My Health Sensor, My Classifier – Adapting a Trained Classifier to Unlabeled End-User Data

KONSTANTINOS NIKOLAIDIS, STEIN KRISTIANSEN, THOMAS PLAGEMANN, VERA GOEBEL, and KNUT LIESTØL, Department of Informatics, University of Oslo
MOHAN KANKANHALLI, Department of Computer Science, National University of Singapore, Singapore
GUNN-MARIT TRAAEN, Oslo University Hospital, Rikshospitalet, University of Oslo
BRITT ØVERLAND, Lovisenberg Diakonale Hospital
HARRIET AKRE, Oslo University Hospital, Rikshospitalet, University of Oslo
LARS AAKERØY and SIGURD STEINSHAMN, St. Olavs University Hospital, Norwegian University of Science and Technology

Sleep apnea is a common yet severely under-diagnosed sleep related disorder. Unattended sleep monitoring at home with low-cost sensors can be leveraged for condition detection, and Machine Learning offers a generalized solution for this task. However, patient characteristics, lack of sufficient training data, and other factors can imply a domain shift between training and end-user data and reduced task performance. In this work, we address this issue with the aim to achieve personalization based on the patient's needs. We present an **unsupervised domain adaptation (UDA)** solution with the constraint that labeled source data are not directly available. Instead, a classifier trained on the source data is provided. Our solution iteratively labels target data sub-regions based on classifier beliefs, and trains new classifiers from the expanding dataset. Experiments with sleep monitoring datasets and various sensors show that our solution outperforms the classifier trained on the source domain, with a kappa coefficient improvement from 0.012 to 0.242. Additionally, we apply our solution to digit classification DA between three well-established datasets, to investigate its generalizability, and allow for related work comparisons. Even without direct access to the source data, it outperforms several well-established UDA methods in these datasets.

CCS Concepts: • **Computing methodologies** → **Learning paradigms**; *Neural Networks*; • **Applied computing** → **Health informatics;**

Additional Key Words and Phrases: Sleep apnea, Machine learning, neural networks, domain-adaptation, sensors, personalization

This work was performed as part of the CESAR project (nr. 250239/O70) funded by The Research Council of Norway.
Authors' addresses: K. Nikolaidis, S. Kristiansen, T. Plagemann, V. Goebel, and K. Liestøl, Department of Informatics, University of Oslo Gaustadalléen 23B, 0373 Oslo, Norway; emails: {konstan, steikr, plageman, goebel, knut}@ifi.uio.no; M. Kankanhalli, National University of Singapore, School of Computing, COM1, 13, Computing Dr, Singapore 117417; email: mohan@comp.nus.edu.sg; G.-M. Traaen and H. Akre, Oslo University Hospital, Rikshospitalet, Sognsvannsveien 20, 0372 Oslo, Norway; emails: {gtraaen, haakre}@ous-hf.no; B. Øverland, Lovisenberg Diakonale Hospital, Lovisenberggata 17, 0456 Oslo, Norway; email: britt.overland@lds.no; L. Aakerøy and S. Steinshamn, St. Olavs University Hospital, Norwegian University of Science and Technology, Prinsesse Kristinas gate 3, 7030 Trondheim, Norway; emails: lars.aakeroy@stolav.no, sigurd.steinshamn@ntnu.no.

2637-8051/2022/10-ART48
https://doi.org/10.1145/3559767

ACM Reference format:
Konstantinos Nikolaidis, Stein Kristiansen, Thomas Plagemann, Vera Goebel, Knut Liestøl, Mohan Kankanhalli, Gunn-Marit Traaen, Britt Øverland, Harriet Akre, Lars Aakerøy, and Sigurd Steinshamn. 2022. My Health Sensor, My Classifier – Adapting a Trained Classifier to Unlabeled End-User Data. *ACM Trans. Comput. Healthcare* 3, 4, Article 48 (October 2022), 24 pages.
https://doi.org/10.1145/3559767

1 INTRODUCTION

Sleep apnea is a common yet severely under-diagnosed sleep related breathing disorder. The overall goal of our work is to enable "anybody" to use low-cost pre-screening solutions at home in which consumer electronics, like smart phones and smart watches, are used for data acquisition and Machine Learning for data analysis. We use as foundation for our research data from a large clinical study, called A3 study, at the Oslo University Hospital and St. Olavs University Hospital. In this study, sleep monitoring data from several hundred patients is collected and analyzed. A portable sleep monitor certified for clinical diagnosis (i.e., Nox T3 [1]) has been used for data acquisition. Currently, we achieve with a **convolutional neural network (CNN)** trained on this data a classification accuracy of approximately 80% [2]. However, we cannot guarantee that this model can generalize well to new data from an end-user, because of potential domain shifts. Such domain shifts can for example be caused by characteristics of individual end-user sleep data that are not represented in the A3 dataset and quality issues. The latter is based on the fact that a sleep monitor that is certified for clinical diagnosis produces data with substantially higher quality than that of data collected by end-users at home with consumer electronics. Furthermore, personalization can improve classification in many cases, because individuals are different in terms of physiology, prevalence, sensor placement and the same signals might be measured with different sensors, for example different smartwatch brands.

Domain Adaptation (DA) has recently been successfully applied to address the problem of domain shift. DA aims to improve learning for a predictive task at a target, assuming a source and a target domain for which the distributions of source and target differ [3, 4]. The predictive tasks are the same across the two domains. A sub-field of DA is unsupervised DA, for which labels are provided only for the source data [5], but not for the target data. The vast majority of unsupervised DA literature focuses on non-medical, easy to label, visual open data, without specific usage constraints.

Existing DA solutions cannot be applied to create a classifier that is adapted to the end-user data, because (1) end-users might not want to give us their data for privacy reasons and we do not have the resources to support many end-users, (2) privacy regulations do not permit to share the A3 dataset (e.g., with end-users or third parties), and (3) source and target data are processed together in the majority of existing DA solutions.

This scenario can be generalized. Assume a host which has trained a model with labeled data for classification, but the model cannot be directly used on data collected by an end-user due to the presence of domain shift. Both entities, i.e., host and end-user, do not want (or cannot) share their data with the other entity, as it is very often the case in the medical domain. Thus, the only way to create a personalized classifier for the end-user is DA without direct access to the labeled source data (See Figure 1 for an example of this case, which we investigate in this work, in comparison to a baseline case). Furthermore, it must be considered that the end-user potentially has less computing resources than the host (e.g., lack of dedicated hardware components, lack of sufficient GPU memory to load the data, etc.). As a result even if the host is willing to share her data, performing joint training on the end-user could be problematic.

To address these issues, we introduce a method called ***Step-wise Increasing target dataset Coverage based on high confidence*** **(***SICO***)** to efficiently perform DA at the end-user. *SICO* performs unsupervised DA with the use of only a classifier trained on the source domain and unlabeled data from the target domain. One of the fundamental ideas in this work is to release only the classifier trained on the source domain, since extracting

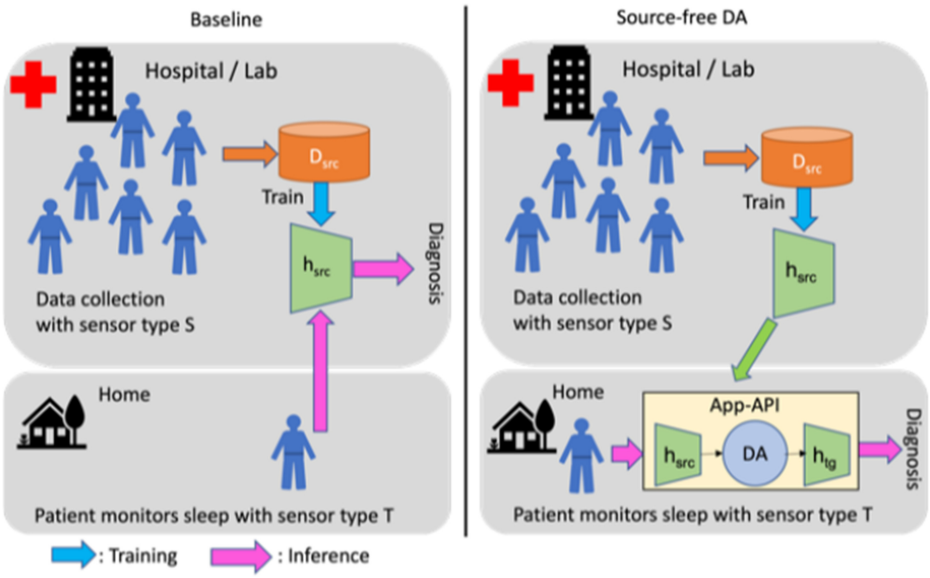


Fig. 1. Left: Baseline scenario. A lab gathers data from multiple patients with sensor types S. A classifier is trained with the collected data. Then a subject gathers data from their home with the use of a different sensor type T -of potentially lower-quality. Then the trained classifier performs automated inference on the subject's data. A diagnosis is given based on the automatic scoring. Right: Source-free DA case. The trained classifier is released to the subject, e.g., via an app. Source-free DA is then performed for the classifier to adapt to the subjects data. Then the adapted classifier performs inference. A diagnosis is given based on the automatic scoring from the adapted classifier.

personal information from a classifier, especially for time series data, is harder than having direct access to the true data of the individual. At the same time, we take into account the possible lack of hardware resources, since training will be done at the end-user. By using only a single trained classifier and only the data from the end-user, the proposed approach is less resource intensive than normal unsupervised DA.

In this work, we include the following contributions: (1) we introduce *SICO*, a DA technique which leverages the neuronal excitation of the output neurons as a means to iteratively select high confidence regions to train with. The goal of this process is to incrementally address the existing domain shift, and generalize well to the data of the end-user; (2) we apply the proposed approach on the real-world problem that we are interested in, namely sleep apnea detection and we show its effectiveness for different physiological sensors and datasets; (3) we use the proposed approach to perform DA for a different type of data and for a different task (i.e., digit classification), showcasing its generalizability and providing a comparison basis with related literature.

The rest of the paper is organized as follows: Section 2 presents the proposed approach. Section 3 compares *SICO* with related literature from unsupervised DA and time-series DA. We place our related work in this location since we discuss methodological differences between *SICO* and other methods, and we want the reader to know how our approach works before reading the related work. Section 4 describes the experiments we performed. Section 5 presents the choices of important hyper-parameters of *SICO*, and the results from our experiments. In Section 6 we discuss the key empirical insights. Finally, Section 7 concludes this paper.

2 METHOD

In this section, we describe the proposed approach, and provide insights about how and why it works. Furthermore, we investigate important properties, and analyze choices for the principal components of *SICO*.

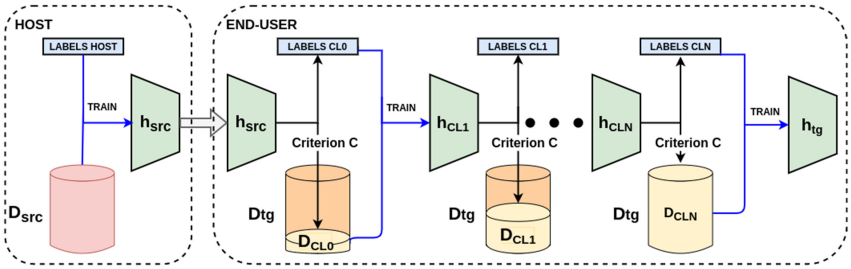


Fig. 2. Steps of *SICO*: We train h_{src} on D_{src}, and we release it to the end-user (target). The end-user incrementally adapts new classifiers to its domain based on the iterative beliefs of $h_{src}, h_{CL1}...h_{CLn}, h_{tg}$ and a criterion C. Classifier h_{CLn} corresponds to the last classifier that contributes labels to the target data, n the final application of C, and D_{CLn} the last dataset (subset or equal to D_{tg}) that will be used.

