ContAuth: Continual Learning Framework for Behavioral-based User Authentication

JAGMOHAN CHAUHAN, University of Cambridge
YOUNG D. KWON, University of Cambridge
PAN HUI, University of Helsinki, Hong Kong University of Science and Technology
CECILIA MASCOLO, University of Cambridge

User authentication is key in user authorization on smart and personal devices. Over the years, several authentication mechanisms have been proposed: these also include behavioral-based biometrics. However, behavioral-based biometrics suffer from two issues: they are prone to degradation in performance (accuracy) over time (e.g., due to data distribution changes arising from user behavior) and the need to learn the machine learning model from scratch, when adding new users. In this paper, we propose *ContAuth*, a system that can enhance the robustness of behavioral-based authentication. ContAuth continuously adapts to new incoming data (*data incremental learning*) and is able to add new users without retraining (*class incremental learning*). Specifically, ContAuth combines deep learning models with online learning models to achieve learning on the fly, thereby preventing a severe drop in the accuracy between sessions (over time). To add new users, ContAuth employs class incremental learning methods. We evaluate ContAuth on multiple behavior-based user authentication modalities: breathing, gait. and EMG. Our results show that our framework can help True Positive Rate (TPR) to remain high (>85 %) compared to other methods for all the modalities except EMG (>70%) across the sessions while keeping False Positive Rates (FPR) at a minimum (0–10%). It can achieve up to 35% improvement in TPR over a traditional deep learning model. Additionally, iCaRL (an incremental learning method) enables ContAuth to allow the addition of new users by alleviating catastrophic forgetting, to a large extent. Finally, we also show that ContAuth can be deployed efficiently and effectively on device, further providing data privacy.

CCS Concepts: • **Security and Privacy** → *Security Services*; • **Human-centered computing** → *Ubiquitous and mobile computing*; • **General and reference** → *Performance.*

Additional Key Words and Phrases: Breathing Gestures, Gait, EMG, Behavior based Authentication, Neural Networks, Incremental Learning

ACM Reference Format:
Jagmohan Chauhan, Young D. Kwon, Pan Hui, and Cecilia Mascolo. 2020. ContAuth: Continual Learning Framework for Behavioral-based User Authentication. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4, Article 122 (December 2020), 23 pages. https://doi.org/10.1145/3432203

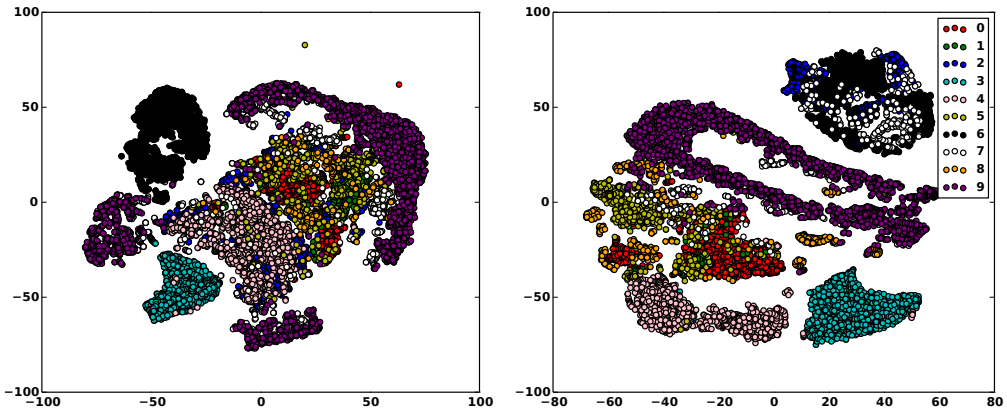
1 INTRODUCTION

Secure access to devices is becoming increasingly essential as users are storing a variety of information of sensitive nature on them. Passwords are the most primitive forms of user authentication. However, the weak security

Authors' addresses: Jagmohan Chauhan, jc2161@cam.ac.uk, University of Cambridge; Young D. Kwon, ydk21@cam.ac.uk, University of Cambridge; Pan Hui, panhui@cse.ust.hk, University of Helsinki, Hong Kong University of Science and Technology; Cecilia Mascolo, cm542@cam.ac.uk, University of Cambridge.

2474-9567/2020/12-ART122 $15.00
https://doi.org/10.1145/3432203

and significant manual effort while using passwords led to the rise of biometrics-based authentication systems. Behavioral-based biometric systems rely on human behavior and exploit the underlying differences of behavior between users to enable secure authentication. Examples of such biometrics include voice [55], touch [26, 57], heartbeats [50], and breathing [6]. Because of their inherent uniqueness, behavioral-based biometrics can provide reasonable security guarantees. However, this very useful characteristic of uniqueness of behavior can also backfire; *system performance (accuracy) may decrease between sessions as data distribution might change over time, due to changes in user behavior.* Figure 1 shows data distribution for a breathing based authentication across two sessions for ten users. The differences can be visually seen. User 6 can be seen as having a separate cluster in Session 1 (left hand side). While in Session 2, User 6 has overlapping data points with User 2 and 7 (top right side). This suggests that while it would be easy to distinguish User 6 initially from other users, it would become more difficult (later) for an authentication system to predict User 6 from incoming data correctly as User 2 and 7 would have similar data distributions. Past works including various biometrics based on acoustics [6, 29], touch [13], and eye movements [46] have also reported drops in accuracy over time. Frank et al. [13] reported a drop of 5% in the accuracy after one week for touch-based biometrics on smartphones, while Song et al. [46] saw a significant decrease (15%) when trying to unlock smartphones using eye movements. Chauhan at al. [8] reported a similar drop for breathing based authentication. Such performance degradation in an application such as authentication is considerable, as they jeopardize user security and affects user experience.



(a) Session 1 (b) Session 2

Fig. 1. TSNE plots for users in breathing based authentication across two sessions.

Another common issue that is unexplored in current authentication systems in the literature is the inability to add new users to the system, without learning the model from scratch every time a new user is enrolled. The current work assumes that a model will be trained only once when the training data from all the users is available, which might not be an acceptable assumption in real-life scenarios. Also, learning models from scratch every time will consume resources on the device. This could be particularly detrimental for resource-constrained devices. Adding users to an authentication system is needed in various scenarios. For instance, to enable authentication on IoT/edge devices: user authentication might be needed to open doors in smart spaces (offices and homes), access smart appliances and controlling voice-enabled interfaces such as Alexa. In these cases, users can be added to the authentication system on the device when required. IL (Incremental Learning) on such devices can be useful while keeping all the user data privately on the device.

Deep learning has achieved state-of-the-art results in computer vision, speech processing, and natural language processing domains and is also increasingly being used in user authentication [8, 29, 42]. However, creating a deep learning system with continual learning involves three key challenges:

(1) **Accuracy:** Because of the nature of the task, a system is acceptable only if it achieves very high accuracy. One of the issues which prevent a machine learning model from obtaining high accuracy is that of overfitting and underfitting especially when the training data size is not huge. So it is important to eliminate these issues to achieve good accuracy,
(2) **Resource Consumption:** User privacy should be preserved. Running the framework on device aids this aim. Running deep neural networks (DNN) on device is expensive, and hence the approach needs to take a resource-friendly stance while preserving user privacy,
(3) **Incremental Learning (IL):** DNN models are prone to a catastrophic forgetting (CF) issue. Solving this issue is essential to maintain high accuracy when new users are added to the system. CF makes a DNN model unlearn most of the things it has learned in the past (already existing users/classes) when encountered with a new class (user) and hence performs poorly while predicting the already known classes (in our case users).

Taking into account these challenges, we design ContAuth. ContAuth is informed by an extensive evaluation of different DNN architectures, hyperparameters and continual learning methods to perform efficient and accurate continual learning on the device for behavioral-based user authentication. ContAuth consists of a combination of deep learning and online learning (OL) models. The purpose of the deep learning model is to extract features that are then utilized for training the online model and performing real-time inference. An online model uses one example at a time and hence can learn continually intrinsically, helping maintaining accuracy. The learning of an online model is resource-efficient as these models are computationally inexpensive and require very small storage. Deep learning models can be updated from time to time, when required, to learn the changing data distributions in new data, which might correspond to user behavior changes. To achieve high accuracy, we used LSTM (Long Short Term Memory) [19] as they are the de facto methods to model time series data in deep learning. We prevent overfitting by trying to use simpler architecture (lesser number of layers and hidden units), leveraging an early stopping strategy and adopting dropout as a regularizer. Simple architectures also lead to a small memory footprint for LSTM models.

Continual learning involves neural network training and incremental learning which can be computationally expensive to perform on device due to gradient calculation and backpropagation. Epochs effectively determine the duration of the continual learning process. Hence, we focus on reducing the number of epochs. To do so, we establish an upper limit on the number of epochs an incremental learning task can reach, along with setting a threshold criteria on the loss observed during the training to perform an early stopping to reduce the number of epochs. We also try to reduce the number of time steps in LSTM networks to allow faster learning on the device. We implemented two states of the art methods for incremental learning: EWC [25] and iCaRL [39] and found iCaRL to be the best IL method which can alleviate the CF issue. However, iCaRL requires storing some training data from all the classes seen. To reduce the storage overhead on the device, we tried with different percentages of the size of the training data to make ContAuth feasible on the device. We also experimented with a transfer learning approach based on fine tuning to check how it performs compared to the IL methods. We evaluate ContAuth on three behavioral-based modalities: breathing, gait and EMG (Electromyography). Note that we mainly picked these three modalities as datasets using these modalities are open-sourced and contain different sessions which allow longitudinal analysis required for our study to be performed. Overall, the main contributions and findings of our work are as follows:

- We propose ContAuth, a system that can support continual learning of machine learning models for user authentication. ContAuth combines deep learning and online learning models. Adding users to the system

is modeled as solving a class incremental task and, thus, does not require learning DNN models from scratch each time a new user is added to the system. The novelty of our work comes from unifying two types of learning models in such a way which offers a continual learning solution that can update itself with minimal resources on a device while trying to provide high-security guarantees and usability (high false negatives often mar user experience) in the field of user authentication.

