CALDA: Improving Multi-Source Time Series Domain Adaptation with Contrastive Adversarial Learning

Garrett Wilson, Janardhan Rao Doppa, *Senior Member, IEEE* and Diane J. Cook, *Fellow, IEEE*

Abstract—Unsupervised domain adaptation (UDA) provides a strategy for improving machine learning performance in data-rich (target) domains where ground truth labels are inaccessible but can be found in related (source) domains. In cases where meta-domain information such as label distributions is available, weak supervision can further boost performance. We propose a novel framework, CALDA, to tackle these two problems. CALDA synergistically combines the principles of contrastive learning and adversarial learning to robustly support multi-source UDA (MS-UDA) for time series data. Similar to prior methods, CALDA utilizes adversarial learning to align source and target feature representations. Unlike prior approaches, CALDA additionally leverages cross-source label information across domains. CALDA pulls examples with the same label close to each other, while pushing apart examples with different labels, reshaping the space through contrastive learning. Unlike prior contrastive adaptation methods, CALDA requires neither data augmentation nor pseudo labeling, which may be more challenging for time series. We empirically validate our proposed approach. Based on results from human activity recognition, electromyography, and synthetic datasets, we find utilizing cross-source information improves performance over prior time series and contrastive methods. Weak supervision further improves performance, even in the presence of noise, allowing CALDA to offer generalizable strategies for MS-UDA.

Index Terms—Transfer Learning, Domain Adaptation, Time Series, Weak Supervision, Adversarial Training, Contrastive Learning.

1 Introduction

Unsupervised domain adaptation can leverage labeled data from past (source) machine learning tasks when only unlabeled data are available for a new related (target) task [1]. As an example, when learning a model to recognize a person's activities from time-series sensor data, standard learning algorithms will face an obstacle when person A does not provide ground-truth activity labels for their data. If persons B through G are willing to provide these labels, then Multi-Source Unsupervised Domain Adaptation (MS-UDA) can create a model for the target person based on labeled data from the source persons. When performing adaptation, MS-UDA must bridge a domain gap. In our example, such a gap exists because of human variability in how activities are performed. Meta-domain information may exist for person A that is easier to collect and can improve the situation through weak supervision [2], such as self-reported frequencies for each activity (e.g., "I sleep 8 hours each night").

In this article, we develop a framework that can construct a model for time-series MS-UDA. Our proposed approach leverages labeled data from one or more source domains, unlabeled data from a target domain, and optional target class distribution. Very few domain adaptation methods handle time series [1], [2], [3] and even fewer facilitate multiple source domains or weak supervision [2]. We postulate adapting multiple time-series domains is particularly critical because many time-series problems involve multiple domains (in our example, multiple people) [4], [5]. Furthermore, we posit that existing approaches to adaptation do not make effective use of meta-domain information about the target, yet additional gains may stem from leveraging this information via weak supervision. We propose a novel framework for time series MS-UDA. This framework, called CALDA (**C**ontrastive **A**dversarial **L**earning for Multi-Source Time Series **D**omain **A**daptation), improves unsupervised domain adaptation through adversarial training, contrastive learning, and weak supervision without relying heavily on data augmentation or pseudo labeling like prior image-based methods.

First, CALDA guides adaptation through multi-source domain-adversarial training [2], [6]. CALDA trains a multi-class domain classifier to correctly predict the original domain for an example's feature representation while simultaneously training a feature extractor to incorrectly predict the example's domain. Through this two-player game, the feature extractor produces domain-invariant features. A task classifier trained on this domain-invariant representation can potentially transfer its model to a new domain because the target features match those seen during training, thus bridging the domain gap. CALDA utilizes adversarial training to align the feature-level distributions between domains and utilizes contrastive learning to leverage cross-source label information for improving accuracy on the target domain.

Second, CALDA enhances MS-UDA through contrastive learning across source domains. Contrastive learning moves the representations of similar examples close together and dissimilar examples far apart. While this technique yielded performance gains for self-supervised [7], [8] and traditional supervised learning [9], the method is unexplored for multi-source time series domain adaptation. We propose to introduce the contrastive learning principle within CALDA. In this context, we will investigate three design decisions. First, we will analyze methods to select pairs of examples from source domains. Second, we will determine whether it is beneficial to pseudo-label and include target data as a contrastive learning domain despite the likelihood of incorrect pseudo-labels due to the large domain shifts in time series data. Third, we will

• *The authors are with the School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA, 99164. E-mail: {garrett.wilson,jana.doppa,djcook}@wsu.edu*

Manuscript received April 19, 2005; revised August 26, 2015.

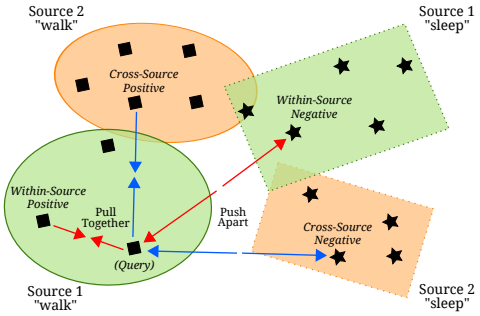

Fig. 1. In the space $z = Z(F(x))$, an illustration of the difference between cross-source (blue) and within-source (red) pairs for contrastive learning. Any-source uses cross-source and within-source pairs.

assess whether to randomly select contrastive learning examples or utilize the varying complexities of different domains to select the most challenging (i.e., hard) examples.

In the case of the first decision, we hypothesize that utilizing cross-source information to select example pairs can improve transfer. Prior MS-UDA methods [2] ignore cross-source label information that can provide vital insights into how classes vary among domains (i.e., which aspects of the data truly indicate a different class label versus the same label from different domains). Utilizing this information can potentially improve transfer to the target domain. If our activity recognition class labels include "walk" and "sleep", we want the feature representations of two different walking people to be close, but the representations to be far apart for one walking person and one sleeping person. We use CALDA to investigate whether such cross-source information aids in transfer by explicitly making use of labeled data from each source domain as well as the differences in data distributions among the source domains. Furthermore, we compare this approach with a different instantiation of our framework that utilizes only labels within each source domain, thus explicitly ignoring cross-domain information. The differences between these approaches are illustrated in Figure 1.

