Scaling Health Analytics to Millions without Compromising Privacy using Deep Distributed Behavior Models

Petar Veličković[‡*], Nicholas D. Lane[‡†], Sourav Bhattacharya[‡]
Angela Chieh[◇], Otmane Bellahsen[◇], Matthieu Vegreville[◇]

[‡]Nokia Bell Labs, [*]University of Cambridge
[◇]Nokia Digital Health – Withings, [†]University College London

ABSTRACT

People are naturally sensitive to the sharing of their health data collected by various connected consumer devices (e.g., smart scales, sleep trackers) with third parties. However, sharing this data to compute aggregate statistics and comparisons is a basic building block for a range of medical studies based on large-scale consumer devices; such studies have the potential to transform how we study disease and behavior. Furthermore, informing users as to how their health measurements and activities compare with friends, demographic peers and globally has been shown to be a powerful tool for behavior change and management in individuals. While experienced organizations can safely perform aggregate user health analysis, there is a significant need for new privacy-preserving mechanisms that enable people to engage in the same way even with *untrusted* third parties (e.g., small/recently established organizations).

In this work, we propose a new approach to this problem grounded in the use of *deep distributed behavior models*. These are discriminative deep learning models that can approximate the calculation of various aggregate functions. Models are bootstrapped with training data from a modestly sized cohort and then distributed directly to personal devices to estimate, for example, how the user (perhaps in terms of daily step counts) ranks/compares to various demographics ranges (like age and sex). Critically, the user's own data now never has to leave the device. We validate this method using a 1.2M-user 22-month dataset that spans body-weight, sleep hours and step counts collected by devices from Nokia Digital Health – Withings. Experiments show our framework remains accurate for a range of commonly used statistical aggregate functions. This result opens a powerful new paradigm for privacy-preserving analytics under which user data largely remains on personal devices, overcoming a variety of potential privacy risks.

ACM Classification Keywords

Applied Computing → Consumer Health; Security and privacy; Computing methodologies → Neural networks

Author Keywords

Deep Learning; Digital Health; User Privacy.

PervasiveHealth '17, May 23–26, 2017, Barcelona, Spain

ACM ISBN 978-1-4503-6363-1/17/05…$15.00
https://doi.org/10.1145/3154862.3154873

INTRODUCTION

Consumer devices capable of collecting health data continue to proliferate. Today, millions of users collect health information such as heart rate, step counts, sleep hours, internal temperature, body weight and glucose levels through various networked platforms [2, 7, 8, 4]. The ability to perform types of aggregate health analytics across large-scale populations is an important building block of this ecosystem. It enables powerful new methods of studying disease, behavior and physiology at much larger scales and spanning more diverse cohorts and conditions than would be otherwise feasible [33, 17, 44] – for example, studying how physical activity in daily life can impact conditions like cerebral palsy [20].

A range of challenges remain in using aggregate health data collected from consumer devices to study medical conditions. But perhaps the most severe of these is user fear over the privacy side-effects they may expose themselves to by sharing data with a variety of potentially untrusted third parties. To perform most of the consumer device based medical studies seen today users must provide their raw data and trust that methods are adopted (e.g., [34, 42]) that ensure the data is used only in ways they have consented. Organizations and companies experienced with sensitive health data (e.g., established research institutions and trusted wearable companies) can adequately protect user privacy through the usage of various best practices in data storage and access, along with adoption of appropriate privacy-preserving algorithms. However, this solution limits the range of third parties able to perform this type of medical study as these measures are costly to implement and maintain, and even once complete a considerable effort is still needed to prove to users that the third party can be trusted with data.

This issue even extends to users of consumer health devices that are not interested in providing data to studies. Almost all health devices provide user feedback regarding behaviors and measures (e.g., sleep patterns, physical activity, body weight) in the form of aggregate statistics and comparisons. For example, users (who opt-in) are ranked against their friends or similar demographic peers; one prime reason for such analysis is that it has been found to be a strong mechanism for behavior change [37], and so is readily adopted in research prototypes as well as consumer devices. The problem is that computing such aggregates and comparisons requires user data to be collected and examined centrally, exposing potential similar privacy concerns as occur in health studies. Again, this risk can be overcome if users limit themselves to the devices and systems offered by organizations that have demonstrated the

ability to handle personal health data with sufficient care; but, this can stifle innovation and prevent users from adopting a potentially excellent behavior change treatment program only because it is offered by a small less known team.

In this work, we investigate the potential to literally turn the paradigm of health analytics on-its-head by enabling the calculation of typical aggregate functions, even those that target a global- or cohort-wide view, to be pushed down to user devices within large-scale health studies and treatment programs. Existing approaches to preserving the privacy of analytics propose careful methods as to how data can be aggregated or how data, once aggregated, is computed on (e.g., [23, 24, 31, 15, 25]) – and in this way, can offer data privacy guarantees. Instead, our framework develops a method for training deep learning models (referred to as *deep distributed behavior models*), each of which estimates the output of aggregate functions based on the data of a single user. Under this approach, calculating, for example, a percentile rank of a user's month-long weight fluctuations relative to those of similar age and sex can be performed as a discriminative task by a deep distributed behavior model – a global view of user data is not required.

To validate this approach, we use *one of the largest health datasets to be ever collected from consumer-grade devices*; for over 22 months and 1.2M users, it contains daily measurements of steps, weight and sleep. We exploit this dataset to validate our techniques for learning deep models that recognize distinctive patterns and characteristics within individual-scale health data that relate to the outcomes of aggregate functions (e.g., a percentile bin). The primary cost of this training phase is an initial bootstrapping cohort, willing to provide health data to bootstrap the model – although our experiments demonstrate the size of this cohort can be manageable (for instance, 10,000 users for the total 1.2M population), and able to be amortized across many variations of aggregate functions. Crucially, once a collection of deep distributed behavior models are trained, analytics can be performed by distributing these models to the user's devices with only the classification results (and not raw data) collected by third parties. Such data can then be used in support of population-scale health studies, or as user feedback commonly provided by consumer health systems.