- Through extensive experiments, we show that ContAuth can achieve a True Positive Rate (TPR) of more than 85% across sessions for all the datasets except EMG. Compared to this, the joint model (traditional DNN model which is not updated) only obtains more than 45% TPR for the same setup. False Positive Rate (FPR) remains lower than 11% for ContAuth. The model sizes for the DNN (LSTM) and the online model (SGD) are small. The LSTM model size ranges from 500 KB–3300 KB, and the SGD model needs less than 25 KB of storage on the device. For the class incremental scenario, we show that iCaRL almost completely alleviates catastrophic forgetting issues, thereby allowing ContAuth to add users on the fly without impacting accuracy. Additionally, iCaRL only needs to store 5%–40% of the total training samples (budget) which requires less than 100 MB of additional storage on the device. iCaRL has been proven effective in solving vision tasks on CNN based models in the current literature. Through our work, we show the effectiveness of iCaRL for time-series mobile sensing data. We also show that it performs better than the transfer learning method.
- We did the first-ever evaluation of EMG and breathing based authentication in a longitudinal fashion. We observed that both the modalities suffer from a decrease in accuracy over time. For example, the TPR for EMG dropped to less than 40% just after one session. With ContAuth, we manage to improve the TPR by 30%–35% across sessions.
- We also explore the feasibility of performing on device incremental learning for the first time. We implemented ContAuth on Nvidia Jeston TX2 and found that data incremental learning can be performed in 25 to 300 seconds on the GPU. While an SGD based online model can be updated in 0.0045–0.0055 seconds. The class incremental learning scenario takes seven minutes to 220 minutes per task on the CPU depending on the size of the dataset, the DNN architecture, the budget size, and the number of LSTM time steps. However, the same task can be performed in 80 seconds to 17 minutes using a GPU. The majority of the total time is spent on training rather than executing an incremental learning method when executing ContAuth on CPU.

2 RELATED WORK

To motivate our work, we review recent research on the continual learning paradigms in the machine learning literature and progress made in the field of user authentication.

2.1 Continual Learning and Similar Paradigms

Continual Learning allows machine learning models to accumulate new knowledge with time from incoming data while retaining old knowledge [37]. It has also been addressed as incremental learning (IL) [39], lifelong learning [37], and sequential learning [34]. Generally, in a continual learning setting, the methods typically suffer from the issue of catastrophic forgetting (CF), thereby forgetting previously learned aspects when trying to learn new things [33, 34]. Many approaches have been proposed to mitigate or avoid CF. The first group of approaches is *regularization-based* methods [25] which add regularization terms to the loss function to minimize changes of important weights of a model for previously learned knowledge to prevent forgetting. Another group of approaches is *replay-based* methods [27] where model parameters are updated for learning a representation by using training data of the currently and sometimes previously available classes [30, 39].

There are other related fields in the machine learning literature which share similar ideas to continual learning but do not solve the issue of catastrophic forgetting. The most common and closest method is that of Transfer Learning (TL) which aims to improve the performance of a target task by using knowledge learned from a source task or domain [22, 36]. Similarly, in Domain Adaptation, the source and target tasks are the same, but the data is drawn from different domains [2]. Khan et al. [22] used domain adaptation to allow Human Activity Recognition (HAR) models trained in one context to be readily adapted to a different contextual domain. Note that unlike IL, the performance on the source task(s) is not considered during training in both Transfer Learning and Domain Adaptation and hence not applicable to be used in ContAuth. We implemented TL by fine tuning in our work and as expected found TL to be performing worse than the best performing IL method. Other methods such as Multi-Task Learning (MTL) and active learning are good at learning many tasks at the same time similar to a naive deep learning model where the number of classes is already fixed. This is completely different from our case in which classes can be incrementally learned, and hence such methods cannot be applied. MTL tries to learn across many different tasks [38, 47] at the same time while exploiting commonalities and differences across tasks. Xu et al. [38] proposed AROMA, a multi-task learning system using deep neural networks to jointly solve simple and complex activity recognition tasks. MTL assumes that simultaneous access of the model to all the tasks is possible. MTL does not include continuous adaptation after the model has been deployed in contrast to continual learning. Active Learning is another related area which provides an unsupervised way of labeling incoming data by actively querying the user or an oracle when in doubt. Active learning helps to learn a classifier in an iterative manner when the number of classes or tasks is fixed and already known [20] In this work, we are leveraging concepts from continual learning to propose a system that can incrementally add new classes (new users) for user authentication. To date, the IL methods developed in the literature have been shown to work on CNN and vision tasks. However, it is unclear if these methods will work for time series data and which method can work effectively for authentication. We are the first to explore the applicability of IL methods in this direction and their impact on resources (memory, computation time, etc.) on the device.

2.2 User Authentication

Passwords are the most primitive method of user authentication. However, they suffer from two prominent issues. The first one is that of shoulder surfing attacks which compromise the security of the user [48]. The other is to memorize and type passwords every time when accessing the device which hinders the usability of password-based schemes [41]. To address both the issues, biometrics-based access methods started to emerge. Biometrics based methods are of two types: physiological and behavioral. Physiological methods rely on the physiological traits of the person such as the face [4], palm [53], iris [31], and fingerprint [12]. Although these methods are highly usable, they often suffer from easily carried out spoofing attacks [18] and may require expensive sensors as in the case of iris scanning.

Behavioral-based biometrics rely on user behavior. Examples of behavioral-based methods include gait [10], voice [40], heartbeat [50], breathing [6] etc. Wang et al. [50] utilized vibration of the chest (as measured from the accelerometer) in response to the heartbeat as biometrics to authenticate users on smartphones. With 20 users, they could achieve an accuracy of 96.49%. Chauhan et al. [6] proposed to use the acoustics of breathing gestures (sniff, normal and deep) as recorded from the microphones of mobile devices for secure access. They achieved an accuracy of over 90% for 10 users.

Recently, with the success of deep learning in other domains such as computer vision and natural language processing, we also see an emergence of using deep learning models for authentication [7, 21, 28, 29, 45, 58]. Liu et al. [29] used DNN methods to achieve a balanced accuracy of 91.4% using an LSTM-based deep-learning model with 29 subjects for a vocal-resonance based behavioral authentication. TrueHeart [28] relies on authenticating

users on WiFi signals obtained when a user respires. EchoPrint [58] leverages acoustics and vision for secure and convenient user authentication without requiring any special hardware on a smartphone.

Behavioral-based authentication methods offer a good alternative to physiological based biometrics but suffer deterioration in the accuracy over time. Vocal resonance-based behavioral authentication [29] suffered a 5% drop in accuracy after two weeks. While breathing based authentication performance is degraded by 15% between two sessions [8]. The current works overlook this challenge. To overcome this challenge, we propose a novel system that can incrementally learn from incoming data. By analyzing the performance of our system on three different modalities (open-sourced data with different sessions), we show that our system can generalize to different types of sensor data and improve the performance of the authentication systems. To the best of our knowledge, no work has tried to address the tackled challenge across these many modalities using either simple traditional machine learning or deep learning techniques. The closest work to ours is done by Wang et al. [49]. This work shows that the performance of gait based authentication (only) can be improved by regularly updating the deep learning model. In comparison, our work employs a combination of deep learning and online learning models which can adapt faster and can be more accurate for a variety of modalities (see Section on Results). Additionally, our system can work in cases where new users are added incrementally while providing good performance, which alleviates the need to retrain the entire model from scratch.

3 SYSTEM DESIGN

This section discusses our continual learning system. We describe the overall architecture of the system followed by how it enables different types of incremental learning scenarios: data and class. Then, we describe the machine learning algorithms which form the basis of our system. Finally, we try to describe a practical scenario where such a system can be used.

3.1 Incremental Learning System

The incremental learning system comprises of deep and online machine learning models and is shown in Figure 2. Four different flows can be seen in the Figure. The decision module checks for the labels of the class and decides if the framework should perform incremental learning or learn from scratch (initial model). Initially when the decision module encounters the data for the first time, a DNN (Deep Neural Network) model is created. This is the case for the classes for which the training data is already collected. We used LSTM (Long Short Term Memory) because of their ability to model temporal relationships in the data. This initial DNN also generates features which are then used to train an online classifier. Such classifiers can learn on a sample by sample basis and hence are termed online, i.e., not batch (using multiple samples at a time to learn). We used SGD (Stochastic Gradient Descent) algorithm as an online classifier. The features are obtained from the last layer of the DNN before the softmax layer.

The incremental learning algorithms are used when the decision module encounters a new class (*class incremental learning (CIL)*) but an initial model is already present in the system. We experimented with EWC [25] and iCaRL [39] as IL algorithms as they are the most prominent and state of the art algorithms able to cope with catastrophic forgetting issues. More details about these two algorithms are presented in the next subsection. These algorithms allow our framework to incorporate new classes on the fly without retraining the whole model from scratch, which is an important advantage for efficient on-device processing.

The framework processes each test sample through two routes. Firstly, each test sample goes to the DNN where features are extracted, and then an inference decision is made on these features by the online classifier. The online classifier also updates itself at the same time from the sample. The process is quite fast, which makes it ideal to use as an online classifier in incremental setting scenarios. In this way, the online classifier always remains up to date and can adapt quickly to the changing data distribution which might correspond to changing user behavior

without consuming excessive computing resources. However, the same setting might not remain very useful if the DNN model is not updated. The reason is that the online classifier depends on features generated by the DNN, and if the DNN is not updated regularly, then the features generated would not represent the changing data distribution. Hence, the incoming test samples are stored and are used to update the DNN model and are termed as *data incremental learning (DIL)*. The decision of when to update the DNN model can be based on factors including time elapsed, need to flush stored samples if memory becomes an issue, or enough samples are collected or the availability of energy (battery level) on the device. In a nutshell, CIL represents a scenario where a new class is to be added to the system, and IL methods are employed to support the scenario. While in DIL, an already existing deep learning model (existing classes) is updated with new incoming data for existing classes (no new class added and no need to employ IL methods such as iCarL or EWC). This is usually achieved by training the existing (already saved) deep learning model for a few epochs. Note that both CIL and DIL forego the need to train a model from scratch.