For the second decision, we propose to utilize contrastive learning only across source domains rather than include pseudo-labeled target data that run the risk of being incorrectly labeled. Prior single-source domain adaptation methods have integrated contrastive losses [10], [11]. Because they utilize a single source domain with the target, they rely on pseudo-labeling the target domain data, creating difficult challenges when faced with large domain gaps [12]. Because CALDA employs more than one source domain, we can leverage a contrastive loss between source domains, thereby avoiding incorrectly pseudo-labeled target data.

For the third decision, we note that prior contrastive learning work has found selecting hard examples to yield improved performance [13], [14]. However, recent theory postulates that hard examples do not need to be singled out - such examples already intrinsically yield a greater contribution to the contrastive loss than less-challenging examples [9]. We hypothesize that both random and hard sampling may offer improvements for multi-source domain adaptation, thus we evaluate both within CALDA.

CALDA integrates all of these components. As in prior work [2], [6], we utilize a domain adversary that aligns feature representations across domains, yielding a domain-invariant feature extractor. Unlike prior approaches, we further utilize contrastive learning to pull examples from different source domains together in the feature space that have the same label while pushing apart examples from different domains that have different labels, though the choice of examples to pull and push depends on the three design decisions.

The key contribution of this paper is the development and evaluation of the CALDA framework for time-series MS-UDA. Specific contributions include:

- We improve upon existing time-series MS-UDA by leveraging cross-source domain labels via contrastive learning without requiring data augmentation or pseudo labeling.
- We incorporate multiple contrastive learning strategies into CALDA to analyze the impact of design choices.
- We offer an approach to time series MS-UDA that makes use of class distribution information where available through weak supervision.
- We demonstrate performance improvements of CALDA over prior work with and without weak supervision. Improvements are shown for synthetic time series data and a variety of real-world time-series human activity recognition and electromyography datasets. These experiments aid in identifying the most promising CALDA instantiations, validating the importance of the adversary and unlabeled target data, and measuring the sensitivity of CALDA to noise within the weak supervision information.[1]

2 Related Work

Here, we discuss related work on domain adaptation and contrastive learning in the context of time-series MS-UDA.

2.1 Domain Adaptation

Single-source domain adaptation methods abound [1], but little work studies multi-source domain adaptation. Zhao et al. [15] developed an adversarial method supporting multiple sources by including a binary domain classifier for each source. Ren et al. [16] align the sources, merging them into one domain that is aligned to the target. Li et al. [17] also unify multiple source domains by updating model parameters to accommodate samples from each source. These approaches, however, do not take advantage of example similarities through contrastive learning to more effectively utilize source labels. Xie et al. [18] propose a scalable method, only requiring one multi-class domain classifier. This approach is similar to the adversarial learning component of our framework. As in our approach, Yadav et al. employ contrastive learning when combining multiple sources. Here, contrastive learning achieves higher intra-class compactness across domains, in the hope of yielding well-separated decision boundaries. Yet, without models that are compatible with time-series, these approaches cannot be used for time series MS-UDA.

Limited research has investigated time-series domain adaptation, although these focus on a single source domain. Numerous approaches introduce explicit linear or nonlinear transformations to align the source and target spaces [19], [20],

1. Code and data is available at: https://github.com/floft/calda.

[21], [22]. Some of these approaches, like CALDA, leverage an adversarial component. Liu et al. [23] introduce a hybrid spectral kernel to characterize the non-stationary elements of the time series. A drawback is they do not make effective use of example similarity through contrastive learning. Purushotham et al. [3] developed a domain-adversarial method for single-source domain adaptation using a variational recurrent neural network (RNN) as the feature extractor. However, in our prior work [2] we found that for both single-source and multi-source domain adaptation using a 1D convolutional neural network outperforms RNNs on a variety of time-series datasets. Thus, we select this network architecture for our experiments.

Domain adaptation has also been studied specifically for electromyography (EMG)-based gesture recognition, mostly utilizing a different type of domain adaptation. Rather than UDA, several methods are proposed to improve supervised domain adaptation performance [24], [25], [26], where some labeled target data are required. One method is developed for unsupervised domain adaptation [27], but they convert the EMG data to images followed by using adaptive batch normalization for domain adaptation [28]. This approach makes the assumption that the domain differences are contained primarily within the network's normalization statistics and not the neural network layer weights. We do not require this assumption in our CALDA framework.

We uniquely incorporate weak supervision into domain adaptation. Weak supervision is inspired by the posterior regularization problem [29], but we consider this for domain adaptation. Along a similar vein, Jiang et al. [30] use a related regularizer for the problem where label proportions are available for the source domains but not the target domain. Hu et al. [31] propose using a different form of weak supervision for human activity recognition from video data, using multiple incomplete or uncertain labels. Pathak et al. [32] develop a method for semantic segmentation using a weaker form of labeling than pixel-level labels. While we study weak supervision in a different context, the benefit of weak supervision in these other contexts in addition to the performance gains observed in our experiments suggests the general applicability of this idea. Additionally, prior weak supervision work fails to address the sensitivity of the weak supervision to noise [2], which we may expect with self-reported data. We analyze weak supervision in the presence of noise.

We select algorithm hyperparameters using cross-validation, but recent work offers alternative methods. Dinu et al. [33] proposed a theoretically-sound method by extending weighted least squares to deep neural networks for time series processing. The method offered by Saito et al. [34] relies on the intuition that a good classifier trained on source domain should embed close-by examples of the same class from the target close to form dense neighborhoods in the learned feature space. Density is measured by computing entropy of the similarity distribution between input examples. Furthermore, You et al. [35] select the best model by embedding feature representations into the validation procedure to obtain a unbiased estimation of the target risk.

