, and, somewhat surprisingly Friends tasks, in line with previous research [4] showing that self-weighing patterns are predictive of a variety of behavioral traits.

5.2 Multi-task Training Results

We observed that the CNN underperforms most markedly on the Alcohol task and hypothesize that this is in part because it is too small to successfully train the CNN. Multi-task training is ideally suited to alleviate this issue. We start by augmenting the Alcohol data set with the Uptake data set and then using the described multi-task strategy for training the model, we achieved a significant improvement, from 0.532 AUC ± 0.013 to 0.584 AUC ± 0.013. We further explored the effect of augmenting the other tasks with the Uptake data set, and achieved the results shown in Table 2. We observed that as the size of the task increases, the benefit of

Table 2: The difference in AUC between multi-task and single-task model for each task other than Uptake. The multi-task model was trained on the combination of the primary data set and the Uptake data set. Differences are shown ± the standard error of the mean difference across cross-validation folds.

Data Size	Primary Task AUC Improvement
815	0.052 ± 0.019
1978	−0.007 ± 0.008
16862	−0.029 ± 0.007

multi-task training diminishes. Interestingly, we observed that applying multi-task training to all the C tasks at the same does not significantly increase the model performance.

6 RELATED WORK

Due to the novelty of mHealth, we are not aware of any research describing the implementation and deployment of a large-scale production mHealth data collection infrastructure. Proof of concept studies have been proposed [13], especially in the IoT literature (see Islam et al. [23]) which mostly focused on medical data transmission and interoperability, rather than the applications needed to enable a research platform. For a survey considering applications in pervasive health see Triantafyllidis et al. [43], which lists several research platforms, each rarely exceeding hundreds of users.

From an analysis perspective, few studies consider large cohorts of mHealth data collected in free-living conditions [1, 33], and most of the previous work has focused on small-sample and condition-specific trials. For a comprehensive review on such research, with an emphasis on mobile sensor capabilities, see Pejovic and Musolesi [32].

From a methods standpoint, modeling of temporal health data has been utilized by researchers in the medical machine learning community for tasks ranging from early detection and prediction [21, 41] to clustering and subtyping [30, 40]. The success of deep learning in other fields has generated an explosion of interest in using neural networks to model temporal health data [7, 27, 38]. As noted by Lane and Georgiev [26], while mobile sensor data shares many properties with data from domains in which deep learning has been applied successfully (e.g., speech), deep models have not been thoroughly investigated for wearable data. To our knowledge, our study is the first attempt at validating and comparing deep neural networks (CNNs and RNNs) as tools for analyzing mHealth data at scale.

7 CONCLUSION

In this work we describe a platform for continuous collection of mHealth data from a very large cohort, used in production settings at Evidation Health. We provide evidence that the data collected from commercial mobile health devices can be effectively used to discover meaningful *behavioral phenotypes*. Just as in medicine where phenotypes are classically defined as observable health conditions, behavioral phenotypes are being defined as pre-clinical and clinical states that can have a substantial impact on health. Insights such as being at risk for weight gain, propensity to engage in health-promoting behaviors, or likely advances in disease severity, are valuable in efforts to improve health outcomes at scale.

While prediction performance varied significantly across the analyses considered, our results show that machine learning in general, and deep learning in particular, can identify relevant traits in mHealth data collected in free-living conditions. Our results are especially remarkable given the limitations of the data, in both scope (no physiology, e.g., heart rate) and detail (reported hours of sleep vs. fine-grained sleep data). Such limitations are quickly being removed by the advent of new consumer-grade sensors able to continuously capture physiologic quantities such as hemoglobin concentration, arterial oxygen saturation (SpO2), pulse rate (PR), perfusion index (PI), and Plethysmograph Variability Index (PVI) [10]. In fact, since the preparation of this manuscript we have shown that similar methods to those presented can be used to derive phenotypes related to clinical manifestations such as self-reported chronic conditions [35, 42], and that the quality of the phenotyping improves with increased temporal resolution and duration of the collected data [36].

The presented infrastructure can easily be extended to ingest new data sources, and the models described are able to accept new quantities measured over time as additional input channels, without requiring any change in architecture or feature engineering. The success of deep learning has been driven in large part by massive data sets. Given the growth trend of mHealth [14], we anticipate even better predictive performance as data sets from larger cohorts - for instance the recent mHealth cohort study initiatives [12, 17, 44] - and richer inputs become available over time.

We believe that our work provides a direct validation of the feasibility of such large cohort studies, enabling Bring Your Own Device (BYOD) models [11] that can significantly speed up recruiting and facilitate the large-scale collection of Patient Generated Health Data (PGHD) in free-living conditions. In the longer term, it is expected that the phenotypes learned through such research initiatives may allow a better understanding of the transition from health to disease [44] and ultimately help predict disease and health outcomes [17].