2.1 Background

We assume two datasets $D_{src} = \{(\mathbf{x}_S^{(1)}, y_S^{(1)}), \ldots, (\mathbf{x}_S^{(M_S)}, y_S^{(M_S)})\}$, and $D_{tg} = \{\mathbf{x}_T^{(1)}, \ldots, \mathbf{x}_T^{(M_T)}\}$, where M_S, M_T are the sizes of the respective datasets. All inputs $\mathbf{x}$ belong to an input space X which is a subset of $\mathbb{R}^d$, where d is the dimensionality of the input space. A domain shift exists between the two datasets, i.e., the generating processes of $\mathbf{x_S}$ and $\mathbf{x_T}$ differ. All labels belong to a label space $Y = \{0, \ldots, N_Y - 1\}$, where N_Y is the number of possible classes.

The goal of Unsupervised DA is to predict the labels for D_{tg} on the *task* [3] which the D_{src} labelling corresponds to. To do this, access to both D_{src} and D_{tg} is assumed, and as such unsupervised DA methods can take advantage of domain-specific characteristics of each dataset to learn the domain shift during training.

In this work, based on the scenario we are interested in, we investigate a more constrained version of unsupervised DA, namely *Source-Free* DA. We assume that access to D_{src} is not given. Instead, a classifier h_{src} is trained with D_{src} at the host which has access to this dataset, and afterwards h_{src} is released to the end-user patient (see Figure 2). The end-user has access to the unlabeled dataset D_{tg} and wants to classify D_{tg} on the same task that h_{src} has been trained for. As such, during training of *SICO*, we only have access to D_{tg} and h_{src}. The goal of *SICO* is to create a new classifier h_{tg} that is adapted to D_{tg} with the use of h_{src}.

2.2 Step-Wise Increasing Target Dataset Coverage based on High Confidence

We assume that h_{src} is a neural network which performs density estimation of the true conditional distribution $p(y|x)$. As such, $h_{src} : X \rightarrow \mathbb{R}^{N_Y}$ with $\sum_c h_{src}^{(c)} = 1$. The core idea of the proposed approach (see Figure 2) is to start with the data of the host D_{src}, and then train h_{src} with D_{src}, and release it. From the unlabeled dataset D_{tg} of the end-user, we select the subset of D_{tg} that satisfies a criterion $C\{h_{src}\}$, which we call D_{CL0}. This subset shall consist of the data for which h_{src} is highly confident about their true label. Thus, we use h_{src} to label D_{CL0}, and train a new classifier h_{CL1} on D_{CL0} with these labels. We repeat the procedure with h_{CL1} labelling from the remaining data of $D_{tg} - D_{CL0}$ in order to create a new dataset D_{CL1} consisting of D_{CL0} together with all the data that satisfy $C\{h_{CL1}\}$ from $D_{tg} - D_{CL0}$. We then train h_{CL2} from D_{CL1}. We repeat these steps until a terminal condition is being met (like, e.g., labelling of the whole D_{tg} or usage of a pre-determined number of classifiers). In algorithmic form, the method includes the following steps, and is shown with Algorithm 1:

- **Step 1:** h_{src} is trained on D_{src} and is released to the end-user patient.
- **Step 2:** Based on a given confidence criterion $C\{h_{src}\}$ (for example that the logits of the classifier for a class are larger than a threshold T) choose a subset $D_{CL0} \subset D_{tg}$ such that $C\{h_{src}\}$ is satisfied $\forall \mathbf{x}_T^i \in D_{CL0}$.
- **Step 3:** Based on the labels Y_{CL0} that h_{src} produces for D_{CL0} train a new classifier h_{CL1} with D_{CL0} and Y_{CL0}.

- **Step 4:** Set $i = 1$. Repeat Steps 5, 6, and 7 while $D_{CL(i-1)} \subset D_{tg}$, and while a terminal condition is not met (e.g., $i \leq N$).
  - **Step 5:** Based on $C\{h_{CLi}\}$ choose D_{CLi} such that $D_{CL(i-1)} \subset D_{CLi} \subseteq D_{tg}$. D_{CLi} is defined as the subset: D_{CLi} =(($D_{tg} - D_{CL(i-1)}$) s.t $C\{h_{CLi}\}$) $\cup D_{CL(i-1)}$. Thus, D_{CLi} contains all $D_{CL(i-1)}$ together with all the datapoints of ($D_{tg} - D_{CL(i-1)}$) which satisfy $C\{h_{CLi}\}$.
  - **Step 6:** Use h_{CLi} to label ($D_{CLi} - D_{CL(i-1)}$). Unite with $Y_{CL(i-1)}$ to create Y_{CLi}.
  - **Step 7:** Based on the labels Y_{CLi} that $h_{CLj}, j = src, 1, \ldots, i$ have produced for D_{CLi} train a new classifier $h_{CL(i+1)}$ with D_{CLi} and Y_{CLi}. Set $i = i + 1$.
- **Step 8:** Return $h_{tg} = h_{CLi}$.

Choosing a good criterion C is important for the success of the algorithm. If we choose an improper C and we have a demanding terminal condition to meet (like having a high threshold of belief $\forall \mathbf{x}_T^i \in D_{tg}$), the algorithm could "get stuck", or the performance could suffer. We discuss choices for C in the next subsections.

2.3 Methodological Analysis

We need two core characteristics for the method to work well: (1) h_{src} must be sufficiently trustworthy such that if we satisfy $C\{h_{src}\}$ for a given datapoint $\mathbf{x}_T^i$ from the input space, there is a high probability that $\mathbf{x}_T^i$ belongs to the true class that h_{src} predicts. Thus, if the data distributions of the host and the end-user are very different, h_{src}, and subsequently h_{tg}, will both have difficulty achieving high performance. This observation is in-line with the theoretical analysis for domain adaptation from [6] (see Theorem 1). (2) h_{CLi} must be trained on a subset of D_{tg} with labels for which we are confident to generalize to new data from D_{tg} with high confidence. This is equivalent to a classifier generalizing well to new data. Thus, the design of h_{CLi} needs to be good enough, and also the dataset $D_{CL(i-1)}$ with which h_{CLi} has been trained should be large enough to give h_{CLi} the ability to generalize well. Characteristic (2) is needed for all classifiers during the algorithm.

At the last step of the algorithm, the empirical risk of h_{tg} (for cross-entropy loss) for the last dataset $D_{CLn} \subseteq D_{tg}$ is:

$$\hat{L}(h_{tg}) = -\frac{1}{|D_{CL_n}|} \sum_{\mathbf{x}_T^j \in D_{CLn}} \sum_c y_{j,CLi}^c log\Big(h_{tg}^c\Big(\mathbf{x}_T^j\Big)\Big)$$

where $y_{j,CLi}^c$ is the element c of the one-hot encoded-vector of $argmax_c\{h_{CLi}^c(\mathbf{x}_T^j)\}$, with $i \in \{0, \ldots, n\}$, the index for the classifier which labelled $\mathbf{x}_T^j$, and 0 referring to h_{src}. Additionally, it is not necessary that $D_{CLn} = D_{tg}$ as we could stop the algorithm before the criterion holds for the whole D_{tg}. Assuming column vectors, we define the true empirical risk of h_{tg} with the true labels as:

$$\begin{aligned} L(h_{tg}) &= -\frac{1}{|D_{CL_n}|} \sum_{\mathbf{x}_T^j \in D_{CLn}} \sum_c y_j^c log\Big(h_{tg}^c\Big(\mathbf{x}_T^j\Big)\Big) \\ &= -\frac{1}{|D_{CL_n}|} \sum_{\mathbf{x}_T^j \in D_{CLn}} \sum_c y_{j,CLi}^c log\Big(h_{tg}^c\Big(\mathbf{x}_T^j\Big)\Big) - \frac{1}{|D_{CL_n}|} \sum_j \delta_j^\intercal \cdot log\Big(h_{tg}\Big(\mathbf{x}_T^j\Big)\Big) \\ &= \hat{L}(h_{tg}) + \frac{1}{|D_{CL_n}|}\Delta(h_{tg}) \end{aligned} \tag{1}$$

with $\delta_j = y_j - y_{j,CLi}$, and y_j the true label vector of $\mathbf{x}_T^j$. Additionally, $\delta_j = \mathbf{0}$ when $y_j = y_{j,CLi}$. Intuitively, Δ can be thought as a form of accumulative error of the algorithm, and the larger it is the bigger the difference of the true loss and the actual loss we are minimizing. This obviously has a negative impact in the performance on new data.

ALGORITHM 1: Step-Wise Increasing Target Dataset Coverage based on High Confidence

Source:
Input: Dataset D_{src}, Host Labels (for D_{src})
h_{src} = Train(h_{src},D_{src}, Host Labels);
Make h_{src} available to Target (Patient), but not D_{src};
Output: h_{src}

Target:
Input: Dataset D_{tg}, h_{src}
Initialize Criterion C;
$D_{CL0} \leftarrow \{\mathbf{x}_{tg} \in D_{tg} | C\{h_{src}(\mathbf{x}_{tg})\}\}$;
$Y_{CL0} \leftarrow$ One-hot(Label(D_{CL0}, h_{src}));
Reinit(h_{src});
$h_{CL1} \leftarrow$ Train(h_{src},D_{CL0}, Y_{CL0});
$i \leftarrow 1$;
while $D_{CL(i-1)} \subset D_{tg}$ *and* $\neg$ *TerminalCondition*($D_{CL(i-1)}$,$Y_{CL(i-1)}$) **do**
  $S \leftarrow \{\mathbf{x}_{tg} \in (D_{tg} \setminus D_{CL(i-1)}) | C\{h_{CL(i)}(\mathbf{x}_{tg})\}\}$;
  $D_{CLi} \leftarrow D_{CL(i-1)} \cup S$;
  $Y_{CLi} \leftarrow Y_{CL(i-1)} \cup$ One-hot(Label(S, h_{CLi}));
  Reinit(h_{CLi});
  $h_{CL(i+1)} \leftarrow$ Train(h_{CLi},D_{CLi}, Y_{CLi});
  $i \leftarrow i + 1$;
end
$h_{tg} \leftarrow h_{CLi}$;
Output: h_{tg}

In the labels Y_{CLi} for a dataset D_{CLi} there are as many errors expected as there are made from h_{CLi}'s generalization, plus the errors that were "passed" from the generalizations of the previous classifiers. The generalization error of h_{CLi} is also dependent on the generalization error from the previous classifiers, because it uses $D_{CL(i-1)}$ for training with the labels of the previous classifiers. Based on this discussion, and assuming that we use n classifiers, Δ can be rewritten recursively in the following form:

$$\begin{aligned}
\Delta(h_{tg}) &= \Delta_n(h_{tg}; h_{src}, h_{CL1} \dots h_{CLn}) \\
&= - \sum_{\mathbf{x}_T^j : y_j \neq y_{j,CLi}} \delta_j^\intercal \cdot log\Big(h_{tg}\Big(\mathbf{x}_T^j\Big)\Big) \\
&= - \sum_{\mathbf{x}_T^j \in (D_{CLn} - D_{CL(n-1)})} \delta_j^\intercal (h_{src}, h_{CL1} \dots h_{CLn}) \cdot log\Big(h_{tg}\Big(\mathbf{x}_T^j\Big)\Big) \\
&+ \Delta_{n-1}(h_{tg}; h_{src}, h_{CL1} \dots h_{CL(n-1)})
\end{aligned} \tag{2}$$

where we define as $\Delta_0(h_{tg}; h_{src}) = -\sum_{\mathbf{x}_T^j \in D_{CL0}} \delta_j(h_{src}) \cdot log(h_{tg}(\mathbf{x}_T^j))$. From this form it is straightforward that the earlier classifiers play a more important role in the performance of h_{tg}, since their error in a sense "propagates" through the next training iterations. Thus, h_{src} plays the most important role as the first classifier in the algorithm.