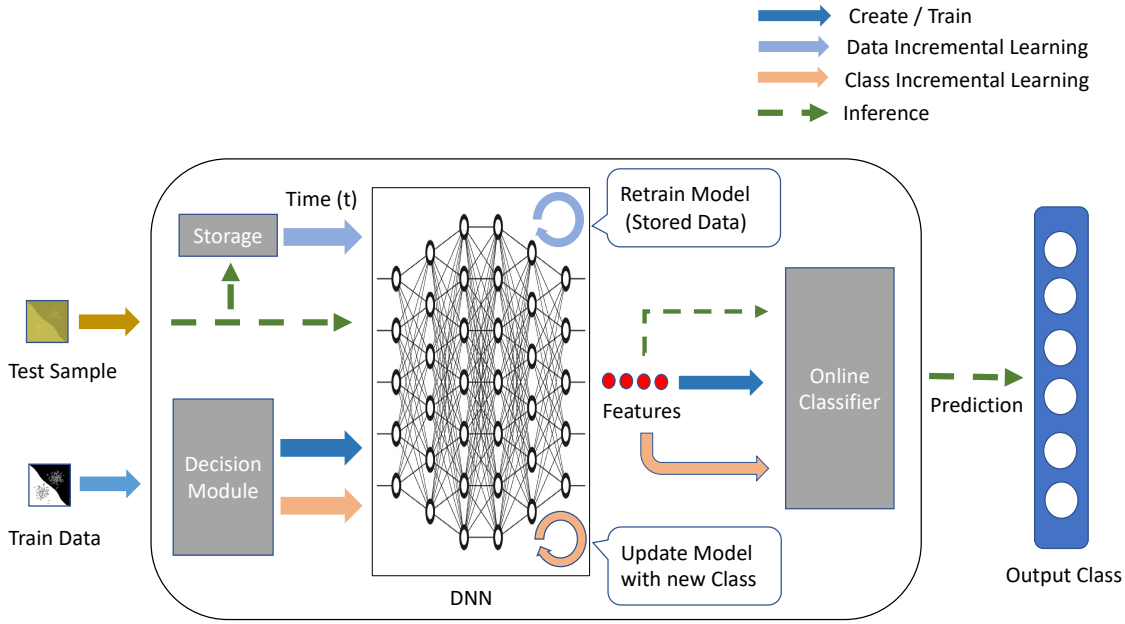


Fig. 2. Framework for Incremental Learning

3.2 Incremental Learning Algorithms

We used EWC and iCaRL as incremental learning algorithms. **EWC** is the most well known IL algorithm based on regularization which adds a penalty to regular loss function when learning a new task. In detail, EWC imposes a quadratic penalty on the difference between the weights for the old and the new tasks. More importantly, given two tasks, EWC attempts to protect the performance of the previous task while learning a new task by identifying the most important parameters for the previous task. Then, from a probabilistic perspective, the relative importance of the parameter θ regarding a data D of a given task can be modeled as the posterior

distribution $p(\theta|D)$. Assuming that we have two independent tasks A with D_A and B with D_B, the mentioned conditional probability is given by the Bayes' rule as follows:

$$logp(\theta|D) = logp(D_B|\theta) + logp(\theta|D_A) - logp(D_B) \tag{1}$$

Note that the posterior probability $logp(\theta|D_A)$ embeds all of the information about task A, i.e., the previous task and thus contains which parameters were important to task A, which is a key component to EWC. Yet, this term is intractable. Hence, EWC approximates the posterior as a Gaussian distribution with mean given by the parameters θ^*_A and a diagonal precision given by the diagonal of the Fisher information matrix F. Based on this approximation, the loss function L that EWC aims to minimize is given by

$$L(\theta) = L_B(\theta) + \lambda/2 \sum_{k=0} F_k(\Theta_k - \Theta^*_{A,k})^2 \tag{2}$$

When EWC is given more than two tasks, EWC try to keep the model parameters close to the learned parameters with respect to task A and B. In a more generic scenario where the number of tasks to be learned is i, the loss function of EWC can be modeled as follows:

$$L(\theta) = L_i(\theta) + \lambda/2 \sum_{j=0}^{i-1} F_j(\Theta_i - \Theta^*_j)^2 \tag{3}$$

where $L(\theta)$ is the total loss, θ is the network's parameters, $L_i(\theta)$ is the loss for the new task, and Θ^*_j are the important parameters of all previous tasks. λ is a hyperparameter that controls how much importance should be given to previous tasks compared to the new task. F is the Fisher matrix used to constrain the parameters important to previously learned tasks to stay close to their old values. Note that the task in our case means adding a new user.

iCaRL stands for Incremental Classifier and Representation Learning (iCaRL) and is the most widely used algorithm for IL which relies on storing data from previous tasks (i.e., exemplars) to do incremental learning. The classification is performed by using a nearest-class-mean (NCM) rule which relies on features extracted from the deep learning model, where the class means are calculated from the stored examples. When new tasks (classes) arrive, iCaRL creates a new training set combining the exemplars from all the previous tasks with the data samples of the new task. Next, for each example present in the exemplar set, the current model is evaluated, resulting in outputs for all previous tasks. Finally, the model parameters are updated by minimizing a loss function which encourages the model to output the correct class for the new task (classification loss) and to reproduce the scores stored in the previous step for the old tasks (distillation loss) using each data sample of the new task.

3.3 Online Learning Algorithms

We used SGD as an online learning algorithm. SGD is an iterative process that is used to find the values of the parameters of a function that can lead to the lowest cost for the function. Contrary to batch gradient descent, SGD calculates the cost function using a single example at each iteration instead of using all the examples. Interested readers can find more details about SGD in an article by Buttou et al. [3]. SGD is an ideal candidate for performing data incremental learning as it can learn from one data sample at a time and hence always remain updated when data distribution changes quickly without the need to update the computational heavy DNN.

3.4 Practical Scenario

We envisage that ContAuth can be useful in authentication systems where multiple users need to be added, and the underlying machine learning model has to be updated over time (not from scratch), which is quite common in smart homes, offices and where a central system coordinates the access. New users can register (mobile app,

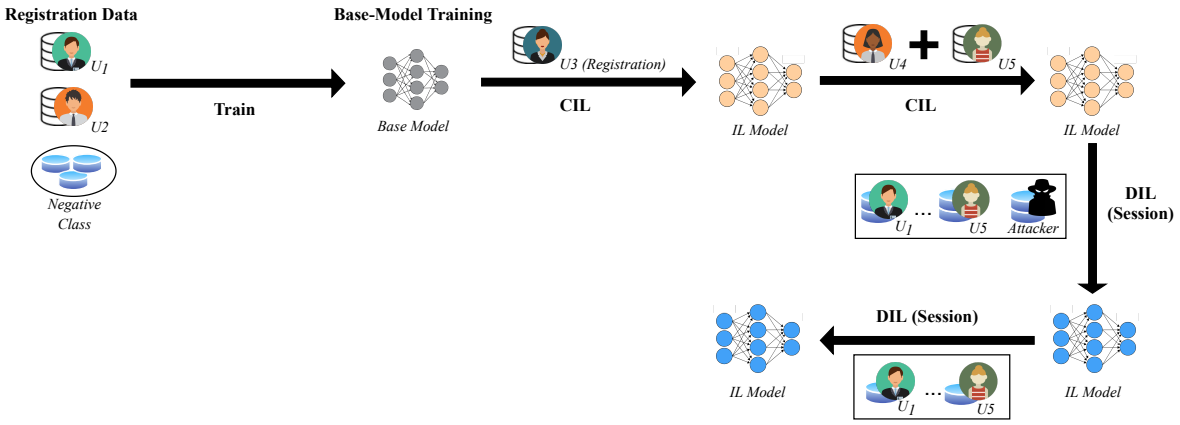


Fig. 3. Overview of ContAuth illustrating a practical, and real-world scenario.

web form or a GUI) to the ContAuth by providing their data, and the system assigns a designated id (label) to these users. The users can provide as much data as they want until they want to proceed to use the system. All registered users form positive classes in our case. To counter cases where a third party (an attacker) wants to access the system, we create an extra class (one negative class) to represent the attackers or the users who have not registered. This is akin to creating a "null" class in human activity recognition (HAR) applications where the null class represents all other activities which are not to be classified. Initially, such a class in our case can be created by using open-sourced data (see next section). After creating an initial model, ContAuth can incrementally add newly registered users (CIL) or adapt itself to the new incoming data (DIL). Over time we expect that the data for positive classes will continue to arrive as the existing users will use the system. The data for a negative class is expected to arrive when a real attacker tries to gain system access, which might happen infrequently. So, from time to time, attacker data can be retrieved from open-sourced large databases to simulate attack scenarios. Figure 3 illustrates an overview of a practical, real-world scenario where our system can be deployed.

4 EVALUATION METHODOLOGY

In this section, we discuss the evaluation methodology. We first describe the datasets we used in our study. Then we discuss our experimental setup. Finally, we provide details about the metrics and the device we used in our work.

4.1 Datasets

We evaluated ContAuth on six different datasets: breathing (sniff and deep breathing), longitudinal breathing (sniff and deep breathing), gait, and EMG. We now describe the preprocessing and the datasets in more detail.