The scientific contributions of this work include:

- We show – *for the first time* – a design of deep distribute behavior models, a discriminative health data deep learning model that is able to reliably estimate for individuals the response of aggregate functions (e.g., percentiles), that have a global view of the population. In sharp contrast, our deep learning formulation provides the same result using only the data from the individual, once the model is trained on a small bootstrapping cohort.
- Enabled by these deep distributed behavior models, we propose a *distributed* paradigm of health analytics that is unlike virtually all existing approaches. Under this framework, subjects no longer share their raw data centrally with a third party that aggregates sensitive data to compute analytics. Instead, behavior models are distributed to users and executed locally on their devices with only the results of models (i.e., the answers to the aggregate functions) collected centrally.
- We empirically verify this method of protecting user privacy using the sleep, steps and weight information for 1.2M people world-wide as collected using consumer health devices (e.g., smartphones, networked weight scales, wearables and watches). Our results show that deep distributed behavior models are highly accurate across a range of representative aggregate functions; critically, we demonstrate these relatively complex models can also be adapted for constrained consumer devices (like scales) to run entirely locally – and can be bootstrapped from only 10,000 people.

DEEP DISTRIBUTED BEHAVIOR MODELS

Our distributed approach to health analytics is comprised of four phases that are illustrated in Figure 1. Algorithmic details of our framework are detailed in the following section.

Phase 1: Bootstrapping Behavior Models. A critical mass of data must be first collected from which our analytic-targeting behavior models can be built. But importantly, we find our design of these models only requires a surprisingly small number of bootstrapping users (see later evaluation section). For instance, experiments show that from as few as 10,000 users aggregate models are possible that remain accurate in a population of 1.2M people for behaviors like steps, sleep, and weight. Two likely sources of bootstrapping users are either a subset of the study or consumer population, who are open to their raw data being shared with a third-party organization (often motivated for the good this brings others, and advancing the knowledge of society); or, a independent cohort recruited by an organization with the specific purpose of training these deep distributed behavior models. Independent cohorts to build health systems of such scale (e.g., 10,000 users) already occur commercially; for example, Apple engaged large-scale paid data collection of people performing activities to build models used in the Apple Watch [9] – and in research, controlled data collection of this scale is also relatively common (such as [14]). Investment in such data collection is amortized across the potentially millions of users who no longer have to share their raw data.

Phase 2: Training Behavior Models. The core objective of these deep distributed behavior models is to recognize the discriminative patterns and signatures in the data of an individual user, that allows the prediction of an aggregate query. For example, a behavior model may take as input a few weeks of daily step count data for a user, and classify it to be likely as sitting in the 65th percentile for steps per day across the entire population. Our experimental results show the ability to train such models at high levels of accuracy.

Details of the training process are given in the next section, but these operations are applied to the bootstrapping data that provide, for instance, a daily time-series of sleep hours. One or more labels of ground-truth corresponding to aggregate query results are then assigned (e.g. percentile rank of sleep hours relative to other by sex and age). Training data is reused across multiple individual models that each correspond to a distinct

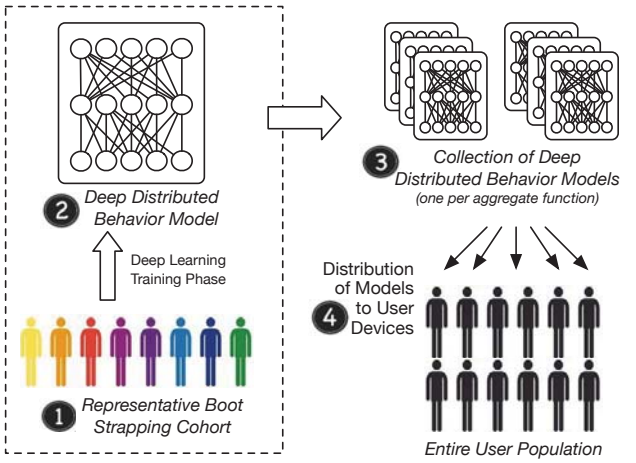


Figure 1: Overview of the distributed deep model approach to health analytics. First, models are trained under a representative cohort of modest scale. Second, models that estimate separate aggregate operations on a particular data type (like sleep hours) are distributed for local execution on user consumer devices. Raw data from the study population never has to leave the user's device, or be shared with a third party.

aggregate query (one related to sleep, the other to steps – for instance); the difference is in the labels that are applied.

All models are trained by a third party within remote cloud infrastructure. Techniques for scaling the training process of deep learning are well-known commercial practices. Thus, training behavior models aimed at data like sleep or steps, even at the scale of millions of people, should present no issues.

Phase 3: Distribute Models to Edge Devices. Once models are trained, they can be distributed to participants in a study, or users of a health system that provides some form of user feedback (e.g., system and applications targeting behavior change outcomes). Our deep distributed behavior models are able to execute directly on, for example, a smart scale [2] or a smartphone, depending on which device is best suited. Because models able to recognize the patterns of aggregate behavior can be distributed, the user's own data is never needed to be given to a third-party during the generation of the analytics. Later in our evaluation, we show that the resource footprint of behavior models (for distribution, and local execution) are feasible for typical consumer devices.

Phase 4: Result Aggregation and Model Usage. Each deep distributed behavior model given to users, or executed on an edge-device, will locally report the analytics outcome (i.e. a classification result) on the user's data. For example, a behavior model may classify a few weeks of user data as likely being in the 65th percentile for steps per day across the entire population. There will be one deep distributed behavior model executed on the user device for each aggregate query needed by the third party.

If the scenario is to study aggregate behavior within a population, then all model results are returned to the organization performing the analysis (but never raw user data). Alternatively, if aggregate statistics are to be used within a health application in order to provide feedback to users and/or understand how their actions compare with others, then results of the model can also remain local and be presented directly on a smartphone or wearable display – in this case the third party never receives the output of any of the models, nor the user data itself; the operation of the system becomes entirely distributed and self-contained. Periodically, if the aggregate behavior of users change sufficiently, the third party will need to update the distributed models – it is likely a system will need to provide some minimal information about users to the third party to help with such a decision.