Furthermore, for every step i of the algorithm, the generalization capability of the classifier h_{CLi} plays an important role in the accumulative Δ. If h_{CLi}'s generalization capability is low, it will assign many erroneous labels during the labelling of its high confidence dataset for criterion $C\{h_{CLi}\}$. This will also affect the next steps,

since we have to expect worse generalization capability for the next classifiers, because these data will become training data for the next classifiers. The above analysis can be applied to any h_{CLi} and its respective $D_{CL(i-1)}$ instead of h_{tg} and D_{CLn}.

2.4 Choosing the Belief Criterion

An essential part of the proposed method is the criterion C that we use to choose for a classifier h_{CLi} the new subset of D_{tg} to include to $D_{CL(i-1)}$, i.e., $(D_{CLi} - D_{CL(i-1)})$. We label this part with h_{CLi}, and then combine it with $D_{CL(i-1)}$ and use to train $h_{CL(i+1)}$. C is a very important part of the algorithm, as it is responsible for the choice of a subset which is largely from erroneous labels. In our case, we assume that all h_{CLi}, h_{src}, h_{tg} are neural networks with softmax output. We therefore take advantage of the neuronal excitation of the output class neurons, and use it as an indication of confidence that $\mathbf{x}_T^i$ belongs to a certain class. We hypothesize that such high-confidence datapoints are more likely to indeed belong to their assigned class.

There are many ways we can use the excitation of the output neurons as a criterion to assign labels for a high confidence sub-dataset. Examples include thresholding, selecting the m datapoints which lead to the strongest activation for each class neuron, or selecting the top p_{cl} percent of datapoints that activate each class neuron cl the most.

2.5 *SICO* and Curriculum Learning

At first glance *SICO* and **Curriculum Learning (CL)** [7] can appear to be quite similar. In CL, the training "starts small" by using a small training set with easy examples identified with the use of a scoring function [8]. Afterwards, CL progressively utilizes more difficult examples which are added to the curriculum. Similarly, we utilize easier examples in terms of domain similarity, and progressively train with harder datapoints. The previous classifier's class probabilities give us a measurement of the datapoints that are easier for the classifier in terms of the classifiers' higher confidence regarding these points (lower entropy). We hypothesize that datapoints that are easier in terms of lower entropy for a classifier trained in a different domain are more likely to be more similar to datapoints from the source domain, in terms of class separation.

However, the basic vanilla CL uses a static scoring function during training. *SICO* utilizes instead a sequence of scoring functions (i.e., h_{CLi}) that are learned dynamically as the training process continues. The creation of a new scoring function depends on the dataset and the previous scoring function. Additionally, the first scoring function, i.e., h_{src}, is independently trained and acts as a Teacher that provides the initial scoring paradigm, in conjunction with the belief criterion used. We compare our method with other more relevant and recent works of CL for DA in Section 3.

3 RELATED WORK

There is a large body of work in DA, mainly focused on the visual domain, which is to a large extent covered by existing surveys [9, 10]. In one of these surveys [10] Wang et al. separate DA based on the type of learning (supervised, semi-supervised, unsupervised), whether or not the feature space is the same in source and target domains, and whether the work performs one-step or multi-step DA. We follow a similar separation but due to space limitations we focus on the works that are mostly related to ours. Thus, we separate on the basis of supervised or unsupervised DA, with more emphasis on unsupervised DA, and for the unsupervised DA case whether the proposed work has direct or indirect access to the source data (see Figure 3). Furthermore, we separate for visual and time-series applications or methods. For the works which resemble our work, we compare and discuss about specific similarities and differences.

For the case of unsupervised visual DA, a large body of literature use variations of adversarial DA for visual applications [11, 13–17]. A good example of an advanced technique that also includes adversarial elements is CyCADA [18]. It is a technique for unsupervised DA which uses a team of models trained on a unified loss which

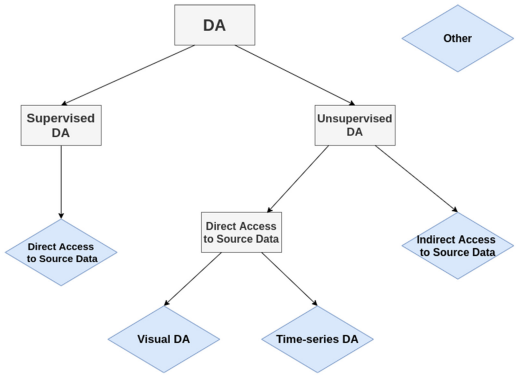


Fig. 3. Classification of presented related works in supervised or unsupervised DA. We show with blue rhombuses the leaf subcategories for which we discuss related works.

consists of task losses, cycle-consistent loss [19], GAN losses, and semantic consistency loss. The goal is to adapt representations at pixel and feature level while enforcing cycle and semantic consistency loss. Other works for visual applications that do not directly include adversarial training are [20, 21]. Additionally, works exist which, like our proposed method, make use of pseudo-labelling on the target domain [22–27]. However, all of these approaches take advantage of both source and target domain data during their training phase. Furthermore, they use different methods for classifier adaptation on the conditional distribution of the target domain during their training phase, like entropy minimization, minimization of the conditional distribution's MMD between source and target, pseudo-labelling with multiple classifiers, and so on.

As mentioned, most of the above techniques perform unsupervised DA with access to labeled source data and unlabeled target data. However, we focus on DA at the end-user with access only to unlabeled target data and a trained classifier at the source data and labels, making it an inherently harder problem. Despite these additional restrictions, *SICO* yields an on average comparable or superior performance relative to several other well-established unsupervised DA works for the digit datasets. As such, one key advantage of *SICO* relative to other well-established unsupervised DA approaches is that *SICO* is inherently designed to handle the source-free DA case. This means that for *SICO* to work joint training with the source and target data is not necessary.

Additional works that provide a solution to unsupervised DA without direct access to the labeled source data include [28–30]. Wulfmeier et al. [28] address the problem of degraded model performance due to continuously shifting environment conditions. They develop incremental adversarial DA with which they redefine DA as a stream of incrementally changing domains, to enable a classifier to adapt for example to changing weather conditions or day night circle. Additionally, they propose an extension to use only target data assuming an additional GAN generator which has learned the encoded feature marginal distribution of the source data. Similarly, our method uses the source data indirectly via h_{src}, but we investigate the normal DA scenario, and not a case of transitioning incremental domain shifts. Li et al. [29] introduce **Adaptive Batch Normalization (AdaBN)**, and use only a classifier and the target data. They standardize each neuron's output with the average and standard deviation of its output from the images in the target domain. This standardization ensures that each neuron receives data from a similar distribution, no matter which domain the data come from. Using this idea, Zhang et al. [31], train a convolutional network with layers of various kernel sizes for fault diagnosis on raw vibration signals. They then perform AdaBN for the CNN model trained in the source domain. This approach does not train with the target domain data, and is only a form of cross domain normalization for the trained classifier. As a result we hypothesize that it does not take full advantage of the knowledge potential of the target sample. Several recent methods have been developed to address the problem of Source-free DA, like [32], where a

multi-term adversarial objective with additional contrastive and rotation loss terms is used to achieve Source-free DA, or [33], which aims to achieve universal Source-free DA (where labels from source and target domains can both include domain-private labels). *SICO* differs from these works as we do not use adversarial objective, and also, for the scenario we study in this work, our source-target label-set relationship is closed-set (same labels in both domains). Additionally, Yang et al. [30] introduce **Generalized Source Free DA (G-SFDA)**. With G-SFDA, a network is pre-trained on the data and labels of the source domain. Then the pre-trained model is trained in an unsupervised manner with the data of the target domain. Specifically, the model is trained on the target domain via identifying the K-closest neighbors of each datapoint based on feature similarity. A class homogeneity loss on each datapoint is imposed based on the classes of their neighbors. Additionally, to avoid forgetting source-specific or target-specific patterns sparse domain attention is used, where source and target masks hinder forward and backward passes for data from each domain. Hence, specific source and target information flows are imposed to the learning process of the model. Kim et al. [34] introduce a similar technique for Source-free DA. By using a trained model from the source domain (feature extractor and classifier), they train a new feature extractor and two new classifiers which adapt to the target data. One of the two classifiers maintains a source loss with the use of the source classifier, mainly for regularization. The other classifier performs the actual DA via periodically labelling an adaptive low-entropy prototype set per-class. These sets serve as class-representatives through which a per-class similarity metric is established for all target data. An additional confidence-based filtering mechanism is applied on the target data with which a reliable sample is established only when the prototypes of the most similar class are closer than prototypes of the second most similar class. The final loss takes into account both the source loss and the established self loss defined via the aforementioned process. One advantage of *SICO* in comparison to this approach is that inherently *SICO* is a simpler approach that needs less memory on a best case scenario. That is since *SICO* does not need a feature bank and a score bank which are needed in G-SFDA to identify and compare the closest neighbors, or sufficient memory for utilization of multiple models, as is the case for [34]. This is an important advantage in our case, since the scenario we are interested in is inherently resource-constrained (i.e., local device of the end-user patient).

Finally, Chidlovski et al. [35] explore the idea of source-free DA, given access to either a few representative source samples (specifically class means) or a set of classifiers. In the first case, a **stacked Marginalized Denoise Autoencoder (sMDA)** [36] is used. The class means are fed jointly with the target domain data in the sMDA. The denoised class means are then used to classify the target domain examples, with the use of a weighted softmax distance. In the second case, the output of the cassifiers is jointly fed with the target samples to the sMDA and then the reconstructed classifier's output is used to predict the class of the respective target domain sample. Both of these approaches differ substantially from our approach. Specifically, *SICO* does not utilize a generative denoise framework, and more importantly, the target data modify the classifiers and not directly the labels.

For the case of unsupervised DA for time-series data, Purushotham et al. [37], use Variational Recurrent Neural Nets [38], and employ adversarial DA to train in order to achieve DA from the latent representations. In [39], Aswolinskiy et al. propose Unsupervised Transfer Learning via Self-Predictive Modelling. In the proposed approach, a linear transformation Q is learned that minimizes the error identified by a self-predictive model between the source and the transformed by Q target domain. Then a classifier trained in the source domain is applied in the transformed target domain. We assigned this work under the umbrella of unsupervised DA since, the authors present a general DA framework and because it fits the data access criteria we specify. Other works include [31, 40, 41]. Chai et al. [40] perform EEG DA for sentiment recognition by using a linear transformation function that matches the marginal distributions of the source and target feature spaces. They additionally employ a pseudo-labelling approach to transfer confident target data to the transformed source domain which relates to our own approach for choosing confident samples. Natarajaan et al. [41] use DA in order to mitigate the negative effects from the lab-to-field transition for cocaine detection using wearable ECG.

In the context of supervised DA, Persello et al. [42, 43], use Active Learning for DA for classification of remote sensing images. This approach loosely relates to ours, in the sense that it iteratively trains the classifier with a

differentiating training set. However, it assumes access to the source data (for data points to be removed from the training set), and a user in the role of supervisor, which assigns labels for chosen samples.

Finally, other works that are related in the context of our work include [44–46]. Fawaz et al. [44] use transfer learning for time series classification. They evaluate on the UCR archive for all possible combinations, and use a method that relies on Dynamic Time Warping to measure the similarity between the different datasets. Regarding CL, both works included apply DA in the context of semantic segmentation. Zou et al. [45] utilize pseudolabels and self-training to learn the new domain. Contrary to our work, they use a loss which contains terms for the loss of the source and the target domain plus a L_1-regularization term on the pseudolabels. They optimize in a two-step fashion, optimizing first the pseudolabels and then the classifier weights. Zhang et al. [46] derive their curriculum by separating between the hard task of semantic segmentation from the easy task of learning high-level properties of the unknown labels. They minimize a joint loss which includes terms from both domains. For their easy task they infer the target labels and landmark pixel labels by utilizing training and labelling in the source domain (e.g., retrieving the nearest source neighbors for a target image and transfer the labelling).