4.1.1 Breathing. We used the breathing dataset [6] which showed the feasibility of using breathing acoustics (obtained through microphone with various gestures such as sniff and deep breathing) as a means to user authentication. The dataset has 70 breathing samples (audio files) for each breathing gesture collected from 10 participants in three different sessions. The time gap between each session was 3–4 days. 30 samples were collected in the first two sessions, and 10 samples were obtained in the last session. A sniff gesture consists of two quick consecutive inhalations. Whereas, a deep breathing gesture involves a long inhalation, followed by a long

exhalation. For our experiments, we pick first consecutive 45 samples for training (80%) and the next 5 samples for validation (20%) from each user. The next consecutive 10 samples (the last 10 samples from the second session) were used to assess security performance (we refer to it as *test 1*). The 10 samples from the last session used to measure security performance are referred to as *test 2*. As was shown in Figure 1, the performance between two sessions can change due to changes in data distributions of the user, and it reflects as a drop in accuracy between the two sessions. We ultimately want to prevent the degradation in the accuracy, as well as improve the accuracy by continuously adapting the models. To incorporate a negative class, we used data from five users chosen randomly from the longitudinal breathing dataset (10 users) for training and validation. The negative class represents an attacker class that contains data from many people to resemble a universal attacker class. We picked some random samples from these five users for our purpose to avoid a large data imbalance issue between the positive classes and one negative class. The data from other four users were used to launch the attacks during the testing session: *test 1*. We did not launch any attacks in the last session testing to simulate a practical scenario when there would be no attacks on the system. This setup of the negative class was followed for all the other datasets. Note that we did not utilize data from one user for this and the next dataset (Longitudinal breathing) as they were present in both the breathing datasets.

We created 10ms (sniffing) and 100ms (deep breathing) frames from each sample using Hamming window smoothing, and for each frame, we calculated 96 Gamma Frequency Cepstral Coefficients (GFCC) features; 32 GFCC, 32 delta GFCC, and 32 double delta GFCC. GFCC is based on signal power in a set of frequency bands. We used GFCC instead of MFCC as Chauhan et al. [6] showed that GFCC provided better accuracy than MFCC. The ability of GFCC to provide finer resolution at low frequencies than MFCC makes GFCC not only more robust in noisy environments [56], but also good at detecting low-frequency sounds which is the case for breathing as its dominant energy lies at lower frequencies. Next, we combine frames to create different windows of different sizes. Finally, we created overlapping windows for a given breathing sample. We used the window sizes of 30 and 25 (corresponding to a duration of 300 ms and 2500 ms), respectively, for the sniff and deep breathing gesture with an overlapping window of size one. Note that we used a small window size for deep breathing gesture than the one used by Chauhan et al. (250). This is because in our early experiments we found that training LSTM with very large window size (corresponds to the number of timestamps in LSTM network) such as 250 takes a few hours to train on Nvidia Jetson CPU which would be infeasible in real scenarios. We recommend using small time stamp sizes (50 or less) to enable on-device IL for LSTM. Nonetheless, the aforementioned approach is used to create windows for training data from breathing samples present in the training set. Similarly, windows for validation data and test data are created from breathing samples for the validation set and test sets, respectively. The created windows henceforth are used as the data for training, validation, and testing.

Table 1 shows the statistics for datasets employed in our study, including the training set, validation set, and different testing set sizes. We tried to use simpler architectures and smaller time steps to allow practical on-device incremental learning. There are three and four testing sessions for longitudinal breathing (sniff and deep) and EMG dataset, respectively. For all other datasets, there are two test sessions. They are referred to as Test # for a particular numbered session.

4.1.2 Longitudinal Breathing. We used an additional dataset to study the performance of ContAuth on longitudinal breathing data. This dataset has 10 users (6 males and 4 females) who performed breathing gestures (sniff and deep breathing) on their phones for four weeks on the microphone while doing different activities (walking, standing, sitting and after exercising) and in different contexts. In total, this dataset has 2355 sniffing and 2233 deep breathing samples. The preprocessing steps to create the windows remain the same as the breathing dataset except that the size of the overlapping windows was set to three. The datasets were divided into five parts: training, validation, *test 1*, *test 2*, and *test 3*. The data from the first week was used for training (90%) and validation (10%). The data from the subsequent weeks was used for testing sessions: second-week data for *test 1*, third-week

Table 1. Overview of the employed datasets. Task represents adding a class (new user).

Dataset	Dimension	#Train	# Valid	# Test 1	# Test 2	# Test 3	# Test 4	# Classes	Architecture	Tasks
Breathing - Sniff	30*96	8424	887	1782	1699	-	-	10	1/64	6
Breathing - Deep	25*96	9920	1142	2302	1660	-	-	10	1/128	6
Longitudinal Breathing - Sniff	30*96	192815	2317	13628	11076	8772	-	10	1/64	6
Longitudinal Breathing - Deep	25*96	12346	1278	9382	7005	6116	-	10	1/128	6
Gait	50*4	13683	2114	2115	9829	-	-	14	1/256	8
EMG	50*8	62222	7296	7996	42773	42246	40419	10	1/256	6

data for *test 2*, and fourth week's data for *test3*. For the negative class, we used data from five users chosen randomly from the breathing dataset (10 users, previous subsection) for training and validation. The data from other four users were used to launch the attacks during *test 1* and *test 2* sessions: two attackers in each session.

4.1.3 Gait. As an additional modality, we incorporated a gait dataset consisting of inertial data from the 3-axis accelerometer and the 3-axis gyroscope [14]. The gait dataset, IDNet [14], was collected based on Android smartphones from 50 volunteers. The data was recorded when the smartphone was worn in the participants' trousers' right front pocket. The data was acquired in such a way that a subject was asked to walk comfortably in variable conditions (e.g., different shoes and clothes) to mimic the real-world scenarios. Also, six devices were used for the data acquisition: Asus Zenfone 2, Samsung S3 Neo, Samsung S4, LG G2, LG G4, and a Google Nexus 5. We select 14 subjects who have atleast three sessions recorded in the dataset, similar to the breathing dataset. Then, analogous to the breathing datasets, we choose the first session and the first half of the second session in the gait dataset for training. For the second session's remaining data, the former half is used for validation, and the latter half is used for testing: *test 1*. After that, the third session is used for *test 2*. The rest of the 36 subjects were used as attackers. 26 out of 36 subjects were chosen randomly to train and validate the negative class in our system. We used the remaining 10 subjects to cast attacks during the testing session: *test 1*.

As our preprocessing steps, we downsample the gait data to 50Hz and normalize them to zero mean and unit variance. We adopt the same sliding window size of one second as in [14]. After that, to augment the gait dataset, we use a 50% overlap between two neighboring windows for all subjects, similar to [59]. For each data point at the timestamp of $i = 1, 2, \ldots$ the magnitude of the accelerometer data and gyroscope data is computed as follows:

$$mag(i) = \sqrt{x(i)^2 + y(i)^2 + z(i)^2} \tag{4}$$

Hence, the dimension of our resulting features is 50×4 for the accelerometer only data and 50×8 for the accelerometer and gyroscope combined data. We experimented with both the aforementioned datasets and identified that the accelerometer only data produces the best results. Thus, we adopt the accelerometer only data in further analysis throughout our experiments.

4.1.4 EMG. Since the surface electromyography (EMG) can also be used in the authentication task as in [11], we employed an EMG dataset from a Ninapro database which is widely used in research relying on EMG signals [1, 35]. Ninapro Database 6 (DB6) [35] was chosen in our experiments because it contains EMG signals of multiple sessions over the course of five days, thus providing different sensor modalities than microphones and IMU sensors present in breathing and gait datasets, respectively as well as allowing us to conduct a longitudinal

Table 2. Overview of the experimental setup with respect to different learning strategies. Note that in None, classes are added incrementally but without employing any IL method. Transfer Learning is similar to None except that classes are added incrementally by fine tuning. Baseline - Incremental and ConAuth uses IL methods such as iCaRL.

Setup	Data Incremental	Class Incremental	Online
Joint	x	x	x
Joint-Incremental	✓	x	x
None	x	✓	x
Transfer Learning	x	✓	x
Baseline - Incremental	✓	✓	x
ContAuth	✓	✓	✓

analysis. We present the detailed summaries of the dataset and how we use it in our experiments as follows. The Ninapro DB6 includes EMG data recordings from 10 subjects while performing seven gestures (e.g., Index finger extension, medium wrap, and writing tripod). Besides, each subject performs 12 repetitions per movement with four seconds rest between the repetitions. To construct the training dataset, we use recordings on day 1 and the first half of the data on day 2. For the remaining recordings on day 2, the first half of the remainder (25%) is used for the validation set, and the other (25%) is used for *test 1*. Thereafter, *test 2*, *test 3*, and *test 4* consists of all the recordings on day 3, 4, and 5 respectively. To get data for attackers, we used Ninapro Database 2 (10 users). It is similar to Database 6 except that it had only a single session. We used seven users to train and validate our negative class. The rest of the three users were used as attackers: one in each session: *test 1*, *test 2*, and *test 3*.

As our preprocessing steps, we downsample the sEMG data to 200 Hz and normalize them to zero mean and unit variance. We used a sliding window size of 250 ms with a 50% overlap similar to the one used in [54]. Note that we tested with the different sizes of the window (e.g., 200 ms, 250 ms, 500 ms, 1 s), and the trained model with a window size of 250 ms performs the best.

4.2 Experimental Strategy

The core DNN architecture of our framework used layered LSTM with 32, 64, 128, or 256 hidden units followed by a dropout layer, then a fully connected (FC) layer and a final output layer providing softmax based predictions. The configurations which provided us the best results are shown in Table 1. ADAM was used as an optimizer [24] with a default learning rate of 0.001 unless otherwise stated and mini-batch sizes of 32 (default). To prevent overfitting, we adopted two techniques: early stopping and dropouts. For dropouts, we used the values between 0.3 and 0.7 at steps of 0.1. Early stopping comes into play when the validation loss does not improve consecutively for 10 epochs and is not decreased by a certain threshold. We tried three different thresholds of 0.001, 0.0001, and 0.00001. We apply a weighted loss function [23] by estimating the inverse class distribution. So we can give more importance to the loss of a class with fewer data samples to tackle the data imbalance issue.