8 ACKNOWLEDGEMENTS

The authors gratefully acknowledge Evidation's Claire Meunier, Deb Kilpatrick, David Stück, Wei-Nchih Lee, and Christine Lemke for the insightful comments and suggestions. David Kale was supported by the Alfred E. Mann Innovation in Engineering Doctoral Fellowship.

REFERENCES

[1] Tim Althoff, Eric Horvitz, Ryen W White, and Jamie Zeitzer. 2017. Harnessing the web for population-scale physiological sensing: A case study of sleep and performance. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 113–122.

[2] American Heart Association. 2017. My Research Legacy. https://myresearchlegacy.org/. (2017). Accessed: 2017-06-09.

[3] Cinnamon S Bloss, Nathan E Wineinger, Melissa Peters, Debra L Boeldt, Lauren Ariniello, Ju Young Kim, Judith Sheard, Ravi Komatireddy, Paddy Barrett, and Eric J Topol. 2016. A prospective randomized trial examining health care utilization in individuals using multiple smartphone-enabled biosensors. *PeerJ* 4 (2016), e1554.

[4] Lora E Burke, Jing Wang, and Mary Ann Sevick. 2011. Self-monitoring in weight loss: a systematic review of the literature. *Journal of the American Dietetic Association* 111, 1 (2011), 92–102.

[5] Rich Caruana, Shumeet Baluja, Tom Mitchell, and others. 1996. Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation. In *Advances in Neural Information Processing Systems (NIPS) 8*. 959–965.

[6] MA Case, HA Burwick, KG Volpp, and MS Patel. 2015. Accuracy of smartphone applications and wearable devices for tracking physical activity data. *JAMA* 313, 6 (2015), 625–626. DOI: http://dx.doi.org/10.1001/jama.2014.17841 arXiv:/data/Journals/JAMA/932735/jld140043.pdf?v=635587526590670000

[7] Zhengping Che, David C. Kale, Wenzhe Li, Mohammad Taha Bahadori, and Yan Liu. 2015. Deep Computational Phenotyping. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM, 507–516.

[8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[9] François Chollet. 2015. Keras. https://github.com/fchollet/keras. (2015).

[10] Jonah Comstock. 2016. CES 2016: Running list of health and wellness devices. http://mobihealthnews.com/content/ces-2016-running-list-health-and-wellness-devices. (2016). Accessed: 2016-01-13.

[11] Stephen Joel Coons, Sonya Eremenco, J Jason Lundy, Paul O'Donohoe, Hannah O'Gorman, and William Malizia. 2015. Capturing patient-reported outcome (PRO) data electronically: the past, present, and promise of ePRO measurement in clinical trials. *The Patient-Patient-Centered Outcomes Research* 8, 4 (2015), 301–309.

[12] Department of Veterans Affairs Office of Research and Development. 2017. Million Veteran Program. https://www.research.va.gov/mvp/. (2017). Accessed: 2017-06-09.

[13] Charalampos Doukas and Ilias Maglogiannis. 2012. Bringing IoT and cloud computing towards pervasive healthcare. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*. IEEE, 922–926.

[14] C Farr. 2016. The Future of Biosensing Wearables. Rock HealthHow Fitbit Became The Next Big Thing In Corporate Wellness. http://www.fastcompany.com/3058462/how-fitbit-became-the-next-big-thing-in-corporate-wellness. (2016). Accessed: 2016-05-20.

[15] Eric A Finkelstein, Benjamin A Haaland, Marcel Bilger, Aarti Sahasranaman, Robert A Sloan, Ei Ei Khaing Nang, and Kelly R Evenson. 2016. Effectiveness of activity trackers with and without incentives to increase physical activity (TRIPPA): a randomised controlled trial. *The Lancet Diabetes & Endocrinology* 4, 12 (2016), 983–995.

[16] Susannah Fox and Maeve Duggan. 2013. *Tracking for health*. Pew Research Center's Internet & American Life Project.

[17] Precision Medicine Initiative Working Group and others. 2015. Report to the Advisory Committee to the director: The Precision Medicine Initiative Cohort Program—Building a Research Foundation for 21st Century Medicine. *Washington, DC: National Institutes of Health* (2015).

[18] Eran Hammer-Lahav. 2015. The OAuth 1.0 Protocol. RFC 5849. (14 Oct. 2015). DOI: http://dx.doi.org/10.17487/rfc5849

[19] Dick Hardt. 2015. The OAuth 2.0 Authorization Framework. RFC 6749. (14 Oct. 2015). DOI: http://dx.doi.org/10.17487/rfc6749

[20] Harvard University. 2013. Healthy Heart Score. https://healthyheartscore.sph.harvard.edu/. (2013). Accessed: 2016-05-20.