2.2 Contrastive Learning

Our framework leverages the contrastive learning principle in addition to the adversarial learning from prior works. Early uses include clustering [36] and dimensionality reduction [37]. More recently, numerous research efforts have incorporated contrastive learning. These methods typically rely on data augmentation to generate positives and negatives but sometimes use labels instead [9]. While such methods yield large gains in other contexts [7], [8], little work has explored contrastive learning for time series domain adaptation.

For single-source adaptation of image domains, prior methods [10], [11] consider a different contrastive loss. In addition to not utilizing an adversary, which we demonstrate is a vital component to our framework, these methods rely on data augmentation and pseudo labeling which may be problematic for time series. Data augmentation is standard and key to the success for many difficult image domain adaptations [38], but is still being explored for time series [39] and is not required for CALDA. Second, prior methods perform contrastive learning on the combined (single) source domain and target domain. This critically depends on accurate target domain pseudo-labeling. However, pseudo-labeling remains a challenging problem that is particularly difficult when faced with the large domain gaps [12], that frequently occur in time series data. Because we support multiple source domains, in CALDA we avoid pseudo-labeling and instead leverage the contrastive loss across source domains. Other contrastive domain adaptation work focus on different problems: labelless transferable representation learning for image data [40] and image adaptation to a sequence of target domains [41]. As with the other prior work, these too rely on data augmentation [40], [41] and typically pseudo labeling [41].

CALDA's final component is hard sampling, which is beneficial in some contrastive learning contexts. Schroff et al. [13] found it necessary for the triplet loss, a special case of contrastive learning [9]. Similarly, Cai et al. [14] found including the top hard negatives to be both necessary and sufficient. While Khosla et al. [9] state that sampling is not necessary since hard examples contribute more to the loss, this impact depends on having a large number of positives and negatives, which may not be optimal on all datasets, as demonstrated in our experimental results.

3 Problem Setup

Here, we formalize Multi-Source Unsupervised Domain Adaptation (MS-UDA) without and with weak supervision.

3.1 Multi-Source Unsupervised Domain Adaptation

MS-UDA assumes that labeled data are available from multiple sources and unlabeled data are available from the target [42] to achieve the goal of creating a model that performs well on the target domain. Formally, given $n > 1$ source domain distributions $\mathcal{D}_{S_i}$ for $i \in \{1, 2, \ldots, n\}$ and a target domain distribution $\mathcal{D}_T$, we draw s_i labeled training examples i.i.d. from each source distribution $\mathcal{D}_{S_i}$ and t_{train} unlabeled training instances i.i.d. from the marginal distribution $\mathcal{D}_T^X$:

$$S_i = \{(\mathbf{x}_j, y_j)\}_{j=1}^{s_i} \sim \mathcal{D}_{S_i} \quad \forall i \in \{1, 2, \ldots, n\} \tag{1}$$

$$T_{train} = \{(\mathbf{x}_j)\}_{j=1}^{t_{train}} \sim \mathcal{D}_T^X \tag{2}$$

Here, each domain is distributed over the space $X \times Y$, where X is the input data space and Y is the label space

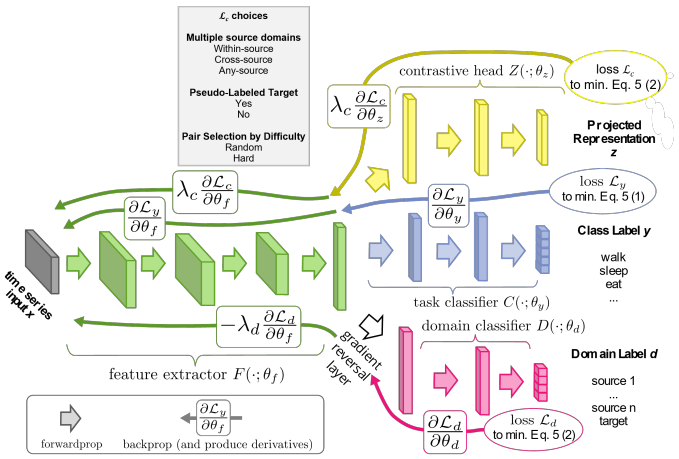

Fig. 2. CALDA incorporates adversarial learning via a domain classifier and contrastive learning via a contrastive loss on an additional contrastive head in the network. Through the CALDA instantiations, we determine how to best utilize contrastive learning for MS-UDA.

$Y = \{1, 2, \ldots, L\}$ for L classification labels. After training a MS-UDA model $f : X \rightarrow Y$ using S_i and T_{train}, we test the model using a holdout set of t_{test} labeled testing examples (input and ground-truth label pairs) drawn i.i.d. from the target distribution $\mathcal{D}_T$:

$$T_{test} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{t_{test}} \sim \mathcal{D}_T \tag{3}$$

In the case of time series domain adaptation, $X = [X^1, X^2, \ldots, X^K]$ for K time series variables or channels. Each variable X^i for $i \in \{1, 2, \ldots, K\}$ consists of a time series $X^i = [x_1, x_2, \ldots, x_H]$ containing a sequence of real values observed at equally-spaced time steps $1, 2, \ldots H$ [43].

3.2 MS-UDA with Weak Supervision

When MS-UDA is guided by weak supervision [2], the target-domain label proportions are additionally available during training, which we can utilize to guide the neural network's representation. Formally, these proportions represent $P(Y = y)$ for the target domain, i.e., the probability p_y that each example will have label $y \in \{1, 2, \ldots, L\}$:

$$Y_{true}(y) = P(Y = y) = p_y \tag{4}$$

4 CALDA Framework

We introduce CALDA, a MS-UDA framework that blends adversarial learning with contrastive learning. First, we motivate CALDA from domain adaptation theory. Second, we describe the key components: source domain error minimization, adversarial learning, and contrastive learning. Finally, we describe framework alternatives to investigate how to best construct the example sets used in contrastive loss.