4 EXPERIMENT DESCRIPTION

The main goal of this work is to perform successful domain adaptation for bio-sensor signals for the purpose of sleep apnea detection. Additionally, we complement the physiological datasets with datasets for digit classification, i.e., USPS, MNIST, and SVHN for two reasons: (1) the majority of the related literature uses these datasets and so we aim to compare *SICO* with related works on their premises; (2) we want to investigate the generalizability of our approach for different types of data.

4.1 Datasets

We use the following six datasets to evaluate *SICO*:

- **Apnea-ECG** [47] (*AE*) is an open dataset from Physionet, containing sensor data from chest respiration, abdomen respiration, **nasal airflow (NAF)**, oxygen saturation and **Electrocardiograph (ECG)**. *AE* has been used in the *Computers in Cardiology* challenge [47] and it contains high quality data. It has been collected with Polysomnography in a sleep laboratory. From the 35 ECG recordings in the dataset, 8 recordings (from 8 different patients) contain data from all the sensors. Each recording has a duration of 7–10 hours. The sampling frequency of all sensors is 100Hz, and labels are given for every one-minute window of breathing. The labels identify which minutes are apneic and which are not (i.e., if a person experiences an apneic event during this minute). From *AE*, we use the NAF, chest respiration, and oxygen saturation signals.
- **MIT-BIH** [48] (*MB*) is an open dataset containing recordings from 18 patients. The recordings contain different respiratory sensor signals. In 15 recordings, the respiratory signal has been collected with NAF. For this reason, we focus on the NAF signal for the *MB* dataset. Due to misalignment of the different signals and lack of labels for the apneic class in four recordings, we utilize 11 of the 15 recordings (slp60, slp41, and slp45 and slp67x are excluded). The data/labelling quality of the *MB* dataset is low, which leads to low classification performance for *MB* compared to the other respiratory datasets that we investigate [49]. The labels are given for every 30 seconds and the sampling frequency of all sensors is 250Hz.
- The **A3 study** [50, 51] (*A*3) investigates the prevalence, characteristics, risk factors and type of sleep apnea in patients with paroxysmal atrial fibrillation. The data were obtained with the use of the Nox T3 sleep monitor with unattended sleep monitoring at home, which in turn results into lower data quality than data from polysomnography in sleep laboratories. An experienced sleep specialist scored the recordings manually using Noxturnal software such that the beginning and end of all types of apnea events is marked in the time-series data. To use the data for the experiments in this paper, we labeled every 60 second window of the data as apneic (if an apneic event happened during this time window) or as non-apneic. The data

we use in the experiments is from 438 patients and comprises 241,350 minutes of sleep monitoring data. The ratio of apneic to non-apneic windows is 0.238. We use only the NAF signal from the A3 data in the experiments, i.e., the same signal we use from Apnea-ECG.

- **MNIST** [52] (M) is a dataset containing 60,000 28×28 images of digits (handwritten black and white images of 0–9) as a training set. The test set comprises of 10,000 images.
- **USPS** [53] (U) is another handwritten digit dataset (0–9), which contains 7,291 grayscale 16×16 training and 2,007 test images.
- **SVHN** [54]: (S) is s a real-world image dataset obtained from house numbers in Google Street View images. Similarly to the previous datasets, classification is performed for digits 0–9. It contains 73,257 digits (32×32 colored images) for training, 26,032 digits for testing, and 531,131 additional training data. We use only the original training dataset of 73,257 digits.

4.2 Preprocessing

The data in all sleep apnea datasets is standardized (per physiological signal), downsampled to 1Hz and the windows are shuffled randomly. The data from the $A3$ study, are very unbalanced, i.e., it contains many more non-apneic than apneic one-minute windows. Therefore, we rebalance the dataset to contain equal amount of apneic and non-apneic one-minute windows. Since the labels in MB are given every 30 seconds, while AE and $A3$ are labeled every 60 seconds, we adapt the labelling in MB to 60 seconds by using the following rule: if both 30 second labels are non-apneic then the 60 seconds label is non-apneic, elsewise it is apneic. For $A3$, we use 80% of data as training and 20% as test set. For AE we use 25% of the data as test set, and for MB we use 15% of the data as test set. We use less test set data for MB because we want to utilize more data for training due to the low quality.

We rescale the data in all digit classification datasets from 0–255 to 0–1. Additionally, for U and S, we restructure the data so that it is in similar form to the M data. We up-scale the images in U from 16×16 to 28×28, and we downscale the images in S from 32×32 to 28×28. Additionally, we convert the color images in S to grayscale images.

Finally, it is worth noting that for all classifiers in all experiments we attempt to train them generally for as few epochs as possible while maintaining their generalization capability. We do this based on findings from our previous work [55] where we establish that under-training classifiers under label noise can yield performance advantages when generalizing to new -noise-free- data. In essence, the labelling we obtain from previous classifiers can be considered noisy due to the inherent likelihood of error inclusion (see Equation (2)). As such with this strategy (i.e., training for fewer epochs), we attempt to alleviate the accumulation of errors, independently from the good choice of criterion C.

4.3 Experimental Set-Up

We follow in the experiments the steps outlined in Section 2, i.e., we train h_{src} such that it can generalize well for the test set of D_{src} and release h_{src} to the end-user. Then we use h_{src} and a subset of D_{tg} to iteratively create h_{tg}. The performance of h_{tg} is evaluated with the test set of D_{tg}. This means that we evaluate the proposed method on the test set of D_{tg}. We use the convention $D_{src} \rightarrow D_{tg}$ to indicate that h_{src} is trained on D_{src} and h_{tg} is evaluated on D_{tg}. Further details about other minor algorithmic decisions can be found in Appendix A. Since D_{tg} is not labeled, we do not have access to a validation set during the training of the proposed method. Thus, we train each classifier h_{CLi} for a fixed number of batch iterations.

5 EXPERIMENTS AND RESULTS

In this section we present our results. First we discuss how important hyper parameters affect *SICO*. Then we present our main results in two sets of experiments. The first set of experiments focuses on sleep apnea detection

with physiological sensors. The application scenario that we are interested in focuses on the transition from high quality sensor data to low quality sensor data. Additionally, we investigate combinations for which the low quality data are used as D_{src}, in order to get a more complete picture of the capabilities of the proposed algorithm. We focus on the NAF signal for the combinations $A3 \rightarrow AE$, $A3 \rightarrow MB$, $MB \rightarrow AE$, $AE \rightarrow MB$, $AE \rightarrow A3$, and evaluate the abdominal respiration (Resp A), the chest respiration (Resp C) and the oxygen saturation (Sp02) sensor signals for the $A3 \rightarrow AE$ and $AE \rightarrow A3$ combinations. The second set of experiments evaluates *SICO* for digit classification and showcases comparison with several related works for three combinations that are commonly used in literature, i.e., $M \rightarrow U$, $U \rightarrow M$, $S \rightarrow M$.

Devices: Please note that for the sleep apnea set of experiments, the devices used are of the same type across the different datasets (*AE*, *MB* and *A3*) i.e., nasal thermistors for the NAF signal, **Respiratory Inductive Plethysmography (RIP)** (Chest-Abdomen belts) for the Resp A and C signals, and pulse oximeter for the measurement of oxygen saturation Sp02.

Metrics: For the Digit Classification experiments, we use accuracy as performance metric since it is commonly used in related literature for the particular task, assuming a well-balanced dataset. For the apnea detection experiments, we use the kappa coefficient [56] as performance metric since it better captures performance characteristics in a single metric than accuracy, as it takes into account the possibility that two annotators (real labels and predictions in our case) agree by chance. For completion, we present the accuracy, specificity, and sensitivity results in Appendix B. All experiments are repeated five times, and we present the average results and the standard error.

5.1 Hyper-Parameters and Tuning

In this subsection we discuss the effects of important hyper-parameters on the *SICO* algorithm. Specifically, we analyze the behaviour of *SICO* for different Belief Criteria and Labelling Subset sizes of classifiers h_{CLi}.

5.1.1 Belief Criteria. We empirically investigated several different methods for the choice of high-or low - confidence data. Figure 4 showcases the performance for 10 runs of each method for $U \rightarrow M$. We outline the chosen methods:

- **All-In:** We label all data with h_{src} and use them all directly as D_{CL0}.
- **Weighted All-In:** We label all data with h_{src} and use them all directly as D_{CL0}. We weight the loss of each datapoint with the maximum probability of the softmax output of this datapoint. As such high-confidence datapoints are prioritized relative to low-confidence datapoints. We relabel with h_{CLi} and repeat the process.
- **Threshold (Low, High):** We label D_{tg} with h_{src}. We then use a static threshold (typically between 0.5 and 0.99) to determine high-confidence datapoints, based on whether their top class probability surpasses the threshold. We train h_{CL1} with these high-confidence data and repeat the process until we include all D_{tg}.
- **Threshold (Dynamic-Average):** We label all data with h_{src}. We then calculate for each class the average threshold of the datapoints that belong to this class based on the labelling from h_{src}. We use these thresholds as belief criterion, i.e., we include into D_{CLi} data that surpass any of these thresholds. We again repeat until we cover the whole D_{tg} dataset.
- **Top-m:** Based on labelling of h_{src} we choose to include to D_{CL0} the top m datapoints in terms of class probability for the respective class. We repeat until we cover all of D_{tg}.
- **Top-p%:** Based on labelling of h_{src} we choose to include to D_{CL0} the top p% of datapoints in terms of class probability for the respective class. We repeat until we cover all of D_{tg}.
- **Min Entropy:** Similar to the threshold case, but instead we choose whether or not to include datapoints to D_{CLi} based on a minimum entropy threshold for the class distribution of each datapoint.
- **Low Only:** Only use datapoint for which their max class probability is less than a certain threshold (in order to exploit data near the class separation region instead of high-confidence datapoints).

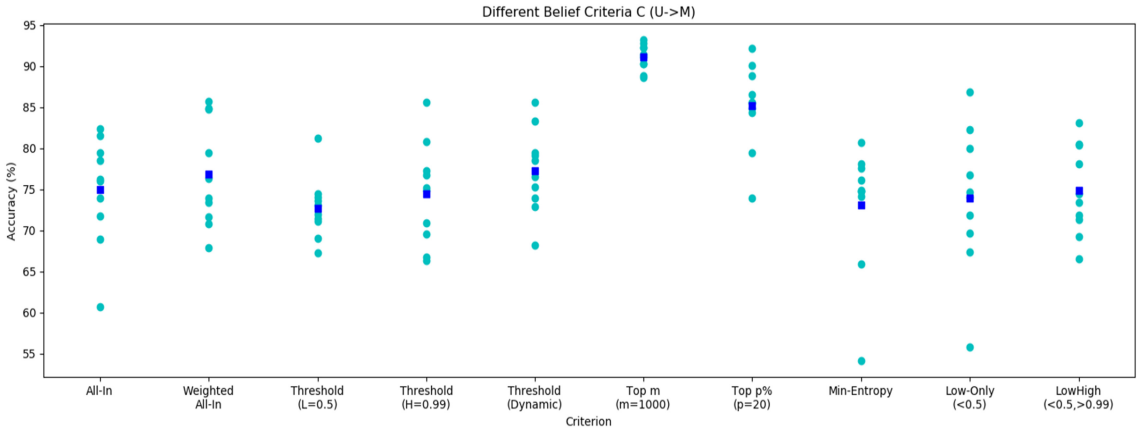


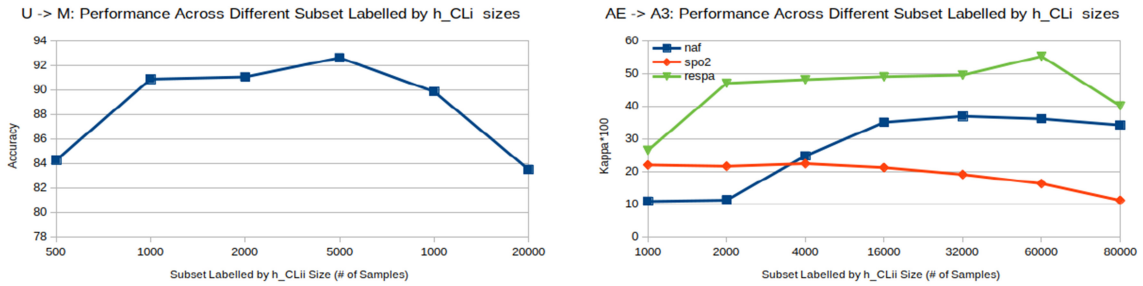
Fig. 4. Comparison of several criteria as C. Cyan points show performance (Accuracy) of indiviual runs, and blue points show average performaces.