We tried a standard SGD classifier with a hinge loss function, l1 and l2 penalty (regularization), and optimal learning rate. We changed alpha between 0.0001 and 0.1 where alpha is a term that is multiplied with the regularization.

4.2.1 Experimental Setup. Here, we describe our experimental setup including various strategies (baselines). Table 2 shows an overview of the comparison.

Joint: It represents a scenario when the training data for all the classes is present from the beginning, and a model is trained as usual (naive, usual traditional deep learning case). This is a case which represents models that do not update themselves.

Joint-Incremental: It is similar to joint except that the deep learning model are updated (only data incremental) after every testing session.

ContAuth: In this setting, we assume that we have training data available for only half of the classes and the attacker class. We picked at least half of the users so that we have a reasonable amount of data to train our first DNN model. The rest of the classes are added incrementally one by one as it will happen in real-life scenarios. As stated earlier, we experimented with EWC and iCaRL which have their own parameters that can be tuned depending on the application. We treat each task as adding a new user to the system. In the case of EWC, we tried values of 1, 10, 100, 1000, 10000, 100000, 1000000 for parameter λ which controls how much importance is to be given to old tasks compared to the new task. For iCaRL we tried four different budget sizes: 1%, 5%, 10%, 20% and 40% of total training data. Recall that budget size determines how many samples need to be stored to perform class incremental learning. The initial task (first DL model) was trained in a usual manner as any DL model would be trained with N/2 users and an attacker class. Then the next tasks (adding one more user) were performed using the IL algorithm. A learning rate of 0.001 (default) was tried for all the tasks. At the end of each task, the features generated from the DL model for users newly added to the system are used to incrementally update the OL model. Note that after all the tasks are finished (all users added), the system can function as a data incremental learning setup to update according to new incoming data until a new class is encountered. As explained earlier, a new user has to register to use the system and provide their data to which the system provides an id (used as a label in the deep learning model).

As more data comes in (during test time once a model is deployed, *test 1* and subsequent test sessions), the OL model does the inference using features generated from the LSTM model and also updates itself with every single sample. At this time, the LSTM model is not updated. It is updated only when enough samples are collected (we assume after each testing session is finished). Updating the LSTM model here means training the last saved model to a few more epochs with stored data (from the last test session). The stored data is divided into 80% for training and 20% for validation. Overfitting in data incremental scenario was prevented by early stopping as explained earlier. The OL model keeps getting updated as more data arrives at the system during each testing session.

Baseline-Incremental: This case is similar to ContAuth except that the online model component is absent. Both CIL and DIL happen in this setup.

None: This represents the case when classes are added incrementally, but no incremental learning method is used. Catastrophic forgetting (CF) can have a considerable impact on such cases. This case is similar to ContAuth but without any IL method being used. DIL and online learning are also absent.

Transfer Learning: To compare our system's performance with similar methods, we tried the transfer learning approach (the most common method to design adaptive deep learning models). The experimental setup remains the same as that of ContAuth, but we used finetuning (a type of transfer learning) instead of the IL method. In this case, we try to add a new user by unfreezing the last FC layer (learning, backpropagating) with the new user's data while keeping the initial LSTM layers frozen (not learning). This is done to avoid overfitting the existing model by the new user's data. We repeat this procedure for every new user (class) after learning an initial model like ContAuth. DIL and online learning are absent as in None setup.

4.3 Metrics and Device

The following metrics were used to assess the performance of our IL framework: *True Positive Rate (TPR), False Positive Rate (FPR), False Negative Rate (FNR), True Negative Rate (TNR), storage and computation time or latency.* TPR is defined as the rate at which a target user is correctly identified as the true (correct, positive) user. FNR is the opposite of TPR, where a true user (positive class) is mistaken as someone else and could not access the system. FPR is the rate at which an attacker (negative class) can access the system maliciously. In comparison, TNR signifies when the system blocks the access of an attacker. In a reasonably accurate authentication system,

TPR should be very high with a very low FPR. We also discuss storage requirements for the DNN models, OL models, and storage associated with the IL algorithms. In terms of computation time, we show how much time it takes to do incremental learning in case of data and class settings for both IL and OL models. We implemented ContAuth on the Nvidia Jetson TX2 platform which consists of hex-core ARMv8 64-bit CPU, 8 GB RAM, and GPU (closely resembling high-end smartphones of current generation such as Google Pixel 4 and Samsung Galaxy S10).

To explain why IL method such as iCaRL works well, we also calculate other metrics used in the literature [5]. We consider how much an IL method forgets previous tasks and learns new tasks after it was trained from task 1 to k to assess the actual performance of IL methods by considering the following metrics.

- **Average Performance Measure (A)**: We denote the performance measure of a model on the j-th task ($j \leq k$) as $a_{k,j} \in [0, 1]$ after the model is trained from task 1 to k. The average performance measure at task k is defined as follows:

$$A_k = \frac{1}{k} \sum_{j=1}^{k} a_{k,j} \tag{5}$$

  The output space consists of $\cup_{j=1}^{k} \boldsymbol{y}^j$, and $a_{k,j}$ is based on a weighted F1-score in this work. Note that $a_{k,j}$ can be used to indicate an accuracy, proportion of correctly classified samples.
- **Forgetting Measure (F)**: The forgetting measure provides an estimate of how much a model forgets about the task given its present state. The forgetting for the j-th task after the model has been trained up to task $k > j$ can be quantified as:

$$f_j^k = \max_{l \in 1,\ldots,k-1} a_{l,j} - a_{k,j}, \quad \forall j < k \tag{6}$$

  The average forgetting at k-th task is denoted as $F_k = \frac{1}{k-1} \sum_{j=1}^{k-1} f_j^k$ by normalizing the number of tasks seen previously. The lower the F_k, the less forgetting on previous tasks.
- **Intransigence Measure (I)**: Intransigence is defined as the inability of a model to learn new tasks. To quantify the inability to learn, the joint model, often considered upper bound, which has access to all the datasets seen so far ($\cup_{l=1}^{k} D_l$) is compared and its performance is denoted as a_k^*. We then denote the intransigence for the k-th task as:

$$I_k = a_k^* - a_{k,k} \tag{7}$$

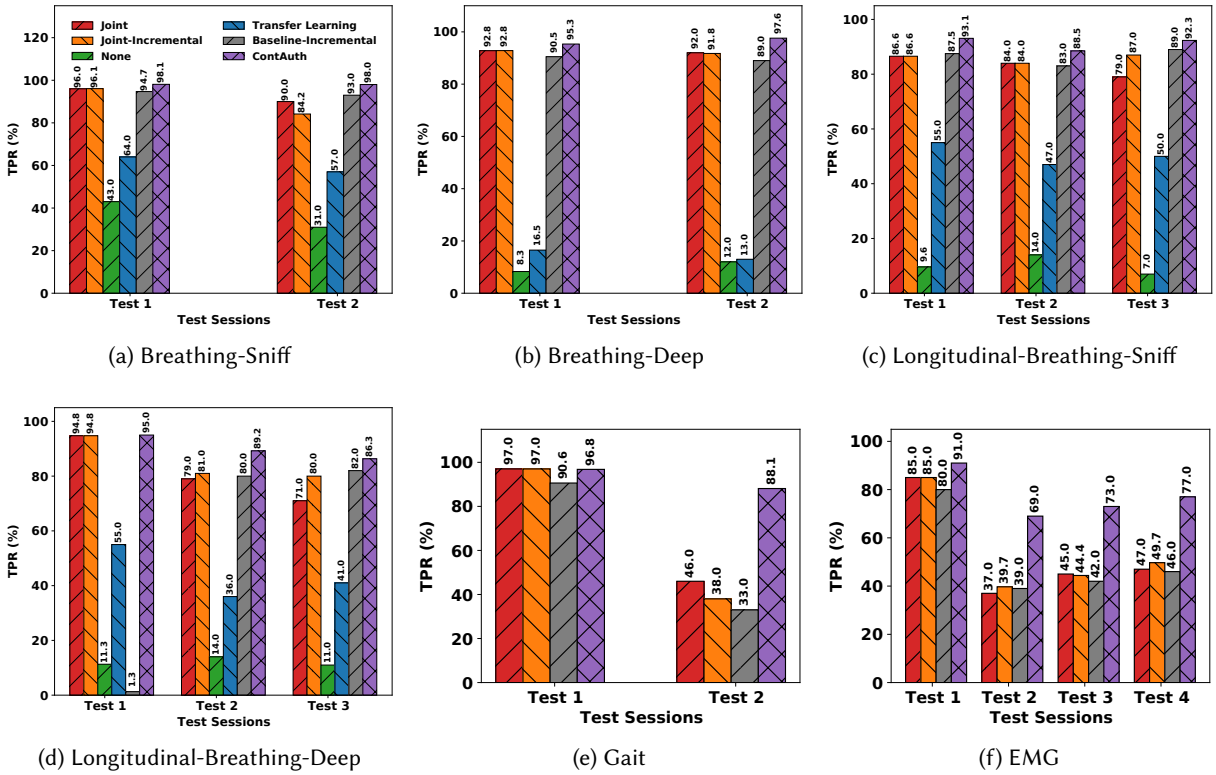
  where $a_{k,k}$ represents the performance of a model on the k-th task trained up to task k. Lower I_k implies that a model performs as close as a joint model or performs even better than the joint model when intransigence is negative ($I_k < 0$). Note that we use $a_{k,k}$ and I_k as the main performance indicators of a model since we are interested in the current performance of the model on all learned tasks from 1 to k.