4.1 Theoretical Motivation

Zhao et al. [15] offer an error bound for multi-source domain adaptation. Given a hypothesis space $\mathcal{H}$ with VC-dimension v, n source domains, empirical risk $\hat{\epsilon}_{S_i}(h)$ of the hypothesis on source domain S_i for $i \in \{1, 2, \ldots, n\}$, empirical source distributions $\hat{\mathcal{D}}_{S_i}$ for $i \in \{1, 2, \ldots, n\}$ generated by m labeled samples from each source domain, empirical target distribution $\hat{\mathcal{D}}_T$ generated by mn unlabeled samples from the target domain, optimal joint hypothesis error λ_α on a mixture of source domains $\sum_{i\in[n]} \alpha_i S_i$, and target domain T (average case if $\alpha_i = 1/n \; \forall i \in \{1, 2, \ldots n\}$), the target classification error bound $\epsilon_T(h)$ with probability at least $1-\delta$ for all $h \in \mathcal{H}$ can be expressed as:

$$\epsilon_T(h) \leq \sum_{i=1}^{n} \alpha_i \Big(\overbrace{\hat{\epsilon}_{S_i}(h)}^{(1)\text{ source errors}} + \underbrace{\frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{D}}_T, \hat{\mathcal{D}}_{S_i})}_{(2)\text{ divergences}} \Big) + \overbrace{\lambda_\alpha}^{(3)\text{ opt. joint hyp.}} + \underbrace{O\left(\sqrt{\frac{1}{nm}\left(\log\frac{1}{\delta} + v\log\frac{nm}{v}\right)}\right)}_{(4)\text{ due to finite samples}} \tag{5}$$

In Equation 5, term (1) is the sum of source domain errors, (2) is the sum of the divergences between each source domain and the target, (3) is the optimal joint hypothesis on the mixture of source domains and the target domain, and (4) addresses the finite sample sizes. Note that the first two terms are the most relevant for informing multi-source domain adaptation methods since they can be optimized. In contrast, given a hypothesis space (e.g., a neural network of a particular size and architecture), (3) is fixed. Similarly, (4) regards finite samples from all domains, which depends on the number of samples and for a given dataset cannot increase.

We introduce CALDA to minimize this error bound as illustrated in Figure 2. First, we train a *Task Classifier* to correctly predict the labeled data from the source domains, thus minimizing (1). To minimize (2), we better align domains based on adversarial learning and contrastive learning. As in prior works, we use feature-level domain invariance via domain adversarial training to align the sources and unlabeled data from the target domain. We train a *Domain Classifier* to predict which domain originated a representation while simultaneously training the *Feature Extractor* to generate domain-invariant representations that fool the Domain Classifier. Additionally, we propose a supervised contrastive loss to align the representations of same-label examples among the multiple source domains. The new loss aids in determining which aspects of the data correspond to differences in the class label (the primary concern) versus differences in the domain (e.g., person) where the data originated (which can be ignored). The new loss definition leverages both the labeled source domain data and cross-source information. This contrastive loss is applied to an additional *Contrastive Head* in the model. To address term (3), we consider an adequately-large hypothesis space by using a neural network of sufficient size and incorporating an architecture previously shown to handle time-series data [2], [44].

4.2 Adaptation Components

CALDA's adaptation architecture is shown in Figure 3. The architecture includes a feature extractor, task classifier, domain classifier, and contrastive head. As illustrated in the figure, the feature extractor consists of a fully convolutional network, where the dense last layer acts as the task classifier. The domain classifier, consisting of a multi-layer perceptron, performs the adversary role during training. To handle variable-length time series data, the CNN includes a global

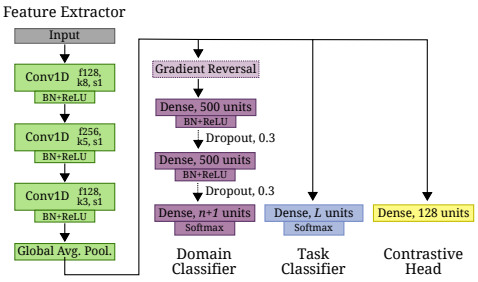

Fig. 3. The CALDA model architecture consists of a CNN task classifier, multi-layer perceptron domain classifier with global average pooling, and a contrastive head.

average pooling layer. Finally, CALDA includes an additional contrastive head to support the contrastive loss.

We describe source domain errors and feature-level domain invariance before moving onto our novel contrastive loss for multi-source domain adaptation, optionally with weak supervision, and the corresponding design choices.

4.2.1 Minimize Source Domain Errors

We minimize classification error on the source domains by feeding the outputs of feature extractor $F(\cdot;\theta_f)$ to a task classifier $C(\cdot;\theta_c)$ having a softmax output. Then, we update the parameters θ_f and θ_c to minimize a categorical cross-entropy loss $\mathcal{L}_y(y,p)$ using one-hot encoded true label y and softmax probabilities p. To handle multiple sources, we compute this loss over a mini-batch of examples drawn from each of the source domain distributions $\mathcal{D}_{S_i}$ for $i \in \{1,2,\dots n\}$:

$$\arg\min_{\theta_f,\theta_c} \sum_{i=1}^{n} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_{S_i}} [\mathcal{L}_y(y, C(F(x)))] \tag{6}$$

We employ the categorical cross-entropy loss, where y_i and p_i represent the ith components of y's one-hot encoding and the softmax probability output vector, respectively:

$$\mathcal{L}_y(y,p) = -\sum_{i=1}^{L} y_i \log p_i \tag{7}$$

4.2.2 Adversarial Learning

If we rely on only minimizing source domain error, we will obtain a classifier that likely does not transfer well to the target domain. One reason for this is that the extractor's selected feature representations may differ widely between source domains and the target domain. To remedy this problem, we invite a domain adversary to produce feature-level domain invariance. In other words, we align the feature extractor's outputs across domains. We achieve alignment by training a domain classifier (the "adversary") to correctly predict each example's domain labels (i.e., predict which domain each example originated from) while simultaneously training the feature extractor to make the domain classifier predict that the example is from a different domain.