It is important to note the completely different paradigm that emerges under our framework; in sharp comparison to the above, a conventional approach requires that user devices provide the third party with raw (or near-raw) user data, that is then aggregated together in a central architectural location and aggregate analytic functions applied as needed.

ALGORITHMIC FOUNDATIONS

We now detail the specific deep learning algorithms adopted to train the deep distributed behavior models that represent the building blocks of our framework. Our approach to training focuses primarily on analysing short sequences of user health metrics (viz. measurements of daily sleep, steps and weight across a 7-day period). The learning task itself is shaped by the aggregate functions we approximate for the experiments detailed in the section after this one – although this can be easily modified if other analysis is required.

Specifically, the task constitutes the assignment of the user's health measurements into one of *C bins*, roughly corresponding to assigning the user to a population percentile range, based on an aggregate metric of the sequence. The relatively short length of the sequences enables usage of a deep fully unrestricted feedforward neural network (also known as a *multilayer perceptron* or MLP) without introducing any taxing loads on low-power devices. The remainder of this section will primarily focus on detailing and justifying the design choices behind the exact MLP architecture (used in all experiments), as well as the employed training and fine-tuning methodologies.

A single *artificial neuron* has a very simple mode of operation: it computes a linear combination of its inputs, $\vec{x}$, according to a *weight vector* $\vec{w}$ (along with an additive *bias* b) and applies an *activation function*, σ, to the result:

$$h(\vec{x};\vec{w}) = \sigma\left(b + \sum_{i=1}^{n} w_i x_i\right) \quad (1)$$

A *neural network* is then constructed by connecting outputs of neurons to inputs of other neurons. In an MLP model, the neurons are organised in layers, such that every neuron in a layer receives a copy of *all* the outputs of the previous layer neurons as its inputs—enabling completely unrestricted information flow (Figure 2). Therefore, an MLP is the most potent feedforward neural network architecture, and extensions such as convolutional neural networks simply impose restrictions to this architecture to enable coping with larger scale inputs.

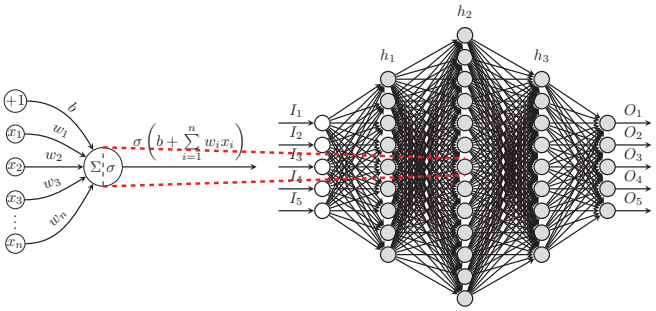


Figure 2: Left: Illustration of a single artificial neuron. Right: Representative MLP architecture used by our framework; depths of hidden layers h_n, and number of units per input ($I.$), output ($O.$) and hidden layers are based on the task at hand. The input is initialised with the data to be classified, with the output layer activated based on the inference made. Note the *deltoid* shape of the network, with gradually building up a feature representation, which is then gradually shrunk before the output stage.

Before network training can begin, several (primarily architectural) parameters, such as the number of layers, neuron counts per layer, choices of activation functions etc. need to be *fixed*—these are commonly called *hyperparameters* for distinguishability. We now turn our attention to the selected hyperparameters for our model and the design decisions underpinning their selection:

- The task of assigning inputs to one of a fixed number of bins follows the exact definition of a *classification* problem, for which it is typical to provide an output neuron per class, expressing the *probabilities* $y_i = \mathbb{P}(C_i|\vec{x},\Theta)$ that the input, $\vec{x}$, is of the ith class, based on the model parameters Θ. The class with the highest confidence is then selected.
- The dimensionalities of the intermediate (*"hidden"*) layers are selected to follow a *deltoid* shape, progressively-increasing-then-decreasing the number of features with depth. The counts are chosen to be relatively small powers of two, taking into account the small input size and the likely platforms for deployment of the model.
- For the hidden layers' neurons, we employ the *rectified linear* (ReLU) [41] activation function, variants of which have become ubiquitous for deep learning applications [38, 27, 46], as a computationally simpler and more potent alternative to the historically preferred *sigmoid* functions. We have used the simplest such variant, which sets negative neuron activations to zero:

$$\sigma(x) = \max(0,x) \tag{2}$$

- For the output neurons, we use the *softmax* activation function, which converts any real-valued vector into a vector of probabilities, while preserving monotonicity:

$$\sigma(\vec{y})_i = \frac{\exp(y_i)}{\sum_j \exp(y_j)} \tag{3}$$

With all of the above choices in mind, our MLP architecture is as follows (neuron counts denoted in parentheses):

Inp. $\rightarrow$ ReLU[64] $\rightarrow$ ReLU[128] $\rightarrow$ ReLU[64] $\rightarrow$ Softmax[C]

The training algorithm for such a network aims to estimate the values of the weights and biases of each of its constituent neurons, such that the outputs model the labels of the provided training data as precisely as possible—this precision is quantified by providing a *loss* function, which the learning algorithm seeks to minimise. For probabilistic classification tasks, a common choice is the *categorical cross-entropy loss*, which focusses solely on maximising the model's confidence in the *correct* class. It is defined as follows, for an *output* probability distribution $\vec{y}$, and its respective *ground-truth* probability distribution $\hat{\vec{y}}$:

$$\mathscr{L}\left(\vec{y},\hat{\vec{y}}\right) = -\sum_{i=1}^{C} \hat{y}_i \ln y_i \tag{4}$$

where C is the number of classes considered in the task. Typically the ground-truth distribution will be a *one-hot encoding* of the target class; e.g. for a four-way classification task, a ground-truth distribution for a target of the second class is $\hat{\vec{y}} = [0\ 1\ 0\ 0]$.