5 RESULTS

This section presents the results for different settings. We report TPR, FPR, storage requirements and the execution time along with metrics explaining the good performance of iCaRL.

5.1 Accuracy

True Positive Rate (TPR) results are shown in Figure 4. Note that we show the best results for the different setups. Joint model always starts with a high TPR in the first session but then begins to degrade as the model is not updated. This degradation varies with datasets with the worst being observed for EMG (50%) and longitudinal deep breathing (15–20%). When the joint model is incrementally updated after each session (Joint-Incremental), we see a marginal increase in the performance except the gait modality. This can happen if the newly arrived data has a completely different data distribution than the one encountered by the existing model previously.



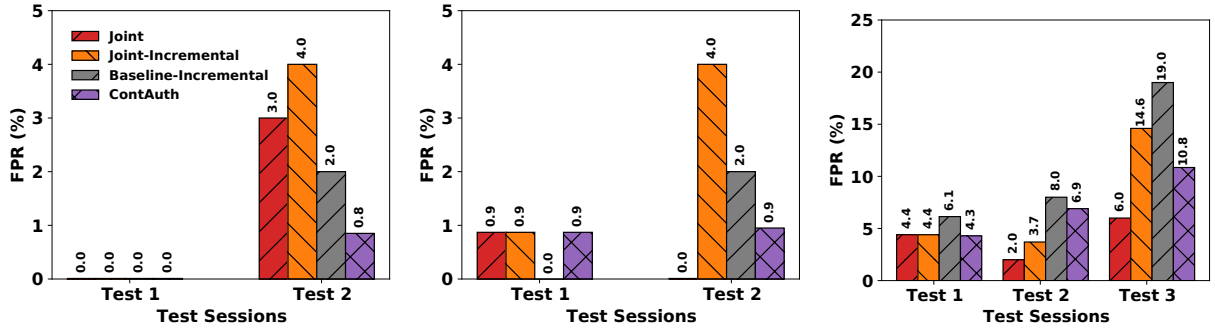
(a) Breathing-Sniff (b) Breathing-Deep (c) Longitudinal-Breathing-Sniff

(d) Longitudinal-Breathing-Deep (e) Gait (f) EMG

Fig. 4. True Positive Rate during different sessions for all the datasets.

The performance of Baseline-Incremental is similar to the Joint-Incremental. In both the setups, models are updated after each session. However, Baseline-Incremental involves learning class incrementally which is absent from the Joint-Incremental setup. Hence, the similar performance shows that the IL algorithm (iCaRL) present in Baseline-Incremental setup helps to alleviate the CF issue to a large extent. ContAuth further shows that IL combined with OL can achieve the best results: highest TPR (also means low false negative rates which translates to good usability) consistently across all sessions and modalities: 85% or more for all the modalities except EMG (>=70%). Nonetheless, ContAuth can improve between 2%-35% over the Joint model (traditional deep learning model) across datasets. This is also evident by looking at the gait dataset where ContAuth degrades by 7% and shows that it can handle the changing data distributions quite quickly. While, at the same time Joint model shows a drop of 50% in TPR. Even when compared to the Joint Incremental setup where deep learning models are updated (Joint–Incremental) with time [49], ContAuth achieves better performance due to the presence of OL model in the system.

We also highlight that in previously published research [8] on breathing based authentication, the *test 2* accuracy was around 75%. We have improved upon that by 15–20% and getting it closer to 99% using ContAuth. This increase is attributed to the use of GFCC as features and measures taken to prevent overfitting. The None and Transfer Learning setup provide the worst performance. This is expected as None does not employ any IL method, and TL simply cannot avoid the CF issue as it tends to forget the prior knowledge when learning new



(a) Longitudinal-Breathing-Sniff (b) Longitudinal-Breathing-Deep (c) EMG

Fig. 5. False Positive Rate during different sessions for datasets. Results for other datasets: Breathing, and gait are described in the text.

tasks. We executed TL for all the breathing datasets and found its performance to be abysmal and hence did not execute it for the rest of the datasets. We followed the same rationale for EWC: It helps alleviate the CF issue to a large extent but falls short of accuracy that might be acceptable for user authentication. We obtained near to 60% TPR for the breathing datasets with EWC.

False Positive Rate (FPR) results are depicted in Figure 5. The FPR obtained for the three datasets not shown in the Figure are: 7%, 0%, and 28% for Joint (and Joint-Incremental) setup; 8%, 0.3%, and 11% for Base-Incremental; 0.85%, 0.6%, 11% for ContAuth for breathing – sniff, breathing – deep, and gait dataset respectively. None setup always ended up in 100% FPR for all the datasets. In general, ContAuth achieves the best results (reasonably low FPR) across datasets. FPR for breathing-based modality is quite low (< 1%) for it to be deployed as a standalone modality in authentication systems. We observe that for the gait and EMG dataset the achieved FPR is a bit high (~10%) which might not be acceptable for applications requiring very strong security guarantees (e.g., financial transactions) but acceptable to applications where a very high degree of security is not required (e.g., enable personalized interactions in a smart home). Not to mention, the FPR we reported is similar to the reported FPRs in some of the existing literature [11, 15, 16, 32, 43]. We think that one of the reasons for FPR on EMG based authentication modality being a bit high in our case is because we combined all the seven gestures to be used as data, which has the potential to confuse the classifier and degrade the overall performance. The existing works on EMG utilize data from only a single or very few gestures for authentication [11, 43, 44]. Similarly, the existing works on gait-based authentication seem to benefit from using multiple gait cycles-based inference [52]. We aim to perform a study using aforementioned two approaches to improve the performance of ContAuth in future for EMG and gait based authentication: identifying a few gestures which can provide a very low FPR (EMG), and performing a majority voting based decision (based on multiple samples) at the inference time to further decrease the FPR. At the least, we believe that gait or EMG can be used as a secondary modality as in a multi-factor (modality) authentication system to provide extremely strong security. Further, ContAuth can also be extended to accommodate multi-modal authentication scenarios.

5.2 Storage

The model sizes required to be stored on the device to enable continual learning are shown in Table 3. The LSTM models only need to store the parameters (weights and biases) and are thus extremely light–weight. Depending on the number of hidden units, the LSTM model sizes varied between 500KB to 3.3MB. The higher the number

of hidden units, the larger the size of the model. The same pattern can be seen in the sizes of the SGD based online model. The size increases almost linearly (2x–3x times) with the feature size. Note that the OL model is constructed using features generated from the LSTM model which can be 64, 128 or 256 in length depending on the number of hidden units present in the LSTM model. The lower and upper bound of storage required for models to be stored thus lies in between 500–3400 KB, which is meager compared to the amount of storage on modern devices. The extra temporary storage needed to store the samples for updating DNN after each testing session depends on the number of testing data samples present in each testing session. In our case, it ranges from 4 MB to 316 MB for all the datasets. This storage is flexible, depending on the device's storage capacity and other requirements as to when to update the DNN model and how often. The devices with a large storage capacity can store more samples, and hence DNN models can be updated less frequently, resulting in lower energy consumption. Otherwise, DNN models need to be more frequently updated on devices with a smaller storage capacity, which can cause faster battery depletion. However, one can always process samples to check if it needs to be stored or not. For example, if the sample is highly similar to the already stored samples, then it can be discarded.

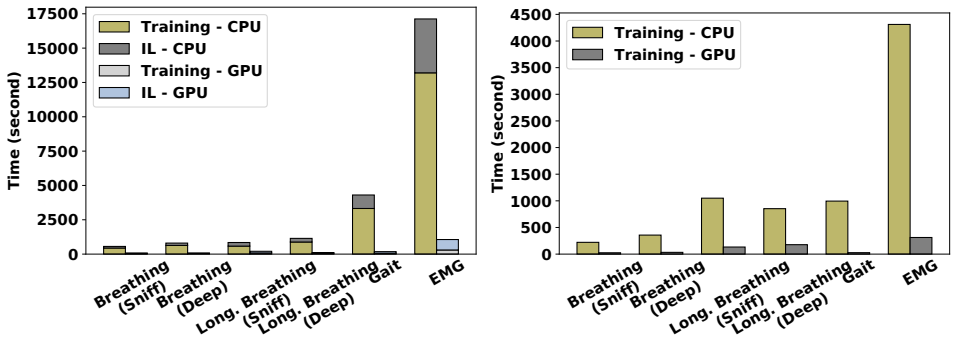
Table 3. Model Size (KB) for DNN and Online Model

Hidden Units and Feature Size	LSTM	SGD
Breathing - 64	508	6.31
Breathing - 128	1400	11.43
Gait - 256	3300	24
EMG - 256	3300	24

Table 4. Extra Storage (MB) for ContAuth. Results in bold are the budgets where ContAuth obtained best performance. Bold entries inside () are the dropout rates.

Dataset – Hidden Units	Budget -> 1%	5%	10%	20%	40%
Breathing (Sniff) – 64	2.5	12.5	**25 (0.3)**	50	100
Breathing (Deep) – 128	2.45	**12.25 (0.5)**	24.5	49	98
Longitudinal Breathing (Sniff) – 64	4.7	23.5	47	**94 (0.6)**	188
Longitudinal Breathing (Deep) – 128	2.58	12.9	**25.8 (0.3)**	51.6	103.2
Gait – 256	0.25	1.25	2.5	5	**10 (0.5)**
EMG - 256	2.2	11	22	44	**88 (0.4)**

Every IL algorithm requires extra storage in order to function. EWC stores fisher matrices and means for each task. The extra storage required is proportional to $2 * \text{number of tasks} * \text{model size}$. For iCaRL, the budget size defines how much storage is needed. 1% of the budget means that 1% of the size of the total training samples would be required on the device. Hence, EWC requires less storage than iCaRL but does not match its accuracy. The storage needed on a device for ContAuth using iCaRL is shown in Table 4. We realize that a higher budget does not necessarily translate into better performance. Mostly, smaller budget sizes (10%, 20%) obtains better results than higher budget sizes. This is an important result as it shows that the extra storage needed for ContAuth to perform well on the device can be quite low: in our case, it ranges between 10 MB - 94 MB.