We define a multi-class domain classifier with a softmax output as the adversary [2]. The domain classifier $D(\cdot;\theta_d)$ follows the feature extractor F in the network. However, we place a gradient reversal layer $\mathcal{R}(\cdot)$ between F and D, which multiplies the gradient during backpropagation by a negative constant $-\lambda_d$, yielding adversarial training [6]. Given domain labels $d \in \{0,1,\dots,n\}$, that map target examples to label $d_T = 0$ and source i examples to label $d_{S_i} = i$ for $i \in \{1,2,\dots n\}$, we update the model parameters θ_f and θ_d:

$$\begin{aligned}\arg\min_{\theta_f,\theta_d} \sum_{i=1}^{n} & \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}_{S_i}} [\mathcal{L}_d(d_{S_i}, D(\mathcal{R}(F(x))))] \\ & + \mathop{\mathbb{E}}_{x\sim\mathcal{D}_T^X} [\mathcal{L}_d(d_T, D(\mathcal{R}(F(x))))]\end{aligned} \tag{8}$$

This objective incorporates a categorical cross-entropy loss $\mathcal{L}_d$ similar to $\mathcal{L}_y$ that uses *domain labels* instead of class labels. Given the one-hot encoded representation of the true domain label d and the domain classifier's softmax probability output vector p, we compute the loss:

$$\mathcal{L}_d(d,p) = -\sum_{i=1}^{n} d_i \log p_i \tag{9}$$

4.2.3 Contrastive Learning

The above domain invariance adversarial loss does not leverage labeled data from the source domains. Prior work [2] only indirectly leverages labeled source domain data through jointly training the adversarial loss and task classifier loss on the labeled source domains. To better utilize source labels, we propose employing a supervised contrastive loss [9] to pull same-labeled examples together in the embedding space and push apart different-labeled examples, thereby making use of both positive and negative cross-source label information.

While the exact details vary based on the design decisions we will discuss in the next section, in general, contrastive learning has two roles: (1) pull same-label examples together and (2) push different-label examples apart. This process operates on pairs of representations z of examples (z_1, z_2). We call the first z_1 the "query" or "anchor", i.e., $z_1 = q$, where $q \in Q$ is drawn from the set of all example representations. To pull examples together, we create a pair (q,p), where the "positive" $p \in P$ is drawn from the set of all example representations having the *same label* as q. To push examples apart, we create another pair (q,n), where "negative" $n \in N$ is drawn from the set of example representations that have a *different label* than q. CALDA allows additional constraints to be placed on how positives and negatives are selected, such as selecting examples from the same domain, a different domain, or any domain. Figure 1 illustrates one query positive pair (q,p) and negative pair (q,n) for the cross-source and within-source cases. We may create additional positive and negative pairs for each query similarly.

We propose using a *supervised contrastive loss* based on a multiple-positive InfoNCE loss [9], [45]. Given the projected representation z of a query, the corresponding positive and negative sets P and N, a temperature hyperparameter τ, and cosine similarity $\text{sim}(z_1,z_2) = \frac{z_1^T z_2}{\|z_1\|\|z_2\|}$, we obtain:

$$\begin{aligned}&\mathcal{L}_c(z,P,N) = \\ &\frac{1}{|P|}\sum_{z_p\in P}\left[-\log\left(\frac{\exp\left(\frac{\text{sim}(z,z_p)}{\tau}\right)}{\sum_{z_k\in N\cup\{z_p\}}\exp\left(\frac{\text{sim}(z,z_k)}{\tau}\right)}\right)\right]\end{aligned} \tag{10}$$

Conceptually, for a given query, Equation 10 sums over each positive and normalizes by the number of positives. Inside of this sum, we compute what is mathematically equivalent to a log loss for a softmax-based classifier that classifies the query as the positive [8]. The denominator sums over both

the positive and also all the negatives corresponding to the query. Note, alternatively, the sum over the positive set could be moved inside the log, but keeping this sum outside has been found to perform better in prior works using InfoNCE [9].

Finally, we update weights by summing over the queries for each source domain and normalizing by the number of queries. In some framework instantiations, we similarly compute this loss over queries from the pseudo-labeled target domain data. Formally, given the source domain queries Q_{S_i}, the pseudo-labeled target domain queries Q_T, positive and negative sets $P_{d,y}$ and $N_{d,y}$ (construction depends on the instantiation of our method, discussed next), and the indicator function $\mathbf{1}$, we can update the model parameters θ_f and θ_z:

$$\begin{aligned}\arg\min_{\theta_f,\theta_z} \sum_{i=1}^{n} \left[\frac{1}{|Q_{S_i}|} \sum_{(z_q,y_q)\in Q_{S_i}} \mathcal{L}_c(z_q, P_{d_{S_i},y_q}, N_{d_{S_i},y_q}) \right] \\ +\mathbf{1}_{PL=True} \frac{1}{|Q_T|} \sum_{(z_q,\hat{y}_q)\in Q_T} \mathcal{L}_c(z_q, P_{d_T,\hat{y}_q}, N_{d_T,\hat{y}_q})\end{aligned} \tag{11}$$

4.2.4 Total Loss and Weak Supervision Regularizer

We jointly train each of these three adaptation components. Thus, the total loss that we minimize during training is a sum of each loss: source domain errors from Equation 6, adversarial learning from Equation 8, and contrastive learning from Equation 11. We further add a weighting parameter λ_c for the contrastive loss and note that the λ_d multiplier included in the gradient reversal layer $\mathcal{R}(\cdot)$ can be used as a weighting parameter for the adversarial loss.