The network is trained in a typical *supervised learning* fashion, iteratively updating the weights and biases within it in order to minimise the loss function via gradient descent-based methods (propagating errors from the output neurons backwards through the hidden layers, updating their respective weights along the way).

Stochastic gradient descent (SGD) is a variant of gradient descent used for efficiently training deep models, and beyond being relatively simple to implement it offers the important benefit of coping with the challenges occurring with use of gradient approximations. In this scenario, for each iteration of the algorithm, we sample random *minibatches* $\mathscr{B} = \{(\vec{x}_1,\hat{\vec{y}}_1),\ldots,(\vec{x}_{bs},\hat{\vec{y}}_{bs})\}$ of the training data (where bs is the *batch size* hyperparameter; $bs = 128$ in our models), and aggregate their loss gradients to perform a single update as follows:

$$\vec{w}_{t+1} \leftarrow \vec{w}_t - \eta_t \sum_{j=1}^{bs} \left.\frac{\partial \mathscr{L}(h(\vec{x}_j;\vec{w}),\hat{\vec{y}}_j)}{\partial \vec{w}}\right|_{\vec{w}_t} \tag{5}$$

where η_t is the *learning rate* parameter for the current iteration. This parameter usually requires special care—and several *adaptive optimisers* exist to automatically fine-tune it based on the magnitudes of the historical updates. Here we used the *Adam* optimiser (with hyperparameters as originally given in [32]), which has been shown to perform well across a variety of learning tasks.

Aside from properly adapting the scaling of gradient updates, care has been applied to making sure that the network commences training from a position that simplifies the passage of both forward and backward signals through the layers. This has been done through utilising *He initialisation* [29] to sample initial weights for each of the hidden layers of the network—this scheme is optimised specially for facilitating training with the ReLU activation.

Finally, we have applied two regularisation techniques to prevent our network from overfitting the training data:

- *Batch normalisation* [30] (renormalising layer outputs to have zero mean and unit variance across a minibatch, followed by a trainable shift and scale to prevent losing generalisation properties) has been applied to the output of every hidden layer, to suppress any changes of signal statistical properties throughout the network (dubbed the *internal covariance shift* effect by the original authors), and found that it aided training performance significantly.
- *Dropout* [43] (randomly eliminating neurons from the network during training only, with $p = 0.25$ in our case) has been applied to the output of every hidden layer, in order to avoid the network becoming overly reliant on individual neurons' presence.

EVALUATION

In what follows, we outline the necessary preparatory steps towards verifying that our proposed model achieves all of its design goals, as well as directly reporting our results across a variety of scenarios. As will be shown, our results confirm that our model is in fact well suited for the applications previously discussed, both in terms of predictive power and deployability on low-power devices.

Dataset and Preprocessing

We performed our investigation on anonymised data obtained from several devices across the Nokia Digital Health – Withings [1] range. The dataset spans $\sim$1.2M users and contains weight, sleep and steps measurements. Weights are typically directly obtained through the Nokia Withings scale. All other data is inferred (e.g. sleep) from the Nokia Withings application through use of wearables.

The dataset was preprocessed significantly to produce a dataset suitable for training. We performed the following transformations to the data to obtain the final dataset:

- Obvious outlying users (reporting e.g. weight gains/losses of several kilograms per day) have been discarded from further consideration;
- As we do not study the effects of interpolation on predictive performance, we disregard sequences which are not contiguous for seven days in our analysis.
- Steps and sleep are recorded on a per-day basis while weight measurements are recorded at the user's discretion; to align weight measurements with the other two modalities, we applied a moving average to the person's recorded weight throughout an individual day.
- In line with known best practices in deep learning, data is normalised to have mean zero and standard deviation one per-feature.

Finally, in order to obtain different data size scenarios and to aid memory consumption, we have only processed and extracted seven-day sequences from a fragment of the raw weights file, discarding the remainder. This provided us with scenarios of medium and even small training data availability across particular gender/age categories.

Ultimately, we obtain $\sim$ 940K sleep, $\sim$ 27M steps and $\sim$ 180K weights sequences in total. This dataset contains a strong distribution that spans many combinations of age and gender; for the three data types (viz. sleep, steps and weight), when each is partitioned across 5 age ranges (30$-$, 31–40, 41–50, 51–60, 61$+$) within each sex category – at a minimum, 3,000 users are present in every age/gender combination. Because the dataset spans 1.2M users, it is not uncommon for several sequences to come from the same user. Weight sequences have been extracted especially to provide small-data scenarios – the full dataset is will be significantly larger than the steps one.

Analytics Scenarios

For the purpose of our experiments, we have partitioned all extracted sequences into 30 disjoint evaluation sets, based on three features: type of data (sleep/steps/weight), gender (male/female) and age category (30–, 31–40, 41–50, 51–60, 61+). We will refer to a "scenario" as a single such evaluation set, corresponding to one combination of these parameters. The scenarios span a wide range of data availabilities (from a few thousands to millions of sequences).

While our proposed framework allows for many elements within the privacy-preserving pipeline (such as various levels of disclosure of results, occasionally retraining/evolving the model, etc.), here we direct our attention solely to a key enabling, pipeline-agnostic issue, from which the entire system can be constructed: verifying that compact neural models for performing aggregate queries are indeed possible, and deployable on low-end devices. We leave the expansion of other aspects of the pipeline to future work.

Experiment Methodology

As described above and mentioned earlier, we primarily focus on the general task of assigning the input sequences into one of C bins. Practically, we decided to focus on classifying user sequences into one of 10 bins, based on their aggregate value. The bins' limits are inferred based on the empirical cumulative distribution function (ECDF) over the training dataset, which signifies that the i-th bin semantically corresponds to roughly a $(i \times 10)$-th percentile of the observed population. Modelling approaches similar to this have already seen application in several sensing-based domains [18, 28].