- **LowHigh:** Use datapoint for which their max class probability is less than a certain threshold or greater than another higher threshold.

Overall, as we observe from Figure 4, most methods did not perform well. We hypothesize this is due to the biases which are transferred from the source domain to the target domain towards certain classes. These biases result into unbalanced D_{CLi}, which can negatively affect the overall generalization performance. Given that the simple All-In case can serve as a baseline, we notice that generally Weighted All-In, Dynamic Threshold, Top-m and Top-p% cases yield on-average superior results. As expected Weighted All-in case yields superior results to All-In as it de-prioritizes low-confidence datapoints. Both static threshold cases yield mediocre results relevant to All-In,which can potentially be attributed to the class biases of the source domain. Dynamic Threshold compensates for this issue by taking averages per-class as thresholds, however, we do not observe significant improvements relative to the other cases, potentially due to the inclusion of too many medium-confidence datapoints. Interestingly, Min-entropy performed inferior to most Threshold cases. We attribute this behaviour to the difficulty of choosing appropriate cut-off as minimum entropy. Low-Only performed better than anticipated, surpassing the Threshold(Low) case. This showcases the viability of including only low-confidence datapoints to identify the class separation regions for the task we are interested in. Yet Low-Only is not superior to Threshold(High). Finally, LowHigh yields the second best Threshold results, (Only behind Threshold (Dynamic)), which exhibits the viability of combining low-confidence and high-confidence datapoints.

Overall, Top-m and Top-p% yield the best results. This can potentially be attributed to the selective nature of these criteria, as we generally use small m and p in our experiments. For this reason, we select the m datapoints per-class which excite the class output neurons the most as belief criterion $C\{h\}$ in all experiments (except $M \rightarrow U$). We use this criterion in order to maintain the class balance since all datasets with the exception of U that are used as D_{tg} are relatively well-balanced. Since U is not well balanced, we choose a criterion that gives more "freedom" to the classifier to perform the balancing. Instead of choosing an absolute number N, as $C\{h\}$, we select a percentage p of the datapoints which activate a class output neuron the most.

5.1.2 Size of h_{CLi}'s Labelling Subset. A hyper-parameter that plays an important role in *SICO* for proper generalization is the size of the new subset that the classifier h_{CLi} labels (i.e., $(D_{CLi} - D_{CL(i-1)})$). In Figure 5, we show the effect of the size of $(D_{CLi} - D_{CL(i-1)})$ on the performance of *SICO* for $U \rightarrow M$, and three sensors of $AE \rightarrow A3$, namely *NAF*, *RespA*, and *Sp*02. For all cases, the performance improves with increasing subset size up to a certain point and drops afterwards with larger sizes. That is because if the subset is too small, proper



(a) (b)

Fig. 5. The impact of size of the new subset ($D_{CLi} - D_{CL(i-1)}$) in the performance of *SICO* for $M \rightarrow U$ (a), and $AE \rightarrow A3$ (b) (Blue: *NAF*, Orange: *Sp*02, Green: *RespA*).

generalization cannot happen, and as such all subsequent models yield lower performance. On the other hand, if the subset is too large, it will contain low certainty regions. Thus, the subset will most likely contain more errors. An interesting characteristic visible in Figure 5(b) is that the sensors reach peak performance with different subset sizes. For example, the peak performance of $Sp02 : AE \rightarrow A3$ is reached with a much smaller subset size than the other sensors. Since we want in our experiments to have parametric homogeneity when this is possible, we tune our parameters such that we strike a performance balance across the different sensor behaviours.

5.2 Source-Free DA for Sleep Apnea Detection

Figure 6 shows examples from the different sleep apnea datasets (before pre-processing). From the *MB* dataset we use only the NAF signal in our experiments. For this reason, we include only the NAF signal from *MB* in Figure 6. It is difficult to visually assess the respiratory signals and extract the domain specific features, especially since the variance of the data is very high. This is apparent in Figure 6 when trying to compare between the NAF data from *AE* and *MB*. Generally, though data from *MB* seem to have higher variance than the data from *AE* between apneic and non-apneic periods, and also higher fluctuations in the breathing pattern of the patients.

To perform sleep apnea DA, we use identical architectures for all classifiers, i.e., $h_{src}, h_{CL1} ... h_{CLn}, h_{tg}$. We use a 1D CNN (see Table 1), and use relu activations, dropout on the fully connected layers, and softmax activations on the output. When the *A*3 study is D_{src}, we train h_{src} for 15 epochs and when *MB*, or *AE* are D_{src} we train h_{src} for 20 epochs (in order to have more training iterations since *MB* and *AE* are smaller than *A*3). We use in all experiments a batch size of 128, learning rate of 0.001, and the Adam optimizer [57]. All differences in results between h_{src} on the D_{tg} test set and h_{tg} on the D_{tg} test set are statistically significant based on the one-tailed paired t-test (for $p = 0.05$), with the exception of Resp A: $A3 \rightarrow AE$, NAF: $A3 \rightarrow MB$, and Resp C: $AE \rightarrow A3$.

We use in all experiments with *AE* and *MB* as D_{tg} the fixed data criterion with 500 datapoints per-class for h_{src} and 200 datapoints per class for all subsequent h_{CLi} as belief criterion. With *A*3 as D_{tg}, we use the fixed data criterion with 10,000 datapoints per-class for h_{src} and 500 datapoints per class for all subsequent h_{CLi}, since *A*3 is much larger. The algorithm stops when we do not have any more unlabeled data in D_{tg}.

Table 2 presents the classification performance of the three combinations for h_{src} on D_{src} and h_{tg} and h_{src} on D_{tg}. From Table 2 we initially observe the significant impact of the quality of the datasets on the performance of h_{src}. For $MB \rightarrow AE$, the quality of the *MB* data is low enough that the evaluation of h_{src} on the test set of *MB* performs worse than the evaluation of h_{src} in *AE*. Since *AE* is a high quality dataset, it is easier to have much better performance. This is reflected by the very big difference of kappa (×100) between the two datasets for h_{src}(i.e., 41.69 vs. 94.39). For *A*3, we expected that it would have better transferability to the other datasets since it is much larger, thus covering a wider variety of cases (both from patient and from data quality perspective). Though this holds for the *AE* case, i.e., $A3 \rightarrow AE$, it is not the case for *MB*, for which ($A3 \rightarrow MB$) h_{src} performs

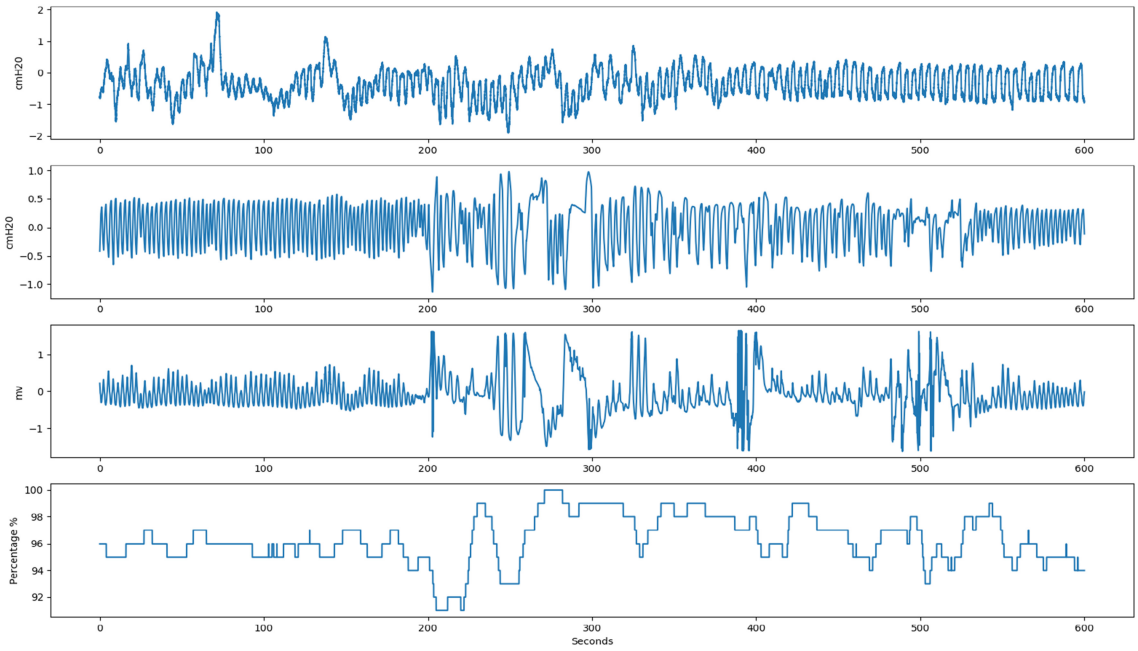


Fig. 6. Different respiratory signals from *MB* and *AE*. Rows show randomly chosen 600sec windows of respiratory signals from *MB* (first row with NAF) and *AE* (Second Row with NAF. Third row with Resp A. and Fourth row with Sp02). All *AE* data come from the same randomly chosen window.

Table 1. Architectures used for the Experiments (Conv: Input Channels × Output Channels × Filter, MP: Max Pooling, Fc: Fully Connected, Input × Output)

Layers	$M \leftrightarrow U$	$S \rightarrow M$	Sl.Apnea
Conv + MP	$1 \times 32 \times 5 \times 5$	$1 \times 32 \times 5 \times 5$	$1 \times 16 \times 4$
Conv + MP	$32 \times 28 \times 5 \times 5$	$32 \times 64 \times 3 \times 3$	$16 \times 32 \times 4$
Conv(+MP)	$28 \times 28 \times 5 \times 5$	$64 \times 128 \times 5 \times 5$	$32 \times 64 \times 4$
FC	$(7 \times 7 \times 28) \times 1024$	$(7 \times 7 \times 128) \times 3072$	$(8 \times 64) \times 64$
FC	1024×128	3072×1024	64×32
out	128×10	1024×10	32×2

Table 2. Performance for DA between Different Dataset Combinations for the NAF Sensor Signal

SICO h_{tg} performance (kappa × 100) for NAF			
NAF:	h_{src},D_{src}	h_{src},D_{tg}	h_{tg}, D_{tg}
A3 → *AE*:	69.33 ± 0.21	67.46 ± 5.38	84.07 ± 4.76
A3 → *MB*:	69.33 ± 0.21	10.26 ± 1.13	19.30 ± 1.78
AE → *MB*:	94.39 ± 0.49	11.96 ± 1.15	13.14 ± 0.63
MB → *AE*:	41.69 ± 1.60	65.27 ± 3.03	78.88 ± 2.25
AE → *A3*:	94.39 ± 0.49	29.67 ± 1.09	36.68 ± 2.60

Table 3. Performance of $A3 \rightarrow AE$ and $AE \rightarrow A3$ for Resp A, Sp02, and Resp C

SICO h_{tg} Performance (kappa × 100) for different sensors			
	Resp A	Resp C	Sp02
$A3 \rightarrow AE$: h_{src},D_{src}:	66.74 ± 0.38	66.91 ± 0.30	71.84 ± 0.67
$A3 \rightarrow AE$: h_{src},D_{tg}:	78.95 ± 1.92	57.23 ± 8.38	61.23 ± 4.44
$A3 \rightarrow AE$: h_{tg},D_{tg}:	80.68 ± 0.84	81.47 ± 1.12	74.23 ± 2.07
$AE \rightarrow A3$: h_{src},D_{src}:	92.31 ± 0.50	90.97 ± 0.45	88.93 ± 0.36
$AE \rightarrow A3$: h_{src},D_{tg}:	27.35 ± 1.04	23.07 ± 1.89	−0.32 ± 0.00
$AE \rightarrow A3$: h_{tg},D_{tg}:	48.47 ± 1.20	26.00 ± 0.53	19.37 ± 1.58

very poorly. It is important to mention that we get significantly better results with *SICO* for $A3 \rightarrow MB$ than for $AE \rightarrow MB$. In summary, we identify D_{src} data quality and variation as two important factors which affect the performance of h_{src} and h_{tg} for D_{tg}.