Additionally, for the problem of MS-UDA with weak supervision, we include the weak supervision regularization term described in our prior work [2]. While individual labels for the unlabeled target domain data are unknown, this KL-divergence regularization term guides training toward learning model parameters that produce a class label distribution approximately matching the given label proportions on the unlabeled target data. This allows us to leverage target-domain label distribution information, if available.

$$\arg\min_{\theta_f,\theta_c} \left[D_{KL}\Big(Y_{true} \,\Big\|\, \mathbb{E}_{x\sim\mathcal{D}_T^X}\big[C(F(x))\big]\Big)\right] \tag{12}$$

4.3 Design Decisions for Contrastive Learning

The contrastive losses used in Equation 11 require positive and negative pairs for each query. CALDA supports multiple options for selecting these pairs. Here, we formalize the CALDA instantiations along the dimensions of 1) how to select example pairs across multiple domains, 2) whether to include a pseudo-labeled target domain in contrastive learning, and 3) whether to select examples randomly or based on difficulty.

4.3.1 Multiple Source Domains

When selecting pairs of examples for MS-UDA, we may choose to select two examples within a single domain, from two different domains, or a combination. We term these variations *Within-Source*, *Cross-Source*, and *Any-Source* Label-Contrastive learning. Note that similar terms apply if including the pseudo-labeled target domain. Recall that the motivation behind contrastive-learning MS-UDA is to leverage cross-source information. Because cross-source information is excluded in the Within-Source case, we hypothesize that Within-Source will perform poorly, whereas Any-Source and Cross-Source, which leverage the cross-source information, will yield improved results.

Formally, we define the sets of queries, positives, and negatives for each of these cases using set-builder notation. To simplify constructing these sets, we first create the auxiliary set K, which contains input x, class label y (or in the case of the target domain, the pseudo-label $\hat{y}$), and domain label d for all examples. Given a set of labeled examples $S_i \sim \mathcal{D}_{S_i}$ from each source domain $i \in \{1, 2, \ldots, n\}$, a set of unlabeled instances from the target domain $T_{train} \sim \mathcal{D}_T$, and a *projected representation* defined as the feature-level representation passed through an additional contrastive head $Z(\cdot;\theta_z)$ in the model $z = Z(F(x))$ (e.g., an additional fully-connected layer), we define a set K_S including both the (x, y) pair and the domain label $d = d_{S_i}$ (as defined in the previous section) of all source domains, a set K_T including both the $(x, \hat{y})$ pseudo-labeled pair and the domain label $d = d_T$ of the target domain, and set K, which is the union of K_S and K_T:

$$K_S = \{(x, y, d) \mid (x, y) \in S_i, i \in \{1, 2, \ldots, n\}, d = d_{S_i}\} \tag{13}$$

$$K_T = \{(x, \hat{y}, d) \mid x \in T_{train}, \hat{y} = \arg\max C(F(x)), d = d_T\} \tag{14}$$

$$K = K_S \cup K_T \tag{15}$$

Using K, we define the query set Q_{S_i} for each source domain and the query set Q_T for the target domain:

$$Q_{S_i} = \{(z, y) \mid (x, y, d) \in K, z = Z(F(x)), d = d_{S_i}\} \tag{16}$$

$$Q_T = \{(z, \hat{y}) \mid (x, \hat{y}, d) \in K, z = Z(F(x)), d = d_T\} \tag{17}$$

Next, we define the positive and negative sets for each framework instantiation.

(a) Within-Source Label-Contrastive learning: Positives for each query are selected from the *same domain* as the query with the *same label*. Negatives are selected that from the *same domain* as the query with a *different label*. Formally, we define the positive and negative sets P_{d_q,y_q} and N_{d_q,y_q} for each query of domain d_q and label y_q as follows:

$$P_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), d = d_q, y = y_q\} \tag{18}$$

$$N_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), d = d_q, y \neq y_q\} \tag{19}$$

(b) Any-Source Label-Contrastive learning: Positives for each query are selected having the *same label* and coming from *any domain*. Negatives are selected with a *different label* and from *any domain*:

$$P_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), y = y_q\} \tag{20}$$

$$N_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), y \neq y_q\} \tag{21}$$

(c) Cross-Source Label-Contrastive learning: Positives for each query are selected from a *different domain* with the *same label*. Negatives are selected from a *different domain* with a *different label*:

$$P_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), d \neq d_q, y = y_q\} \tag{22}$$

$$N_{d_q,y_q} = \{z \mid (x, y, d) \in K, z = Z(F(x)), d \neq d_q, y \neq y_q\} \tag{23}$$

Note that these cases are distinguished based on whether $d = d_q$ (Within-Source), $d \neq d_q$ (Cross-Source), or there is no constraint (Any-Source).

4.3.2 Pseudo-Labeled Target Domain

Prior contrastive learning work for single-source domain adaptation utilizes a supervised contrastive loss on the combined single source domain and the target domain. However, since this loss depends on labels, such methods require pseudo-labeling the target domain data. The methods rely on the classifier producing correct class labels, which can then be used in the supervised contrastive loss. Unfortunately, pseudo-labeling the target domain is a challenging problem, and classification errors are likely [12]. If the pseudo-labels are incorrect, then this may hurt contrastive learning performance. Because US-MDA utilizes multiple domains, instead of performing contrastive learning between the source and target domains, we may perform contrastive learning among the source domains, which we may improve performance. We include whether to utilize pseudo-labeled target domain data during training as an additional CALDA dimension. If pseudo-labeled target domain data is included during contrastive learning, $PL = True$ in the contrastive learning objective (Equation 11), otherwise $PL = False$.

4.3.3 Pair Selection by Difficulty

Outside of domain adaptation, selecting *hard* examples has been found beneficial for contrastive learning [13], [14]. However, recent theoretical work suggests that hard examples implicitly contribute more to the loss, thus mitigating the need for explicitly selecting hard examples in contrastive learning [9]. To determine if explicitly selecting hard examples is beneficial in multi-source domain adaptation, we propose a method for hard sampling in CALDA and compare it with random sampling – the final dimension of our CALDA framework. For brevity, we give the equations for Cross-Source Label-Contrastive learning, but the other variations can be constructed by changing the domain constraint of each set.