The testing is performed by partitioning the data into a training and testing dataset (holding out 25% of the data for the purposes of testing with stratification), determining the bin limits and training on the training partition, and then using the trained model to perform classification on the testing partition. For each scenario ({total sleep time, total steps taken, body weight} $\times$ {male, female} $\times$ {age 30$-$, 31–40, 41–50, 51–60, 61$+$}), we reported *confusion matrices* and compared the ground-truth histograms of the testing data (using ECDF training data) to the inferred histograms based on the model's predictions. We note that, unlike a typical classification problem, here the classes are *ordinal* and therefore we are interested not only in comparing the on/off-diagonal values of the confusion matrices, but also in asserting that any misclassifications occur without deviating too far from the main diagonal.

It is expected that varying the bin count and test holdout percentage should have a direct impact on classification performance. We have decided to focus on a single scenario (10 bins,

	Male					Female				
	30−	31–40	41–50	51–60	61+	30−	31–40	41–50	51–60	61+
Sleep	87.72%	91.24%	85.78%	83.83%	92.45%	87.12%	87.18%	90.61%	90.16%	89.34%
Steps	92.19%	93.48%	94.08%	93.20%	86.21%	90.11%	95.07%	92.37%	90.80%	90.25%
Weight	92.39%	90.67%	94.01%	94.97%	92.09%	90.39%	92.84%	90.80%	88.49%	93.92%

Table 1: The classification accuracies of our model with respect to the parameter (sleep/steps/weight) and scenario (male/female and age category) on a 25% held-out test set, after being trained on the remainder.

Type	Parameters	Architecture
DNN	17734	$7 \rightarrow 64 \rightarrow 128 \rightarrow 64 \rightarrow 10$

Table 2: Summary of the architecture of the Deep Distributed Behavioral Models used in experiments

Platform	Latency (Sec.)	Energy (J)
Cortex M0	261	22.4
Cortex M3	8.1	0.692

Table 3: Model Runtime and Energy Requirements

25% holdout), given that these effects should primarily stem from the amount of data available to the training algorithm *per-class*, and our generated data across scenarios already covers a very broad range of such data availabilities.

Model Accuracy

The results of our experiments are summarized in Table 1 and Figure 3. We report *three* kinds of evaluation metrics overall, and summarise our findings below.

- To evaluate the overall predictive power of the model, we have reported its *accuracies* on the held-out test dataset across all 30 data type/gender/age category scenarios, upon training it on the training dataset (Table 1). Given that we have generated the bins to correspond to roughly the actual 10-percentile population ranges through usage of the ECDF, the classes are almost equally represented, making accuracy an appropriate metric to report in this case. We find that the model performs demonstrably well across the entire range of data availability scenarios, consistently placing the inputs in the correct percentile ranges with accuracies between 84% and 96%.

- We selected a representative set of 6 scenarios that span a wide range of data availability levels, and reported the confusion matrices of the ground-truth percentile ranges against the model predictions (Figure 3). The findings of this analysis further compound the conclusions of the reported accuracies: the model *extremely rarely* made misclassification errors that were off by more than one percentile range (i.e. further than one cell off the main diagonal of the confusion matrix). As the original bins were empirically derived in the first place, it may be assumed that the dataset likely contains *"edge inputs"* for which the percentile range might be unclear, to which these misclassifications can be attributed, and the obtained aggregated results are still kept useful for further study without any major outliers.

- Lastly, for the same set of chosen scenarios, we report a comparison between the percentile histogram on the test dataset, and the percentile histogram implied by our model's predictions, by way of aligned bar plots (Figure 3). Observing these plots, it is also clear that the predictions follow closely the actual counts seen in the dataset.

Device Performance

As the representative embedded platform, in this paper, we consider two ARM-based microcontrollers. These processors are present in many health-targeted wearables, like thermometers and pedometers. For example, the Nokia Digital Health – Withings Steel HR smartwatch [3], that tracks vital signs like heart rate and behavioral patterns like step counts, is based on the ARM Cortex M4.

Interestingly, more powerful processors (such as the Qualcomm Snapdragon [13] and Nvidia Tegra K1 [12]) are increasingly becoming present on IoT platforms, such as Microwaves [10] and LG Smartwatches [11]. Processors of this type can efficiently run the deep distributed behavioral models we propose in this work with relative ease. Subsequently, here we focus on two constrained processors by ARM that represent more challenging execution environment that are also common for health devices. We report experiments examining the runtime and energy overhead of executing the deep models on these two processor platforms.

The ARM Cortex-M series are examples of ultra-low power wearable platforms. The smallest of them all is Cortex M0, which consumes 12.5 μW/MHz and support a memory size of 8 KB (Figure 4a). The M3 variant (Figure 4b) of the Cortex has double the processing ability (96 MHz) and supports a 32 KB memory. These low-end microcontrollers often have limited memory management capabilities. In our experiments, we could only use around 5.2 KB memory on Cortex M0 and around 28 KB memory on Cortex M3. Availability of a small memory requires frequent paging, while executing a large model. The I/O capabilities of Cortex M0 were found to be significantly slower than the Cortex M3. For prototyping, we use the MBED LPC11U24 [5] and LPC1768 [6] boards for experiments on the Cortex M0 and M3, respectively.