(a) CIL (b) DIL

Fig. 6. Latency

5.3 Latency

We present latency results for class incremental learning in Figure 6a for different datasets. The results show the average total time in seconds to perform the entire processing which consists of training and incremental learning on CPU and GPU. We observe that IL takes less time than training (two–65 minutes per task) on CPU which is roughly 20%-30% of the whole processing time. Most of the total time, which varies between 7–220 minutes, is spent on training. However, the results on GPU narrates a different story. The total processing time only takes 80–1060 seconds and thus provides 5–20x times acceleration over CPU. This acceleration helps the training to the extent that it becomes at par with the IL times. In some cases, the IL times can be even a bit higher than training time if the iCaRL budget is large (20–40%). The primary reason is that operations involved in training are optimized for GPU while operations involved in iCaRL (construct, reduce, etc.) are not.

SGD does incrementally update (online learning) in an extremely fast manner. On average, the time to update an SGD model takes 0.0045, 0.005, and 0.0055 seconds corresponding to feature lengths of 64, 128, 256, respectively. The latencies for data incremental learning is presented in Figure 6b. Similar to CIL results, GPU accelerates the training process such that even the datasets with more data (longitudinal sniff) and slightly bigger architectures (gait) are executed in roughly five minutes. Latencies on CPU are higher ranging from four to 17 minutes depending on the complexity of the data and LSTM architecture except for EMG (70 minutes). Nonetheless, our results are promising and show that IL is practically possible on the device, especially with the help of the GPU. The results on CPU are also encouraging, showing that it is feasible to execute an end to end IL process (ContAuth) on a very large dataset such as EMG (on average 50K data points across sessions) without breaking the device. As mentioned earlier, the EMG dataset can be reduced in size by only focusing on highly accurate (few) gestures instead of all seven gestures to allow faster execution latency. We discuss more on how to expedite the IL process in the Discussion section.

5.4 Performance on IL Specific Metrics

In this subsection, we further investigate the effectiveness of the IL method, iCaRL, by comparing it to the transfer learning approach. In order to make a fair comparison of our studied methods, we use various IL specific metrics such as average performance measure, forgetting measure, and intransigence measure.

Table 5 shows the results in a summarized way for ContAuth employing iCaRL and Transfer Learning method. Note that we only show results for iCaRL as it performs better than EWC amongst the IL methods we tried.

Table 5. Performance on IL specific metrics.

Dataset – Hidden Units	ContAuth				Transfer Learning			
	A_k	F_k	$a_{k,k}$	I_k	A_k	F_k	$a_{k,k}$	I_k
Breathing (Sniff) – 64	0.93	0.02	0.94	0.02	0.80	0.01	0.61	0.36
Breathing (Deep) – 128	0.94	0.00	0.94	0.00	0.61	0.03	0.26	0.68
Longitudinal Breathing (Sniff) – 64	0.91	0.02	0.89	0.00	0.65	0.01	0.43	0.46
Longitudinal Breathing (Deep) – 128	0.95	0.01	0.94	0.01	0.70	-0.01	0.35	0.59
Gait – 256	0.97	-0.01	0.90	0.05	-	-	-	-
EMG – 256	0.85	0.004	0.83	0.06	-	-	-	-

A_k measure indicates average accuracy on all tasks, and $a_{k,k}$ measure denotes the accuracy after learning all tasks. Besides, F_k allows us to estimate how much an IL method retains old knowledge of previous tasks. Also, I_k means how good an IL method is in learning new tasks. Note that for A_k and $a_{k,k}$ the higher the values, the better the model is. In contrast, for F_k and I_k, a low value represents a better model. For instance, low F_k and I_k mean that the model forgets previous knowledge less and better learns new knowledge, respectively. As shown in Table 5, iCaRL performs excellently on all IL specific metrics on various modalities of the authentication task. In other words, iCaRL is good at learning new classes/tasks (low intransigence, I) while maintaining high performance (high accuracy, A) as well as retaining old knowledge (low forgetting measure, F) and is an excellent component being utilized in ContAuth. Conversely, the same is not the case for Transfer Learning. Although Transfer Learning allows previous knowledge to be retained (low F value), it cannot learn new tasks as indicated by a high value of intransigence (I). Thus, it shows a low performance in general as indicated by average accuracy (A, $a_{k,k}$).

6 DISCUSSION AND CONCLUSIONS

The performance of the current authentication systems needs to be robust and reliable. However, this requires the system to adapt continuously on the fly without impacting the resources on the device, excessively. In this work, we propose *ContAuth*, a system that can add new users without training the underlying machine learning models from scratch and also adapt continuously to the new incoming data. The system does so incrementally and thus always remains updated. To add new users, ContAuth employs class incremental learning methods. It combines deep learning models with online learning models for continuous updating. We evaluated ContAuth on multiple behavioral-based modalities. Our results show that ContAuth can provide good accuracy for different modalities of authentication. Finally, we also showed that ContAuth could be deployed practically on the device, further allowing data privacy.

Like all other systems, ContAuth has some limitations. As of now, ContAuth cannot handle incorrect inputs. For example, someone trying to use voice (sounds such as 'oh', 'ahh', etc.) when a breathing gesture (sniff or deep breathing) is required to use the system. This issue can be handled by creating an entry-level component that can recognize and filter unwanted inputs (sounds as examples) and ask a user to use the correct input. This is usually accomplished by training a binary machine learning classifier on required against all other inputs [9]. The other issue can be that an IL method employed inside ContAuth (iCaRL) might fail to perform well after adding a certain number of classes. This can be considered as equivalent to finding an upper bound on the number of classes that can be added to any IL method without degrading the performance by a considerable margin. We tried to find this upper bound for ContAuth by performing class incremental learning on WhuGait dataset [59] (118 users). We initially trained a model with 10 classes and then kept adding classes (one by one) until the

performance degrades by more than 5% (threshold we used in the paper already). After adding all the 118 users, we still found the performance acceptable in our case (90% in case of joint model vs. 85% with iCaRL on an LSTM layer with 256 hidden units). Note that this is one of the biggest open-source dataset available for the modalities we tried in our work. Also, note that we cannot use this dataset in our earlier analysis since it has only a single session. Nevertheless, we aim to find the breaking point for class incremental learning in the future as datasets with a very high number of users become available. We tried to expedite the IL process by using smaller DNN architectures, early stopping, GPU based acceleration and fewer time steps in LSTM models. However, the IL process can be further expedited on the device, which we discuss next.

Overall, our analysis shows that ContAuth can provide higher accuracy across sessions than other baseline methods for breathing, gait and EMG based behavior authentication and can be executed on device. The most striking observation from our evaluation is that the majority of the IL execution process was dominated by training and seems to the major bottleneck to perform IL on device, especially on the CPU. We believe that there are optimization techniques that might help alleviate the high training latency. The first is to use only FP16 or FP8 formats instead of mixed precision when training DNNs [51]. It will be interesting to see the trade-offs between accuracy and latency when using such techniques. Golub et al. [17] recently proposed a technique to train DNNs faster. Their technique restricts the total number of weights updated during backpropagation to those with the highest total gradients. For the remaining weights, their initial value is regenerated at every access to avoid storing them in memory. This leads to a reduction in the number of memory accesses during training and hence accelerates the training process. Although not a standard technique, like mixed precision training or 16-bit training, it can be utilized to enable faster incremental learning. We leave this as future work.

GPU accelerates the IL and specifically the training process to allow running ContAuth on the device practically. New devices such as Nvidia Xavier, Nvidia Xavier NX, and even some smartphones such as Google Pixel 4 are now shipping while fitted with Tensor cores or Neural Processing units (NPU) specialized in enabling fast neural network processing. Although for now NPUs only support faster inference (through existing APIs), it may be worth exploring how these can be used to expedite training DNN models on the device. We expect that with the advent of NPUs and better optimization techniques, IL can be applied to more complex workloads (e.g., continuous surveillance and human activity recognition) in the future.

ACKNOWLEDGMENTS

This work was supported by ERC Project 833296 (EAR) and a Google Faculty Award 2019.

REFERENCES

[1] Manfredo Atzori, Arjan Gijsberts, Claudio Castellini, Barbara Caputo, Anne-Gabrielle Mittaz Hager, Simone Elsig, Giorgio Giatsidis, Franco Bassetto, and Henning Müller. 2014. Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Scientific Data* 1 (Dec. 2014), 140053. https://doi.org/10.1038/sdata.2014.53

[2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning* 79, 1 (May 2010), 151–175. https://doi.org/10.1007/s10994-009-5152-4

[3] Léon Bottou. 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes* 91, 8 (1991), 12.

[4] Kyong Chang, Kevin W Bowyer, Sudeep Sarkar, and Barnabas Victor. 2003. Comparison and combination of ear and face images in appearance-based biometrics. *IEEE Transactions on pattern analysis and machine intelligence* 25, 9 (2003), 1160–1165.

[5] Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. 2018. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. 532–547. http://openaccess.thecvf.com/content_ECCV_2018/html/Arslan_Chaudhry__Riemannian_Walk_ECCV_2018_paper.html

[6] Jagmohan Chauhan, Yining Hu, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2017. BreathPrint: Breathing acoustics-based user authentication. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 278–291.

[7] Jagmohan Chauhan, Jathushan Rajasegaran, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2018. Performance characterization of deep learning models for breathing-based authentication on resource-constrained devices. *Proceedings of the ACM on*

Interactive, Mobile, Wearable and Ubiquitous Technologies 2, 4 (2018), 1–24.