For *hard sampling*, we select a subset of hard positive and negative examples. This necessitates that we define "hard examples." In each case, the domain constraint is the same for both positives and negatives, thus the key difference is whether they have the same label as the query or not. The examples that would most help the model learn this decision boundary are those that are predicted to be on the wrong side. Thus, we select *hard* examples as examples that are currently predicted to be on the wrong side of this decision boundary. We define *hard positives* as examples with the query's label but with a different predicted label (with respect to the current model predictions) and *hard negatives* as examples of a class other than the query's label but that are predicted to have the query's label. Both are from a different domain than the query (in the Cross-Source case). Focusing on the Cross-Source Label-Contrastive case, for a query with domain d_q and true label y_q and the current model prediction $\hat{y} = \arg\max C(F(x))$, we define hard positive and negative sets $\bar{P}$ and $\bar{N}$:

$$\bar{P}_{d_q,y_q} = \{z \,|(x,y,d) \in K, z = Z(F(x)), \\ d \neq d_q, y = y_q, \hat{y} \neq y_q\} \tag{24}$$

$$\bar{N}_{d_q,y_q} = \{z \,|(x,y,d) \in K, z = Z(F(x)), \\ d \neq d_q, y \neq y_q, \hat{y} = y_q\} \tag{25}$$

However, there is no guarantee there will always be positives and negatives that are misclassified. For example, the task classifier likely makes accurate predictions later on during training. Instead, we propose using a relaxed version of hardness: take the top-k_1 hardest positives and top-k_2 hardest negatives in terms of a softmax cross-entropy loss. To obtain a relaxation of $\hat{y} \neq y_q$ for the positives, we can find positive examples with a *high* task classification loss for that example via $\mathcal{L}_y$. This is because a positive is defined as having the same label as the query, so having a high task loss for the positive corresponds to being on the wrong side of the positive-negative decision boundary. To obtain a relaxation of $\hat{y} = y_q$ for the negatives, we can find negative examples with a *low* task classification loss where we replace the true class label y with the query's class label y_q. This finds the negatives that are most easily misclassified as having the query's class label, i.e., those on the wrong side of the decision boundary.

Thus, we define the relaxed hard positive and negative sets $\tilde{P}$ and $\tilde{N}$ in terms of the softmax-based cross-entropy loss $\mathcal{L}_y$, with loss thresholds h_p and h_n chosen such that we have k_1 positives and k_2 negatives (i.e, $|P| = k_1$ and $|N| = k_2$):

$$\tilde{P}_{d_q,y_q} = \{z \,|(x,y,d) \in K, z = Z(F(x)), d \neq d_q, y = y_q, \\ \mathcal{L}_y(y, C(F(x))) > h_p\} \tag{26}$$

$$\tilde{N}_{d_q,y_q} = \{z \,|(x,y,d) \in K, z = Z(F(x)), d \neq d_q, y \neq y_q, \\ \mathcal{L}_y(y_q, C(F(x))) < h_n\} \tag{27}$$

The contrastive weight update in Equation 11 can now be adjusted to use the relaxed hard positive and negative sets $\tilde{P}$ and $\tilde{N}$. As an alternative to hard sampling, we may instead use *random sampling*, to select a random subset of positives and negatives that pair with each query.

5 Experimental Validation

We validate our hypothesis that CALDA will improve time-series MS-UDA through contrastive learning based on experimental analysis. We also apply CALDA to synthetic and real-world datasets to address the three design decisions, with and without weak supervision. Finally, we validate the impact of the adversary and unlabeled target domain data.

5.1 Datasets

We construct several synthetic time series that vary in complexity to aid in comparing alternative adaptation frameworks. He et al. [46] observe that adapting models to new domains becomes more challenging as the data become more complex. We therefore generate synthetic data of varying complexity to examine corresponding differences in adaptation performance. In the first scenario (SW), we generate a 2D normal distribution for each domain that represents a sum of two sine waves with frequencies $f_{i,1}$ and $f_{i,2}$ (Hz), obtaining a time series example (x_i, y_i), where x_i is a vector. In this sine wave scenario, inter-domain or intra-domain separation is created through translating or rotating the class distributions by a fixed amount (examples are illustrated in the Supplementary Material). In the second scenario, the synthetic data more closely resemble the complexities found in the real-world datasets. We generate multivariate (9 dimensions) data created from a mixture of Gaussians. Separation between domains is created by varying the Gaussian parameters for each dimension. We repeat this scenario for 1, 2, and 3

Gaussians (scenarios 1GMM, 2GMM, and 3GMM) to vary the corresponding complexity of the data distributions and domain-invariant features that must be learned. Note that for both scenarios, as the number of source domains n increases, the source domains cover a larger region of the space compared to the target domain.

We also evaluate the real-world efficacy of the CALDA framework using six real-world multi-variate time series datasets. These include four human activity recognition datasets (UCI HAR [4], UCI HHAR [47], WISDM AR [48], and WISDM AT [49]) and two multivariate EMG datasets collected using a Myo armband (Myo EMG [26] and NinaPro Myo [50]). The Supplemental Material contain details about dataset pre-processing and hyperparameter tuning.

5.2 Ablation Studies

Using a set of ablation studies, we identify the appropriate instantiations of our CALDA framework to compare with the baseline methods. We compare instantiations across each component of our framework: (1) including the pseudo-labeled target domain (P), (2) using within-source (CALDA-In), any-source (CALDA-Any), or cross-source (CALDA-XS) examples for each query, and (3) random sampling (R) versus hard sampling (H). A one-sided paired student's t-test indicates whether accuracy improvements are statistically significant.