[21] Katharine E Henry, David N Hager, Peter J Pronovost, and Suchi Saria. 2015. A targeted real-time early warning score (TREWScore) for septic shock. *Science Translational Medicine* 7, 299 299ra122 (2015), 1–9.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[23] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. 2015. The internet of things for health care: a comprehensive survey. *IEEE Access* 3 (2015), 678–708.

[24] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[25] Quantified Self Labs. 2016. The Quantified Self. www.quantifiedself.com. (2016). Accessed: 2016-05-20.

[26] Nicholas D Lane and Petko Georgiev. 2015. Can Deep Learning Revolutionize Mobile Sensing?. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. ACM, 117–122.

[27] Thomas A Lasko, Joshua C Denny, and Mia A Levy. 2013. Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. *PloS one* 8, 6 (2013), e66341.

[28] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. 2012. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer, 9–48.

[29] Zachary C. Lipton, David C. Kale, Charles Elkan, and Randall Wetzel. 2016. Learning to Diagnose with LSTM Recurrent Neural Networks. In *Proceedings of the 2016 International Conference on Learning Representations (ICLR)*.

[30] Ben M. Marlin, David C. Kale, Robinder G. Khemani, and Randall C. Wetzel. 2012. Unsupervised Pattern Discovery in Electronic Health Care Data Using Probabilistic Clustering Models. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium (IHI)*.

[31] N. Marz and J. Warren. 2013. Big Data: Principles and best practices of scalable realtime data systems. (2013).

[32] Veljko Pejovic and Mirco Musolesi. 2015. Anticipatory mobile computing: A survey of the state of the art and research challenges. *ACM Computing Surveys (CSUR)* 47, 3 (2015), 47.

[33] Arya Pourzanjani, Tom Quisel, and Luca Foschini. Submitted. Adherent use of activity trackers is associated with weight loss. *PLOS ONE* (Submitted).

[34] Kenneth E Powell, Amanda E Paluch, and Steven N Blair. 2011. Physical activity for health: What kind? How much? How intense? On top of what? *Public Health* 32, 1 (2011), 349.

[35] Tom Quisel, David C Kale, and Luca Foschini. 2016. Intra-day Activity Better Predicts Chronic Conditions. *arXiv preprint arXiv:1612.01200* (2016).

[36] Tom Quisel, Lee Wei-Nchih, and Luca Foschini. 2017. Observation Time vs. Performance in Digital Phenotyping. In *Proceedings of The 1st ACM Workshop on Digital Biomarkers*. Association for Computing Machinery.

[37] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. 2015. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072* (2015).

[38] Narges Razavian and David Sontag. 2015. Temporal Convolutional Neural Networks for Diagnosis from Lab Tests. *CoRR* abs/1511.07938 (2015). http://arxiv.org/abs/1511.07938

[39] Sohrab Saeb, Mi Zhang, Christopher J Karr, Stephen M Schueller, Marya E Corden, Konrad P Kording, and David C Mohr. 2015. Mobile phone sensor correlates of depressive symptom severity in daily-life behavior: an exploratory study. *Journal of medical Internet research* 17, 7 (2015).

[40] Peter Schulam, Fredrick Wigley, and Suchi Saria. 2015. Clustering Longitudinal Clinical Marker Trajectories from Electronic Health Data: Applications to Phenotyping and Endotype Discovery. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[41] Ioan Stanculescu, Christopher K Williams, Yvonne Freer, and others. 2014. Autoregressive Hidden Markov Models for the Early Detection of Neonatal Sepsis. *Biomedical and Health Informatics, IEEE Journal of* 18, 5 (2014), 1560–1570.

[42] David Stück, Haraldur Tómas Hallgrímsson, Greg Ver Steeg, Alessandro Epasto, and Luca Foschini. 2017. The Spread of Physical Activity Through Social Networks. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 519–528.

[43] Andreas Triantafyllidis, Carmelo Velardo, Dario Salvi, Syed Ahmar Shah, Vassilios Koutkias, and Lionel Tarassenko. 2015. A Survey of Mobile Phone Sensing, Self-reporting and Social Sharing for Pervasive Healthcare. (2015).

[44] Verily Life Sciences, LLC. 2017. Project Baseline. https://www.projectbaseline.com. (2017). Accessed: 2017-06-09.

[45] World Health Organization. 2006. BMI Classification, Global Database on Body Mass Index. (2006).

[46] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. 2014. Time series classification using multi-channels deep convolutional neural networks. In *Web-Age Information Management*. Springer, 298–310.