Generally, h_{tg} performs for all cases better than h_{src} on D_{tg}. As expected from Equation (2), h_{src} plays a very critical role in the *SICO* process, and we cannot get very good results if the performance of h_{src} is initially very low. We discuss this characteristic in more detail in Section 6. Finally, another observation is the very large standard error for all cases for h_{src} on D_{tg}, and to a lesser extent for *SICO*. The results for h_{src} were not stable among the different iterations of the experiment, e.g., for $A3 \rightarrow AE$ the range of kappa × 100 values is 54.3–81.9. In all cases, h_{tg} consistently outperformed h_{src}, and it additionally provided a stabilizing effect, as the results for h_{tg} did not vary as much -with the exception of $AE \rightarrow A3$.

5.2.1 Other Sensors. We repeat the experiment for the $A3 \rightarrow AE$ combination for the additional respiratory sensors which are included in *A*3 and *AE*, i.e., Abdominal Respiration (Resp A), Oxygen Saturation (Sp02), and Chest Respiration (Resp C). We do not use *MB* for these experiments due to the small number of recordings per-sensor and the already very low performance it yields for all experiments even with the NAF signal, for which we have much more data in comparison to the other signals. The other parameters are the same as in the previous experiments.

The results are found in Table 3. We notice that h_{tg} significantly outperforms h_{src} on D_{tg} (with the exception of Resp A). As before, we have with $A3 \rightarrow AE$ a very large standard error (big variation) for h_{src}, and *SICO* seems to have a stabilizing effect on the variation. For these experiments, this phenomenon is more pronounced than for the NAF experiments. Again, we observe for Resp A: $A3 \rightarrow AE$ the same interesting pattern that we observed for NAF: $MB \rightarrow AE$, i.e., that h_{src} has a higher performance on the target test set than the source test set. We hypothesize that this happens for similar reasons as before, i.e., due to the better quality and potential homogeneity of *AE* relative to *A*3. This hypothesis is strengthened by the fact that for the $A3 \rightarrow AE$ adaptation all sensor signals perform relatively well (with the exception of Resp C). Another interesting characteristic is that Sp02 which has the best performance for *A*3, adapts much worse to the new domain, i.e., *AE*, than Resp A.

In the $AE \rightarrow A3$ experiments h_{tg} outperforms h_{src} again for all cases. This time the variation for h_{src} in D_{src} is smaller than for h_{tg}. This could potentially again be attributed to the larger data variety in A3, which can make DA more stable regardless of the initialization, and training randomness for h_{src}. Interestingly, h_{src} has seemingly not generalized in the $SpO2 : AE \rightarrow A3$ experiments. Intuitively, this leads us to assume that h_{tg} should also fail. However, the results show that to some extent that h_{tg} is still able to learn. For a h_{src} trained with *AE* on Sp02 and tested on *A*3 Sp02, we observe that for the train set, the specificity between the pseudo-labels and the true labels is 0.952 and sensitivity is 0.08. For the same h_{src}, for D_{CL0}, specificity is 0.67, sensitivity 0.33. We notice that we achieve already with the first h_{CL1} a kappa (×100) value of 16. We hypothesize, that the performance improvement occurs due to the recognition from h_{CL1} of unique characteristics of each class. This could be a result of the better balancing between the correct data from each class, together with the potentially

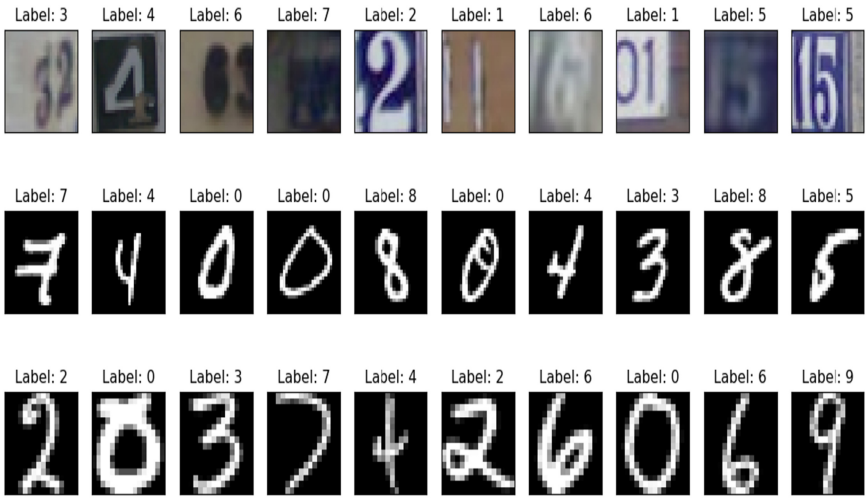


Fig. 7. Examples of different datapoints from the Digit datasets: First row: examples from SVHN. Second row: examples from MNIST. Third row: examples from USPS.

uninformative nature of the wrong class data. Due to this uninformative nature, h_{CL1} does not generalize as much towards a wrong pattern. Intuitively, this means that h_{src} learned a useful structure of the feature space for both classes. Due to the better balancing of D_{CL0}, the "hidden" structure learned for the minority class is uncovered for the classifier trained on D_{CL0}, i.e., h_{CL1}, allowing it to generalize better to new data.

5.3 Comparison with Other Domain Adaptation Approaches: Digit Classification

In this experiment we compare *SICO* with other well-established DA approaches to investigate algorithm's relative performance capability and establish its viability. Figure 7 shows examples from the different digit datasets (before pre-processing). Regarding the digit datasets, the differences between the datasets are more apparent, in comparison to the sleep apnea data. For example some obvious differences we can identify from Figure 7 are: (1) M and U are handwritten digits, whereas S are artificially made, (2) S in many cases contains additional numbers in the image, and (3) digits from U seem to capture a larger part of the image in relation to digits from M.

We use the same architecture for all classifiers (i.e., $h_{src}, h_{CL1}...h_{CLn}, h_{tg}$) in all Digit classification experiments. This means that for any given instance of the algorithm we have only one model in memory. We use a CNN with a similar but wider architecture to LeNet-5 with more weights per layer (see Table 1), and one more fully-connected layer and Convolution layers. We chose this architecture as this is a well-established simple model that is very often used for digit classification. Note that we do not use Max Pooling on the third Convolutional layer for the $S \rightarrow M$, and $M \leftrightarrow U$ experiments. We use more weights to potentially compensate for the larger datasets (i.e., SVHN) and relu activations to all layers, softmax output, and dropout in the fully-connected layers. Additionally, we perform Batch Normalization for the network of the $S \rightarrow M$ experiment. For all experiments, we use a batch size of 128, learning rate of 0.001, and the Adam optimizer [57]. All differences in results (mean accuracies) between h_{src} on the D_{tg} test set and h_{tg} on the D_{tg} test set are statistically significant based on the one-tailed paired t-test (for $p = 0.05$). We use this test as an indication of the importance of the improvement in performance that we observe for h_{tg} compared to h_{src} on the D_{tg} test set.

We trained h_{src} for $M \rightarrow U$ and $U \rightarrow M$ for 4,688 batches (i.e., 600K datapoints). For $S \rightarrow M$, we trained h_{src} for 7,812 batches (i.e., 10^6 datapoints), since h_{src} does not converge with only 4,688 batches when trained on S. When D_{tg} is either M or U we use the fixed number of datapoints criterion $C\{h\}$ as explained in Section 5.1, with $m = 200$ datapoints per-class to construct D_{CL0}, and $m = 100$ datapoints per-class for all subsequent D_{CLi}. The

Table 4. *SICO* h_{tg} Performance on Digit Classification DA (Accuracy)

	$M \rightarrow U$	$U \rightarrow M$	$S \rightarrow M$
(SICO): h_{src},D_{src}:	99.12 ± 0.01	96.62 ± 0.16	90.55 ± 0.40
(DANN, [14]): Source Only:	-	-	54.90
(DANN, [14]): DANN:	-	-	73.85
(Assymetric, [24]): Source Only:	-	-	70.1
(Assymetric, [24]): Assymetric:	-	-	86.20
(CoGAN, [17]): CoGAN:	91.20	89.10	-
(SICO): h_{src}, D_{tg} (Source Only):	79.83 ± 0.51	69.58 ± 2.00	65.42 ± 0.85
(SICO): h_{tg}, D_{tg}:	89.32 ± 0.70	90.88 ± 0.69	88.19 ± 0.74

We include comparison with works that use similar convolutional architectures as ours (i.e., small networks based on LeNet).

exception to this configuration is $M \rightarrow U$, where we choose a fixed percentage instead of a fixed number of data per-class, and use for each h_{CLi} 33% of the data. The algorithm stops when we do not have any more unlabeled data in D_{tg}. For more details please refer to Appendix A.

For our experimental comparisons we use two well-established adversarial methods, namely **Coupled Generative adversarial networks** [17] **(CoGAN)** and **Domain adversarial training** [14] **(DANN)**, and one method which uses ensemble based pseudo-labelling, namely Assymetric tri-training [24] (assymetric). All related works utilize in their respective experiments similar models to ours (i.e., convolutional-fully-connected architectures).

Table 4 presents the classification performance of the three combinations for h_{src} on D_{src}, and h_{tg} and h_{src} on D_{tg}. From Table 4 we notice that h_{tg} outperforms h_{src} for all $D_{src} \rightarrow D_{tg}$ cases. This could potentially be attributed to the use of domain specific knowledge (in the form of training data from D_{src}) to train all subsequent h_{CLi} and the h_{tg} , with a given confidence defined by the used criterion. Notice the steep drop of h_{src} in all cases from D_{src} to D_{tg}, which are expected due to the domain differences. We observe the largest drop for the case of $S \rightarrow M$(24.61%). The largest improvement is observed again for the case of $S \rightarrow M$ (22.77%).

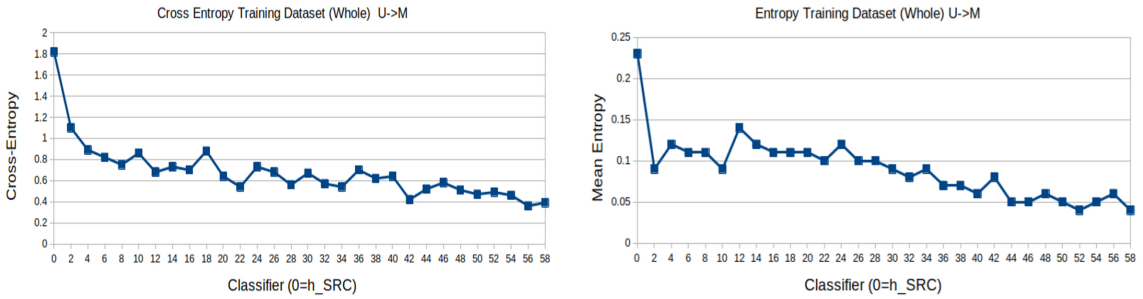
In Table 4 we also include the results from the related works from Unsupervised DA. Despite the additional constraints imposed on the scenario we investigate, i.e., source-free unsupervised DA, *SICO* manages to achieve very good performances for the digit classification tasks which are comparable or superior to the other DA methods we investigated.