[8] Jagmohan Chauhan, Suranga Seneviratne, Yining Hu, Archan Misra, Aruna Seneviratne, and Youngki Lee. 2018. Breathing-Based Authentication on Resource-Constrained IoT Devices using Recurrent Neural Networks. *Computer* 51, 5 (2018), 60–67.

[9] Jagmohan Chauhan, Benjamin Zi Hao Zhao, Hassan Jameel Asghar, Jonathan Chan, and Mohamed Ali Kaafar. 2017. BehavioCog: An Observation Resistant Authentication Scheme. In *International Conference on Financial Cryptography and Data Security*. Springer, 39–58.

[10] Mohammad Omar Derawi, Claudia Nickel, Patrick Bours, and Christoph Busch. 2010. Unobtrusive user-authentication on mobile phones using biometric gait recognition. In *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. IEEE, 306–311.

[11] Boyu Fan, Xuefeng Liu, Xiang Su, Pan Hui, and Jianwei Niu. 2020. Emgauth: An emg-based smartphone unlocking system using siamese network. In *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 1–10.

[12] Faisal Farooq, Ruud M Bolle, Tsai-Yang Jea, and Nalini Ratha. 2007. Anonymous and revocable fingerprint recognition. In *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 1–7.

[13] Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. 2012. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE transactions on information forensics and security* 8, 1 (2012), 136–148.

[14] Matteo Gadaleta and Michele Rossi. 2018. IDNet: Smartphone-based gait recognition with convolutional neural networks. *Pattern Recognition* 74 (Feb. 2018), 25–37. https://doi.org/10.1016/j.patcog.2017.09.005

[15] Davrondzhon Gafurov, Kirsi Helkala, and Torkjel Søndrol. 2006. Biometric Gait Authentication Using Accelerometer Sensor. *JCP* 1, 7 (2006), 51–59.

[16] Davrondzhon Gafurov, Einar Snekkenes, and Patrick Bours. 2007. Spoof attacks on gait authentication system. *IEEE Transactions on Information Forensics and Security* 2, 3 (2007), 491–502.

[17] Lemieux Guy Golub, Maximilian and Mieszko Lis. 2019. Full deep neural network training on a pruned weight budget. In *SysML*.

[18] Abdenour Hadid, Nicholas Evans, Sebastien Marcel, and Julian Fierrez. 2015. Biometrics systems under spoofing attack: an evaluation methodology and lessons learned. *IEEE Signal Processing Magazine* 32, 5 (2015), 20–30.

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

[20] HM Hossain, MD Al Haiz Khan, and Nirmalya Roy. 2018. DeActive: scaling activity recognition with active deep learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 2 (2018), 66.

[21] Chenyu Huang, Huangxun Chen, Lin Yang, and Qian Zhang. 2018. BreathLive: Liveness detection for heart sound authentication with deep breathing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–25.

[22] Md Abdullah Al Hafiz Khan, Nirmalya Roy, and Archan Misra. 2018. Scaling human activity recognition via deep learning-based domain adaptation. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 1–9.

[23] Gary King and Langche Zeng. 2001. Logistic Regression in Rare Events Data. *Political Analysis* 9, 2 (2001), 137–163. https://doi.org/10.1093/oxfordjournals.pan.a004868

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

[25] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* 114, 13 (2017), 3521–3526.

[26] Lingjun Li, Xinxin Zhao, and Guoliang Xue. 2013. Unobservable re-authentication for smartphones.. In *NDSS*, Vol. 56. 57–59.

[27] Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence* 40, 12 (2017), 2935–2947.

[28] Jian Liu, Yingying Chen, Yudi Dong, Yan Wang, Tiannming Zhao, and Yu-Dong Yao. 2020. Continuous user verification via respiratory biometrics. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 1–10.

[29] Rui Liu, Cory Cornelius, Reza Rawassizadeh, Ronald Peterson, and David Kotz. 2018. Vocal resonance: Using internal body voice for wearable authentication. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 19.

[30] David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*. 6467–6476.

[31] Li Ma, Tieniu Tan, Yunhong Wang, and Dexin Zhang. 2004. Efficient iris recognition by characterizing key local variations. *IEEE Transactions on Image processing* 13, 6 (2004), 739–750.

[32] Jani Mantyjarvi, Mikko Lindholm, Elena Vildjiounaite, S-M Makela, and HA Ailisto. 2005. Identifying users of portable devices from gait pattern with accelerometers. In *Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, Vol. 2. IEEE, ii–973.

[33] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. 1995. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review* 102, 3 (1995), 419.

[34] Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*. Vol. 24. Elsevier, 109–165.

[35] Francesca Palermo, Matteo Cognolato, Arjan Gijsberts, Henning Müller, Barbara Caputo, and Manfredo Atzori. 2017. Repeatability of grasp recognition for robotic hand prosthesis control based on sEMG data. In *2017 International Conference on Rehabilitation Robotics (ICORR)*. 1154–1159. https://doi.org/10.1109/ICORR.2017.8009405 ISSN: 1945-7901.

[36] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (Oct. 2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

[37] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks* (2019).

[38] Liangying Peng, Ling Chen, Zhenan Ye, and Yi Zhang. 2018. AROMA: A Deep Multi-Task Learning Based Simple and Complex Human Activity Recognition Method Using Wearable Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 2 (2018), 74.

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2001–2010.

[40] Douglas Reynolds, Walter Andrews, Joseph Campbell, Jiri Navratil, Barbara Peskin, Andre Adami, Qin Jin, David Klusacek, Joy Abramson, Radu Mihaescu, et al. 2003. The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, Vol. 4. IEEE, IV–784.

[41] Florian Schaub, Ruben Deyhle, and Michael Weber. 2012. Password entry usability and shoulder surfing susceptibility on different smartphone platforms. In *Proceedings of the 11th international conference on mobile and ubiquitous multimedia*. ACM, 13.

[42] Cong Shi, Jian Liu, Hongbo Liu, and Yingying Chen. 2017. Smart user authentication through actuation of daily activities leveraging WiFi-enabled IoT. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 5.

[43] Siho Shin, Jaehyo Jung, and Youn Tae Kim. 2017. A study of an EMG-based authentication algorithm using an artificial neural network. In *2017 IEEE SENSORS*. IEEE, 1–3.

[44] Ryohei Shioji, Shin-ichi Ito, Momoyo Ito, and Minoru Fukumi. 2018. Personal authentication and hand motion recognition based on wrist EMG analysis by a convolutional neural network. In *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*. IEEE, 184–188.

[45] Joo Yong Sim, Hyung Wook Noh, Woonhoe Goo, Namkeun Kim, Seung-Hoon Chae, and Chang-Geun Ahn. 2019. Identity Recognition Based on Bioacoustics of Human Body. *IEEE Transactions on Cybernetics* (2019).

[46] Chen Song, Aosen Wang, Kui Ren, and Wenyao Xu. 2016. Eyeveri: A secure and usable approach for smartphone user authentication. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 1–9.

[47] Xu Sun, Hisashi Kashima, and Naonori Ueda. 2012. Large-scale personalized human activity recognition using online multitask learning. *IEEE Transactions on Knowledge and Data Engineering* 25, 11 (2012), 2551–2563.

[48] Furkan Tari, Ant Ozok, and Stephen H Holden. 2006. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *Proceedings of the second symposium on Usable privacy and security*. ACM, 56–66.

[49] Cong Wang, Yanru Xiao, Xing Gao, Li Li, and Jun Wang. 2019. Close the gap between deep learning and mobile intelligence by incorporating training in the loop. In *Proceedings of the 27th ACM International Conference on Multimedia*. 1419–1427.

[50] Lei Wang, Kang Huang, Ke Sun, Wei Wang, Chen Tian, Lei Xie, and Qing Gu. 2018. Unlock with Your Heart: Heartbeat-based Authentication on Commercial Mobile Phones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 140.

[51] Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen, and Kailash Gopalakrishnan. 2018. Training deep neural networks with 8-bit floating point numbers. In *Advances in neural information processing systems*. 7675–7684.

[52] Weitao Xu, Yiran Shen, Yongtuo Zhang, Neil Bergmann, and Wen Hu. 2017. Gait-watch: A context-aware authentication system for smart watch based on gait recognition. In *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*. 59–70.

[53] Jian Yang, David Zhang, Jing-yu Yang, and Ben Niu. 2007. Globally maximizing, locally minimizing: unsupervised discriminant projection with applications to face and palm biometrics. *IEEE transactions on pattern analysis and machine intelligence* 29, 4 (2007), 650–664.

[54] Xiaolong Zhai, Beth Jelfs, Rosa H. M. Chan, and Chung Tin. 2017. Self-Recalibrating Surface EMG Pattern Recognition for Neuroprosthesis Control Based on Convolutional Neural Network. *Frontiers in Neuroscience* 11 (2017). https://doi.org/10.3389/fnins.2017.00379

[55] Linghan Zhang, Sheng Tan, Jie Yang, and Yingying Chen. 2016. Voicelive: A phoneme localization based liveness detection for voice authentication on smartphones. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1080–1091.

[56] Xiaojia Zhao and DeLiang Wang. 2013. Analyzing noise robustness of MFCC and GFCC features in speaker identification. In *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 7204–7208.

[57] Nan Zheng, Kun Bai, Hai Huang, and Haining Wang. 2014. You are how you touch: User verification on smartphones via tapping behaviors. In *2014 IEEE 22nd International Conference on Network Protocols*. IEEE, 221–232.

[58] Bing Zhou, Jay Lohokare, Ruipeng Gao, and Fan Ye. 2018. EchoPrint: Two-factor authentication using acoustics and vision on smartphones. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 321–336.

[59] Qin Zou, Yanling Wang, Qian Wang, Yi Zhao, and Qingquan Li. 2020. Deep Learning-Based Gait Recognition Using Smartphones in the Wild. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3197–3212. https://doi.org/10.1109/TIFS.2020.2985628 Conference Name: IEEE Transactions on Information Forensics and Security.