In Table 2, we summarize the design the Deep Distributed Behavioral Models we used in our experiments. Although the model architecture has 3 hidden layers, the overall number of tunable parameters in the model is bit over 17K. Once the models have been trained on a powerful GPU machine, we apply several runtime deep model compression techniques as presented in [35], to optimize the runtime memory and energy footprints. Table 3 summarizes the runtime and energy demands of executing these models on M0 and M3. Note

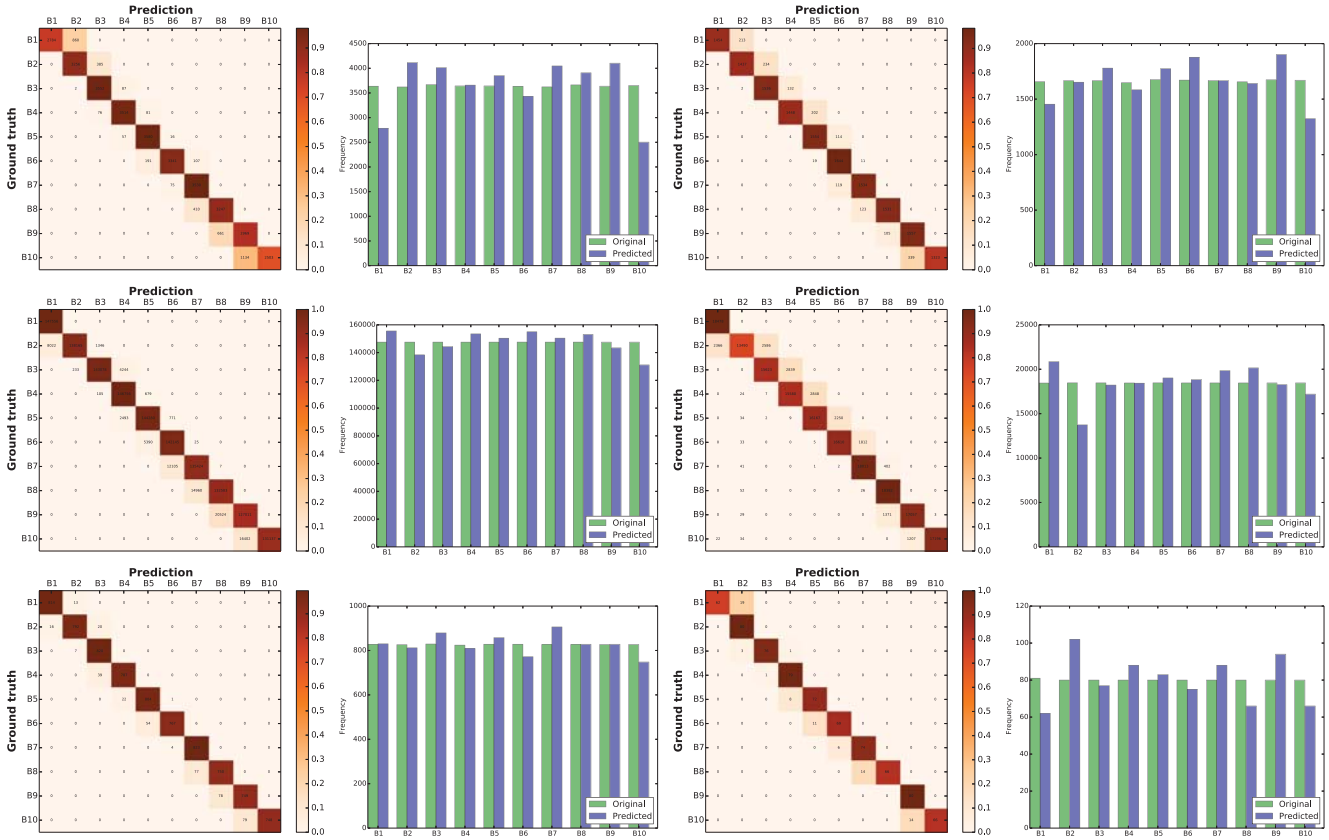
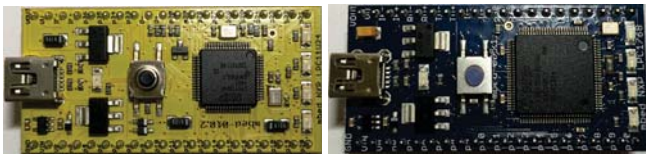


Figure 3: The representative set of obtained confusion matrices and histogram comparisons for a series of experiment scenarios covering a wide range of dataset sizes. For each scenario we illustrate a *confusion matrix* of ground-truth against predicted bins (darker colours corresponding to larger values) and the histogram of the testing dataset (green) compared to the histogram *implied by the model's predictions* on it (blue). The scenarios shown are, **left-to-right, then top-to-bottom:** Sleep, Male, age 31–40; Sleep, Female, age 51–60; Steps, Male, age 41–50; Steps, Female, age 61+; Weight, Male, age 51–60; Weight, Female, age 30−.



(a) Cortex M0 **(b)** Cortex M3

Figure 4: Hardware platforms: (a) ARM Cortex M0 and (b) M3 to evaluate DNN performances.

that, on average, the M3 takes around 8 seconds to run one inference, whereas for the M0 it is 261 seconds. The inference time is due to its smaller page size and a slower data bus.

LIMITATIONS

We briefly discuss key constraints of the results presented.

Support for Various Aggregate Computations. There is a limit to the range of aggregate functions our approach can support. For instance, it is ill-suited to computing precise values, such as means and variances for particular groups. Instead, our framework excels at operations that require comparisons or assignments of users to groups (such as percentiles). We believe that the existing privacy-preserving methods can be used to bridge the gaps in what we can support. It is also likely that our techniques can be used as building blocks, mergeable with other existing algorithms.

Privacy Concerns of Bootstrapping Models. A clear weakness of our framework is that, although the user population experiences significant privacy protection, the bootstrapping cohort must have its health data exposed. We expect that, during this phase, existing privacy-preserving methods can be adopted to provide some level of protection. Furthermore, this phase is aimed for a group of people deciding to opt-in to this process, perhaps compensated by payment or other services. Importantly, our experiments indicate that the necessary size of this cohort can be manageable (10,000) with respect to the size of the user population they can support (1.2M). As discussed in the earlier framework section, companies and researchers already run controlled studies of this size, making this step not an obstacle to adopting our approach. In a way, this bootstrapping stage transfers the privacy cost from the whole population to a much smaller group that are easier to inform, educate and monitor as to potential risks when exposing their data to untrusted third parties that may not take appropriate privacy precautions.