5.4 Training Analysis

In *SICO*, we train sequentially n classifiers, each one to a high-confidence sub-region of the training dataset, which is a superset of the region that the previous classifier has been trained. Thus, assuming that the accumulative error $\Delta_i(h_{tg}; h_{src}, h_{CL1}...h_{CL(i-1)})$ for every classifier h_{CLi} in the sequence does not get too large, we expect that the cross-entropy loss with the true training labels will become smaller, because each classifier learns with an increasing number of datapoints from the training dataset.

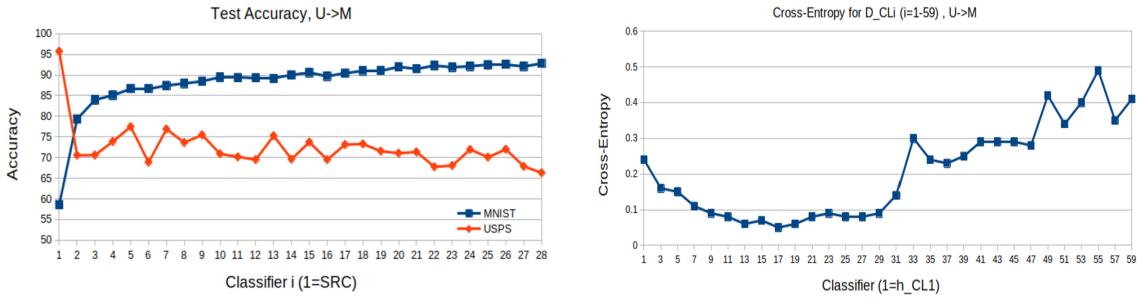
Figure 8(a) presents the training dataset cross-entropy loss for $U \rightarrow M$, for $m = 500$ per-class for h_{src} and $m = 200$ per-class for the subsequent h_{CLi}, based on the fixed number of datapoints criterion. As expected, the loss decreases as the datasets D_{CLi} become larger and the algorithm proceeds. This is mapped also in the performance on new data as shown in Figure 9(a), which depicts the test set performance in terms of accuracy of M for all different h_{CLi}. In this Figure, we also include the performance of the test set of U (orange) for completion. The performance of U is as expected degrading as the algorithm proceeds.

Additionally, Figure 9(b) shows the mean cross-entropy with the real labels for D_{CLi} for all i. We observe that the error initially becomes smaller, which means that the new regions that are included in D_{CLi} to form $D_{CL(i+1)}$ yield better performance than the previous ones. However, after some iterations this does not hold, and the performance with the new $D_{CL(i+1)}$ degrades in relation to D_{CLi}. This means that the mean Δ_i becomes larger.



(a) (b)

Fig. 8. (a) Mean cross-entropy loss with the real labels, and (b) Mean entropy for the whole target training dataset.



(a) (b)

Fig. 9. (a) Test accuracy (hypothetical) for all h_{CLi}, (b) Mean cross-entropy loss for each dataset $D_{CL(i-1)}$ for the respective h_{CLi}. We show the results for every second classifier h_{CLi} (dots) for better readability. All figures were obtained on the same run.

We hypothesize that this behavior could be attributed to the misalignment between the left out data that later iterations represent and the initial high confidence region of h_{src}.

We expect that for increasing values of i the average entropy of h_{CLi} becomes smaller. Since each new h_{CLi} uses a larger part of D_{tg} and trains on hard labels from $h_{CL(i-1)}$, it will be more "confident" for a larger part of D_{tg}. Figure 8(b) showcases this phenomenon. Furthermore, h_{CLi} has been trained with more data and thus can potentially generalize better to new data.

6 DISCUSSION AND LIMITATIONS

From the above evaluation, we observe that the success of *SICO* on D_{tg} depends on how well the initial h_{src} can generalize on D_{tg}. This observation occurs directly from Equation (2), since the error of h_{src}, recursively propagates for all Δ_i which constitute the total Δ_n. Thus, if h_{src} does not perform well on D_{tg} this has a direct repeated negative impact on the whole *SICO* algorithm. Interestingly, even for the cases for which h_{src} performs really bad on D_{tg} (mainly for the cases which include *MB* and *A*3 as D_{tg}) h_{tg} is still able to outperform h_{src}. This potentially happens due to the fact that we expect better performance for high-confidence regions than the average performance, assuming that we trust h_{src} adequately. This means that a smaller error propagates through the algorithm, than for the case in which we have a lower confidence region. This insight is one of the main inspirations for this work.

However, if we use too few data points as our confidence region, then the classifier trained on this region will not be able to generalize well enough to new data, and will have higher generalization error for the new regions.

Thus, there exists a trade-off between confidence and trust on the one hand, and generalization capability on the other hand. This needs to be taken into account to determine how strict the criterion $C\{h\}$ should be, as this decides how many datapoints a new D_{CLi} will have.

A natural question about *SICO* is why should we use a new model in each step (reinitialize), and not use the same model. We made this design choice since we want the subsequent models constructed by *SICO* to focus on the features of the new domain. As such, the use of a new model at each step, serves as a step towards the minimization of domain-specific biases carried by h_{src} (or its labelling) as *SICO* progresses.

We identify two main limitations with the current design of SICO. First, the fact that we need to identify appropriate criterion C in order to decide which datapoints to choose to include in each new D_{CLi}. This choice can be problematic as it adds additional complexity to the learning task and increases the difficulty of the *a priori* algorithmic design. A potential extension to *SICO*, which could resolve the aforementioned issue, is to apply a probabilistic criterion, e.g., based on the max class probability of each datapoint, instead of using a "hard" criterion C as a threshold of acceptance for a given datapoint. This is a more natural approach that better captures the probabilistic nature of density estimation that the classifiers perform, and importantly removes the need of tuning C. As such, we hypothesize that it can yield even better results than the original hard acceptance/rejection method. However, in preliminary experiments (see Section 5.1.1) we were not able to showcase improvement in performance in relation to the original method.

Second, though the use of SICO guarantees that no data will be exchanged between the lab/hospital and the end-user patient, h_{src} is trained on the lab dataset. As such, the use of information leakage attacks against h_{src} can potentially leak information about the lab dataset. To obtain protection from such attacks, we can, e.g., strategically clip and add noise to the gradients of h_{src} during training. With such an application, certain dataset traits could become obfuscated and theoretical guarantees like differential privacy can be obtained. Alternatively, we can adversarially train h_{src} so that certain adversarial defences can be learned to obtain protection against certain information leakage attacks.

7 CONCLUSION

The primary motivation for our work is to enable end-users to create personal classifiers for health applications without labeled data. In particular, we foresee a collaboration in which a host releases a classifier h_{src}, and the end-user (i.e., a patient), uses her data and the classifier to create a new personalized classifier which is adapted to the domain of the end-user. For example, for the sleep apnea case, if we have a person whose datapoints (i.e., one-minute windows) are different due to a variety of potential factors than the datapoints h_{src} has been trained with, we can expect that by applying a suitable method we can get a better performing classifier for the particular individual.

In this work we achieve this by performing *SICO*. *SICO* iteratively adapts classifiers to the new domain based on high-confidence data from the previous classifiers, and without the need of the source data and labels. Based on our scenario, we are more interested in the case of performing DA at the end-user, but obviously *SICO* can also be applied at the host. We apply *SICO* for the case of sleep apnea detection, and use a large real-world clinical dataset for its evaluation. Additionally, we experiment with two open sleep monitoring datasets and the MNIST, SVHN, and USPS datasets for digit classification. By this, we achieve (1) reproducibility, (2) demonstrate the generalizability of *SICO* for another task, and (3) achieve comparability with related work. The results from these experiments show consistently better performance of h_{tg} in comparison to h_{src}, as expected. Depending on the dataset, we get an increase in kappa of up to 0.24. For the task of digit classification DA, *SICO* also achieves a consistently good performance. Despite the additional limitations of the scenario that we investigate, the performance of several well-established related works is lower than the presented results of this paper.

As a next step, we are interested in investigating how well can the technique be applied if the source classifier is trained under differential privacy guarantees. Another interesting application to investigate is the combination of the proposed approach with a style transfer method with the goal of increasing the DA capabilities of the

approach for tasks containing different types of sensors. Furthermore, since the digit classification experiments indicate good generalizability for *SICO*, it would be interesting to apply *SICO* on other image-based health applications (like, e.g., MRI classification on various tasks, or telemedicine applications) in cases where it is appropriate, i.e., existence of a domain shift and inability of data sharing. Finally, assuming that we choose proper criterion C, we hypothesize that we could apply *SICO* to properly adapt in a dataset which is balanced with different class frequencies in relation to the source dataset. This, in the context of condition detection, means that we adapt from patients who do not have many pathological datapoints to patients for which the condition is more expressed.

APPENDICES

A ADDITIONAL DESIGN DECISIONS

In Appendix A, we discuss additional design decisions made during the algorithm.

- **Use of hard or soft labels for labelling of** $D_{CLi}, i = 0..N$: In the original approach we use as labels the one-hot encoding vector based on the maximum argument of the output class probabilities of the classifier which is doing the labelling for the particular datapoint. Thus, each new classifier is trained on one-hot encoding labels. Alternatively, we can utilize directly the output class probabilities to act as labels and for the training of each new classifier. During our experiments, we experimented with this approach. In almost all cases it yielded worse results than the use of hard labels. As a result we use hard labelling in our final experiments. The only exception is the *MB* dataset, for which we hypothesize that due to the poor labelling quality, there is a stronger discrepancy between the true conditional distribution and the hard labelling. This would also result in the optimal (Bayes) classifier having a very high true risk because of the poor labelling. Since we have a finite amount of data, using hard labels could be misrepresentative of the previous classifiers' class probabilities.
- **Training Iterations for** h_{CLi}**:** Depending on the sizes of the different datasets, we train for a minimum of 10 (*A*3) epochs to a maximum of 50 epochs (*MB*, *AE*).

B ACCURACY, SPECIFICITY, AND SENSITIVITY FOR APNEA DETECTION EXPERIMENTS

Appendix B shows the Accuracy, Sensitivity, and Specificity for the Apnea detection DA experiments. A characteristic which was not identifiable from the kappa values is the balancing effect SICO has on specificity and sensitivity. When using h_{src}, in many cases we obtain high specificity at the expense of high sensitivity (e.g., NAF:*A*3 $\rightarrow$ *MB*, NAF:*AE* $\rightarrow$ *MB*). We observe that for h_{tg} this effect is minimized. This means that h_{tg} obtains increased sensitivity compared to h_{src}. This is a positive characteristic since sensitivity (i.e., the percentage of the correctly detected apneic minutes) plays a crucial role in the context of sleep apnea detection as low sensitivity can lead to a false negative diagnosis, making the system inherently untrustworthy.

Table B.1.