Updating Deep Distributed Behavior Models. As the statistical properties of the health data for the targeted population

change significantly, our behavior models will need to be retrained – at the cost of repeating the bootstrapping phase. However, we believe that future versions of our framework may incorporate a level of incremental training directly on user devices that may limit how often this needs to be done. Potentially, this may lead to fewer people needed during bootstrapping – but distributed training in this manner presents also a number of problems that are yet to be solved.

Malicious User Behavior. We must point out that our current framework does not contain provisions to explicitly prevent users manipulating their locally executing models. It is possible for them to report fake model output to manipulate analysis. However, they are only able to impact the results for their own inputs, and therefore would have to collude with large fractions of the study population to impact most forms of analysis. Furthermore, we expect countermeasures to be possible based on user devices—for example, this may be done by reporting intermediate layer states of the deep models during inference execution; these are more difficult to reliably forge than solely the model's output. This problem is not isolated to our framework only, any approach that relies on user device device can be impacted by deliberate tampering with the raw input.

RELATED WORK

Our proposed distributed approach to health analytics touches upon the following topics within existing literature.

Privacy Preserving Analytics. Early definitions and protections on aggregate data analysis like k-anonymity [24] (and l-diversity [39], m-invariance [47]) sought to restrict the resolution of the resulting analysis or data. In the case of k-anonymity, an individual is protected in the sense that at least k other users had the same/similar attributes. A wide variety of existing methods also study how noise and perturbations to source data can be added to provide guarantees of user privacy, although this is known to be challenging for mobile user sensor data [26]. Examples of such approaches include the properties of even using just random noise [31], with those like PoolView [25] being demonstrated to be particularly effective on body weight data analysis. Because our framework seeks to push the computation of analysis to the user device removing the need for data collection itself, these approaches are not as necessary – although they could be adopted in our framework during the bootstrapping of our models, or in the reporting of certain analysis outcomes. Our work fits into an rapidly expanding area that seeks to provide privacy assurances by pushing analytics into edge-devices (e.g., [19, 36]).

Differential privacy [40], in contrast to mechanisms like random perturbation or k-anonymity is typically used as an active sanitizing agent when recruiting data. It provides methods that allow data to be collected such that, in the resulting shared dataset, the presence or absence of an individual is not detectable in analysis performed on the dataset. Works like [23] that have tuned these concepts for application to health data. Once again, because our goal by using distributed deep models is to remove the need to collect raw user data, mechanisms like differential privacy are less required – note, they are likely useful only in the bootstrapping phase of our framework.

Algorithms designed for performing data mining and the aggregate analysis itself that have privacy preserving properties have also been extensively studied [15]. They examine methods by which privacy assurances can be provided as part of *how* the analysis is performed once the data is shared by users and pooled together. Such techniques potentially can be extended to use our deep models that compute aggregate estimates for single users, but this remains to be explored. Again, they are likely only applicable at our bootstrapping phases.

Large-scale Health Studies and Analytics. In addition to methods to protect user privacy, our work touches, of course, upon the existing practices of aggregate health studies using consumer devices. This is still an emerging area that began originally with pioneering work with on-line data such as surveys and social media; example health applications of this type including flu-tracking [21]. Currently, the most mature examples of this type are based on smartphone data; for example, sleep patterns with 400 users [22]. Studies that collect a wide variety of user, social and health information combine sensor data with survey instruments. EmotionSense [37] is an early example that deployed via the App Store and targeted emotion. Others have focused on particular cohorts like co-located communities [16]. Finally, [45] acted as a proof of concept for variety of forms of human study, not just healthcare, spanning multiple years and $\approx$ 1000 people. Our work strongly contrasts all of these studies in two key ways. First, none of these studies have adopted a privacy-preserving approach similar at all to what is proposed here. Second, the scale of this study is far greater than any of those reported previously; while our dataset spans 1.2M users the nearest are in the few thousands.

CONCLUSION

In this work, we propose an unconventional approach to privacy-preserving health analytics based on the *distributed* execution of deep learning models that approximate aggregate functions. Under our framework, individual deep distributed behavior models are able to learn the characteristics present within user health data (sleep hours, daily step counts or body weight) and predict how they will compare or rank with others in the population, without a global view of the data. We demonstrate that these models can even be executed on end-user devices, such as smartwatches or networked scales. As a result, users can participate in large-scale health studies by providing aggregate results to medical researchers, without sharing their personal health data. Similarly, users are able to receive feedback as to how they rank and are progressing against those of similar demographic groups – again, without sharing their personal data. The primary privacy cost for this technique is limited to the need for a relatively modest bootstrapping user cohort that must share their health data, from which a wide variety of aggregate functions can be approximated with separate deep models. We validate this approach with one of the largest real-world consumer device health datasets ever studied, that spans 22 months and 1.2M users. Our empirical findings demonstrate that, even within such a large-scale complex dataset, typical aggregate queries and comparisons still remain accurate and robust. Our work points to an exciting new paradigm for privacy-preserving health analytics from which we anticipate others will build upon.

REFERENCES

1. Nokia Digital Health – Withings. http://www.withings.com/.
2. Nokia Digital Health – Withings Smart Scale. http://www.withings.com/us/en/products/body.
3. Nokia Digital Health – Withings Steel HR. https://www.withings.com/eu/en/products/steel-hr.
4. Nokia Digital Health – Withings Thermo. http://www.withings.com/eu/en/products/thermo.
5. ARM MBED Cortex M0. https://developer.mbed.org/platforms/mbed-LPC11U24/.
6. ARM MBED Cortex M3. https://developer.mbed.org/platforms/mbed-LPC1768/.
7. Fitbit Charge HR. https://www.fitbit.com/it/chargehr.
8. HTC UA Healthbox. http://www.htc.com/us/fitness/ua-healthbox/.
9. Inside Apple's Top Secret Health and Fitness Lab for Apple Watch Development. http://abcnews.go.com/Technology/inside-apples-top-secret-health-fitness-lab-apple/story?id=29765653.
10. June Oven. https://juneoven.com/.
11. LG G Watch R. https://www.qualcomm.com/products/snapdragon/wearables/lg-g-watch-r.
12. Nvidia Tegra K1. http://www.nvidia.com/object/tegra-k1-processor.html.
13. Qualcomm Snapdragon 400. https://www.qualcomm.com/products/snapdragon/processors/400.
14. Wisconsin Longitudinal Study. http://www.ssc.wisc.edu/wlsresearch/.
15. C. C. Aggarwal and S. Y. Philip. A general survey of privacy-preserving data mining models and algorithms. In *Privacy-preserving data mining*, pages 11–52. Springer, 2008.
16. N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland. Social fmri: Investigating and shaping social mechanisms in the real world. *Pervasive Mob. Comput.*, 7(6):643–659, Dec. 2011.
17. E. Årsand, M. Muzny, M. Bradway, J. Muzik, and G. Hartvigsen. Performance of the first combined smartwatch and smartphone diabetes diary application study. *Journal of diabetes science and technology*, 9(3):556–563, 2015.
18. S. Bhattacharya, O. Huhta, and N. Asokan. Lookahead: Augmenting crowdsourced website reputation systems with predictive modeling. In *Trust 2015*, page 143-162. 2015.
19. S. Bhattacharya and N. D. Lane. From Smart to Deep: Robust Activity Recognition on Smartwatches using Deep Learning. In *IEEE WristSense 2016*, pages 1–6. 2016.
20. K. F. Bjornson, B. Belza, D. Kartin, R. Logsdon, J. McLaughlin, and E. A. Thompson. The relationship of physical activity to health status and quality of life in cerebral palsy. *Pediatric physical therapy: the official publication of the Section on Pediatrics of the American Physical Therapy Association*, 20(3):247, 2008.
21. S. J. Carlson, C. B. Dalton, M. T. Butler, J. Fejsa, E. Elvidge, and D. N. Durrheim. Flutracking weekly online community survey of influenza-like illness annual report 2011 and 2012. *NSW*, 2(4,328):4–328.
22. A. Cuttone, P. Bækgaard, V. Sekara, H. Jonsson, J. E. Larsen, and S. Lehmann. Sensiblesleep: A bayesian model for learning sleep patterns from smartphone events. *CoRR*, abs/1608.06108, 2016.
23. F. K. Dankar and K. El Emam. The application of differential privacy to health data. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, EDBT-ICDT '12, pages 158–166, New York, NY, USA, 2012. ACM.
24. K. El Emam, F. K. Dankar, R. Issa, E. Jonker, D. Amyot, E. Cogo, J.-P. Corriveau, M. Walker, S. Chowdhury, R. Vaillancourt, et al. A globally optimal k-anonymity method for the de-identification of health data. *Journal of the American Medical Informatics Association*, 16(5):670–682, 2009.
25. R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher. Poolview: Stream privacy for grassroots participatory sensing. In *SenSys '08*.
26. N. Lane, J. Xie, T. Moscibroda, and F. Zhao. On the feasibility of user de-anonymization from shared mobile sensor data. In *PhoneSense '12*.
27. X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *AIstats*, volume 15, page 275, 2011.
28. N. Y. Hammerla, R. Kirkham, P. Andras, and T. Plötz. On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution. In *ISWC '13*, pages 65–68. 2013.
29. K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV '15*, pages 1026–1034, 2015.
30. S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
31. H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *ICDM 2003*, pages 99–106. IEEE, 2003.
32. D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
33. M. Kirwan, M. J. Duncan, C. Vandelanotte, and W. K. Mummery. Using smartphone technology to monitor physical activity in the 10,000 steps program: a matched case–control trial. *Journal of medical Internet research*, 14(2):e55, 2012.
34. G. S. Kolt, G. M. Schofield, N. Kerse, N. Garrett, P. J. Schluter, T. Ashton, and A. Patel. The healthy steps study: A randomized controlled trial of a pedometer-based green prescription for older adults. trial protocol. *BMC Public Health*, 9(1):1, 2009.
35. N. D. Lane, S. Bhattacharya, C. Forlivesi, P. Georgiev, L. Jiao, L. Qendro, , and F. Kawsar. Deepx: A Software Accelerator for Low-power Deep Learning Inference on Mobile Devices. In *IPSN 2016*.
36. N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, and F. Kawsar. An early resource characterization of deep learning on wearables, smartphones and internet-of-things devices. In *IoT-App 2015*.
37. N. Lathia, V. Pejovic, K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow. Smartphones for large-scale behavior change interventions. *IEEE Pervasive Computing*, 12(3):66–73, 2013.
38. A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, volume 30, 2013.
39. A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *TKDD*, 1(1):3, 2007.
40. F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS'07*, pages 94–103. IEEE, 2007.
41. V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML '10*, pages 807–814, 2010.
42. J. D. Pillay, H. P. Ploeg, T. L. Kolbe-Alexander, K. I. Proper, M. van Stralen, S. A. Tomaz, W. Van Mechelen, and E. V. Lambert. The association between daily steps and health, and the mediating role of body composition: a pedometer-based, cross-sectional study in an employed south african population. *BMC public health*, 15(1):1, 2015.
43. N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
44. D. M. Steinberg, D. F. Tate, G. G. Bennett, S. Ennett, C. Samuel-Hodge, and D. S. Ward. The efficacy of a daily self-weighing weight loss intervention using smart scales and e-mail. *Obesity*, 21(9):1789–1797, 2013.
45. A. Stopczynski, V. Sekara, P. Sapiezynski, A. Cuttone, J. E. Larsen, and S. Lehmann. Measuring large-scale social networks with high resolution. WORKING PAPER. *CoRR*, abs/1401.7233, 2014.
46. L. Tóth. Phone recognition with deep sparse rectifier neural networks. In *ICASSP '13*, pages 6985–6989, 2013.
47. X. Xiao and Y. Tao. M-invariance: towards privacy preserving re-publication of dynamic datasets. In *SIGMOD i*, pages 689–700, 2007.