$SICO$ h_{tg} Accuracy for NAF			
NAF:	h_{src},D_{src}	h_{src},D_{tg}	h_{tg}, D_{tg}
$A3 \rightarrow AE$:	84.75 ± 0.11	84.37 ± 5.62	92.16 ± 2.18
$A3 \rightarrow MB$:	84.75 ± 0.11	54.72 ± 0.58	59.56 ± 0.89
$AE \rightarrow MB$:	97.32 ± 0.24	55.88 ± 0.57	56.60 ± 0.33
$MB \rightarrow AE$:	70.88 ± 0.79	82.90 ± 1.54	89.60 ± 1.13
$AE \rightarrow A3$:	97.32 ± 0.24	64.43 ± 0.64	68.71 ± 1.00
$SICO$ h_{tg} Sensitivity for NAF			
NAF:	h_{src},D_{src}	h_{src},D_{tg}	h_{tg}, D_{tg}
$A3 \rightarrow AE$:	85.82 ± 0.78	77.82 ± 0.50	94.99 ± 1.54
$A3 \rightarrow MB$:	85.82 ± 0.66	26.29 ± 2.48	52.12 ± 3.54
$AE \rightarrow MB$:	96.08 ± 0.24	48.50 ± 0.74	58.34 ± 1.77
$MB \rightarrow AE$:	73.38 ± 0.87	86.66 ± 2,23	95.19 ± 0.76
$AE \rightarrow A3$:	96.08 ± 0.24	56.78 ± 2.32	74.32 ± 0.96
$SICO$ h_{tg} Specificity for NAF			
NAF:	h_{src},D_{src}	h_{src},D_{tg}	h_{tg}, D_{tg}
$A3 \rightarrow AE$:	83.50 ± 0.73	88.93 ± 4.59	90.29 ± 4.24
$A3 \rightarrow MB$:	83.50 ± 0.73	84.06 ± 1.95	67.23 ± 3.71
$AE \rightarrow MB$:	98.13 ± 0.31	63.49 ± 1.16	54.79 ± 1.37
$MB \rightarrow AE$:	68.28 ± 1.76	80.42 ± 2.29	85.92 ± 1.55
$AE \rightarrow A3$:	98.13 ± 0.31	73.39 ± 1.50	62.14 ± 1.21

Table B.2.

$SICO$ h_{tg} Accuracy for different sensors			
	Resp A	Resp C	Sp02
$A3 \rightarrow AE$: h_{src},D_{src}:	83.51 ± 1.92	83.57 ± 0.16	85.97 ± 0.30
$A3 \rightarrow AE$: h_{src},D_{tg}:	89.74 ± 0.81	77.68 ± 4.89	80.12 ± 2.4
$A3 \rightarrow AE$: h_{tg}, D_{tg}:	90.42 ± 0.43	90.88 ± 0.56	87.44 ± 1.01
$AE \rightarrow A3$: h_{src},D_{src}:	96.32 ± 0.24	95.66 ± 0.02	94.76 ± 0.17
$AE \rightarrow A3$: h_{src},D_{tg}:	61.79 ± 0.59	59.49 ± 1.08	47.32 ± 1.26
$AE \rightarrow A3$: h_{tg}, D_{tg}:	74.44 ± 0.58	63.20 ± 0.23	60.34 ± 0.78
$SICO$ h_{tg} Sensitivity for different sensors			
	Resp A	Resp C	Sp02
$A3 \rightarrow AE$: h_{src},D_{src}:	86.3 ± 0.51	85.22 ± 0.66	85.40 ± 1.05
$A3 \rightarrow AE$: h_{src},D_{tg}:	93.83 ± 5.15	92.16 ± 4.44	94.64 ± 0.53
$A3 \rightarrow AE$: h_{tg}, D_{tg}:	98.48 ± 0.07	96.91 ± 0.75	89.44 ± 0.79
$AE \rightarrow A3$: h_{src},D_{src}:	95.85 ± 0.43	97.46 ± 0.52	97.48 ± 0.13
$AE \rightarrow A3$: h_{src},D_{tg}:	30.76 ± 1.27	27.92 ± 2.39	9.53 ± 1.29
$AE \rightarrow A3$: h_{tg}, D_{tg}:	77.46 ± 0.54	65.51 ± 0.70	69.16 ± 0.97
$SICO$ h_{tg} Specificity for different sensors			
	Resp A	Resp C	Sp02
$A3 \rightarrow AE$: h_{src},D_{src}:	80.55 ± 0.60	81.63 ± 0.46	86.63 ± 1.86
$A3 \rightarrow AE$: h_{src},D_{tg}:	87.04 ± 2.56	68.17 ± 9.66	70.59 ± 3.69
$A3 \rightarrow AE$: h_{tg}, D_{tg}:	85.12 ± .0.73	86.91 ± 0.91	86.12 ± 1.53
$AE \rightarrow A3$: h_{src},D_{src}:	96.61 ± 0.24	94.43 ± 0.46	97.48 ± 0.13
$AE \rightarrow A3$: h_{src},D_{tg}:	98.17 ± 0.20	96.49 ± 0.48	90.40 ± 1.16
$AE \rightarrow A3$: h_{tg}, D_{tg}:	70.91 ± 0.87	60.50 ± 0.54	50.00 ± 1.06

REFERENCES

[1] 2020. Nox T3. (2020). https://www.resmed.no/helsepersonell/diagnostikk/nox-t3/. [Online; accessed 12-11-2020].

[2] Stein Kristiansen, Konstantinos Nikolaidis, Thomas Plagemann, Vera Goebel, Gunn Marit Traaen, Britt Overland, Lars Aakeroy, Tove Elizabeth Hunt, Jan Pal Lonnechen, Sigurd Steinshamn, Christina Holt Bendz, Ole Gunnar Anfinsen, Lars Gullestad, and Harriet Akre. 2020. Machine learning for sleep Apnea detection with unattended sleep monitoring at home. *ACM Transactions on Computing for Healthcare* Accepted for publication (December 2020).

[3] Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2009), 1345–1359.

[4] Yaroslav Ganin and Victor Lempitsky. 2014. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495* (2014).

[5] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G. Hauptmann. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4893–4902.

[6] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*. 137–144.

[7] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 41–48.

[8] Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. *arXiv preprint arXiv:1904.03626* (2019).

[9] Gabriela Csurka. 2017. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374* (2017).

[10] Mei Wang and Weihong Deng. 2018. Deep visual domain adaptation: A survey. *Neurocomputing* 312 (2018), 135–153.

[11] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. 2018. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*. 1640–1650.

[12] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7167–7176.

[13] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17, 1 (2016), 2096–2030.

[15] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. 2017. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*. 700–708.

[16] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2017. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2208–2217.

[17] Ming-Yu Liu and Oncel Tuzel. 2016. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems*. 469–477.

[18] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. 2017. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213* (2017).

[19] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*.

[20] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems*. 343–351.

[21] Philip Haeusser, Thomas Frerix, Alexander Mordvintsev, and Daniel Cremers. 2017. Associative domain adaptation. In *The IEEE International Conference on Computer Vision (ICCV)*.

[22] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. 2017. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[23] Xu Zhang, Felix Xinnan Yu, Shih-Fu Chang, and Shengjin Wang. 2015. Deep transfer network: Unsupervised domain adaptation. *arXiv preprint arXiv:1503.00591* (2015).

[24] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2988–2997.

[25] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2016. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 136–144. http://papers.nips.cc/paper/6110-unsupervised-domain-adaptation-with-residual-transfer-networks.pdf.

[26] Qian Wang and Toby Breckon. 2020. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6243–6250.

[27] Yang Zou, Zhiding Yu, Xiaofeng Liu, B. V. K. Kumar, and Jinsong Wang. 2019. Confidence regularized self-training. In *Proceedings of the IEEE International Conference on Computer Vision*. 5982–5991.

[28] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. 2018. Incremental adversarial domain adaptation for continually changing environments. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1–9.

[29] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. 2016. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779* (2016).

[30] Shiqi Yang, Yaxing Wang, Joost van de Weijer, Luis Herranz, and Shangling Jui. 2021. Generalized source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8978–8987.

[31] Wei Zhang, Gaoliang Peng, Chuanhao Li, Yuanhang Chen, and Zhujun Zhang. 2017. A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals. *Sensors* 17, 2 (2017), 425.

[32] Haifeng Xia, Handong Zhao, and Zhengming Ding. 2021. Adaptive adversarial network for source-free domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9010–9019.

[33] Jogendra Nath Kundu, Naveen Venkat, R. Venkatesh Babu, et al. 2020. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4544–4553.
[34] Youngeun Kim, Donghyeon Cho, Kyeongtak Han, Priyadarshini Panda, and Sungeun Hong. 2021. Domain adaptation without source data. *IEEE Transactions on Artificial Intelligence* 2, 6 (2021), 508–518.
[35] Boris Chidlovskii, Stephane Clinchant, and Gabriela Csurka. 2016. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 451–460.
[36] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683* (2012).
[37] Sanjay Purushotham, Wilka Carvalho, Tanachat Nilanon, and Yan Liu. 2016. Variational recurrent adversarial deep domain adaptation. (2016).
[38] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems*. 2980–2988.
[39] Witali Aswolinskiy and Barbara Hammer. 2017. Unsupervised transfer learning for time series via self-predictive modelling-first results. In *Proceedings of the Workshop on New Challenges in Neural Computation (NC2)*, Vol. 3.
[40] Xin Chai, Qisong Wang, Yongping Zhao, Yongqiang Li, Dan Liu, Xin Liu, and Ou Bai. 2017. A fast, efficient domain adaptation technique for cross-domain electroencephalography (EEG)-based emotion recognition. *Sensors* 17, 5 (2017), 1014.
[41] Annamalai Natarajan, Gustavo Angarita, Edward Gaiser, Robert Malison, Deepak Ganesan, and Benjamin M. Marlin. 2016. Domain adaptation methods for improving lab-to-field generalization of cocaine detection using wearable ECG. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 875–885.
[42] Claudio Persello and Lorenzo Bruzzone. 2012. Active learning for domain adaptation in the supervised classification of remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing* 50, 11 (2012), 4468–4483.
[43] Claudio Persello. 2012. Interactive domain adaptation for the classification of remote sensing images using active learning. *IEEE Geoscience and Remote Sensing Letters* 10, 4 (2012), 736–740.
[44] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2018. Transfer learning for time series classification. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 1367–1376.
[45] Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar, and Jinsong Wang. 2018. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 289–305.
[46] Yang Zhang, Philip David, and Boqing Gong. 2017. Curriculum domain adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE International Conference on Computer Vision*. 2020–2030.
[47] Thomas Penzel, George B. Moody, Roger G. Mark, Ary L. Goldberger, and J. Hermann Peter. 2000. The Apnea-ECG database. In *Computers in Cardiology 2000. Vol. 27 (Cat. 00CH37163)*. IEEE, 255–258.
[48] Yuhei Ichimaru and G. B. Moody. 1999. Development of the polysomnographic database on CD-ROM. *Psychiatry and Clinical Neurosciences* 53, 2 (1999), 175–177.
[49] Stein Kristiansen, Mari Sønsteby Hugaas, Vera Goebel, Thomas Plagemann, Konstantinos Nikolaidis, and Knut Liestøl. 2018. Data mining for patient friendly apnea detection. *IEEE Access* 6 (2018), 74598–74615.
[50] Gunn Marit Traaen, Lars Aakerøy, et al. 2019. Treatment of sleep apnea in patients with paroxysmal atrial fibrillation: Design and rationale of a randomized controlled trial. *Scandinavian Cardiovascular Journal* 52:6, pp. 372-377 (January 2019), 1–20.
[51] Gunn Marit Traaen, Britt Øverland, Lars Aakerøy, T. E. Hunt, Christina Bendz, L. Sande, Svend Aakhus, H. Zaré, S. Steinshamn, Ole-Gunnar Anfinsen, et al. 2020. Prevalence, risk factors, and type of sleep apnea in patients with paroxysmal atrial fibrillation. *IJC Heart & Vasculature* 26 (2020), 100447.
[52] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[53] Jonathan J. Hull. 1994. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16, 5 (1994), 550–554.
[54] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. 2011. Reading digits in natural images with unsupervised feature learning.
[55] Konstantinos Nikolaidis, Thomas Plagemann, Stein Kristiansen, Vera Goebel, and Mohan Kankanhalli. 2021. Using under-trained deep ensembles to learn under extreme label noise: A case study for sleep apnea detection. *IEEE Access* 9 (2021), 45919–45934.
[56] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20, 1 (1960), 37–46.
[57] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Received 17 December 2020; revised 16 May 2022; accepted 7 August 2022