5.2.1 Pseudo-Labeled Target Domain

To determine whether to include pseudo-labeled target domain data or not, we compare experimental results with and without pseudo labeling, respectively. Comparing Tables 1 and 2, we find that regardless of the choices for the other components in our framework, pseudo labeling generally performs worse than without pseudo-labeling on the real-world datasets, i.e., the corresponding values in Table 1 are significantly lower than those in Table 2 ($p < 0.01$). Interestingly, random sampling typically performs better than hard sampling when using pseudo labels. This is likely because incorrectly pseudo-labeled target data may often be selected during hard sampling and thereby degrade contrastive learning performance. Random sampling helps to partially reduce this performance degradation by reducing the likelihood that the pseudo-labeled target domain is used in contrastive learning. However, we obtain even better performance by explicitly excluding the target domain in contrastive learning, as shown in Table 2. Thus, in subsequent experiments, we exclude pseudo-labeling. Future work is required to determine which pseudo labeling techniques perform well in a time series context. For the synthetic datasets, we do not observe significant differences between pseudo labeling and not pseudo labeling (see Supplemental Material).

5.2.2 Multiple Source Domains

We next analyze the results in Table 2 to determine whether to use within-source, any-source, or cross-source examples. Starting with the results on real-world datasets, the best method is consistently among the CALDA-Any and CALDA-XS variants. In particular, CALDA-Any,R is the best-performing instantiation on average with the two CALDA-XS variants ranking second, though no variation offers statistically significant improvement. Thus, we construct and include results on the additional synthetic datasets. As expected, all methods perform almost identically for no domain shift. However, when we include various types of synthetic domain shifts, we see significant differences. CALDA-Any,H and CALDA-XS,H are the best methods on average and are both significantly better than CALDA-In,R and CALDA-In,H ($p < 0.01$). Thus, a similar trend emerges from both the real-world data and synthetic data: CALDA-Any and CALDA-XS instantiations outperform CALDA-In. Since CALDA-In ignores cross-source information by only utilizing within-source examples, the other instantiations performing better validates our hypothesis that leveraging cross-source information can yield improved transfer. Thus, we conclude that CALDA-Any and CALDA-XS, which both leverage cross-source information, are the two most promising methods of selecting source domain examples for our framework.

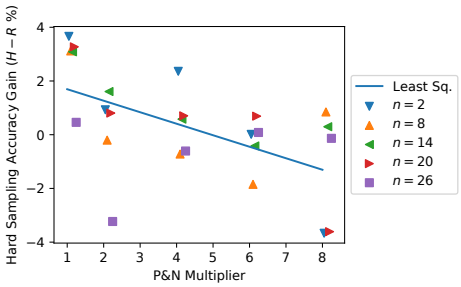

Fig. 4. Comparing hard vs. random sampling as $|P|$ and $|N|$ increase.

5.2.3 Pair Selection by Difficulty

The choice between selecting examples by hard or random sampling is more variable than the above design decisions. Comparing hard sampling with random sampling on the real datasets in Table 2, we observe random sampling for CALDA-Any to perform significantly better than hard sampling ($p < 0.05$). However, random vs. hard sampling differences for the other methods are not statistically significant. On the synthetic datasets with domain shifts, we observe the opposite: hard sampling versions of CALDA-Any and CALDA-XS are significantly better than random sampling ($p < 0.01$). This may indicate this design choice depends on the type of data and domain shift.

We further investigate hard vs. random sampling by running an additional set of experiments for CALDA-XS on WISDM AR, the dataset where CALDA-XS,H performed better than all other instantiations, and specifically, where it outperformed CALDA-XS,R. The results as we increase the number of positives and negatives (two of the hyperparameters) via a positive/negative multiplier are shown in Figure 4. On the left, we observe CALDA-XS,H outperforms CALDA-XS,R. Moving to the right, the performance gain from hard sampling reduces, which is expected since hard and random sampling are no different in the limit of using all positive and negative examples in each mini-batch. We conclude that hard sampling may yield an improvement over random sampling in some situations, particularly those for which the optimal hyperparameters for a dataset include relatively few positives and negatives (such as is the case on the WISDM AR dataset).

Finally, because CALDA-Any,R performed best on the real-world data and CALDA-XS,H was tied for second best

TABLE 1
Ablation study of CALDA instantiations that include the target-domain data via pseudo labeling.

Dataset	*CALDA-In,R,P*	*CALDA-In,H,P*	*CALDA-Any,R,P*	*CALDA-Any,H,P*	*CALDA-XS,R,P*	*CALDA-XS,H,P*
UCI HAR	92.3 ± 2.5	92.0 ± 3.0	92.6 ± 2.6	91.9 ± 3.3	92.1 ± 3.0	92.4 ± 3.3
UCI HHAR	89.2 ± 4.2	89.0 ± 4.5	88.8 ± 4.7	86.2 ± 5.8	89.4 ± 4.3	87.4 ± 5.6
WISDM AR	72.5 ± 9.1	71.1 ± 8.0	74.7 ± 8.3	71.3 ± 9.1	74.5 ± 8.4	73.5 ± 8.3
WISDM AT	68.6 ± 8.7	68.5 ± 7.2	61.6 ± 12.9	63.3 ± 9.8	62.0 ± 10.9	60.2 ± 12.3
Myo EMG	83.4 ± 5.5	82.4 ± 5.4	83.7 ± 5.5	82.9 ± 6.2	84.2 ± 5.3	82.0 ± 6.4
NinaPro Myo	52.0 ± 5.3	52.6 ± 4.5	51.0 ± 4.6	52.4 ± 4.1	51.5 ± 5.3	52.3 ± 4.6
Average	77.2 ± 5.9	76.7 ± 5.5	76.2 ± 6.5	75.4 ± 6.5	76.5 ± 6.2	75.4 ± 6.8

TABLE 2
Ablation study comparing hard and random sampling for each CALDA instantiation. Bold denotes highest accuracy in each row.

Dataset	*CALDA-In,R*	*CALDA-In,H*	*CALDA-Any,R*	*CALDA-Any,H*	*CALDA-XS,R*	*CALDA-XS,H*
UCI HAR	93.5 ± 2.0	93.6 ± 2.2	93.1 ± 2.3	**93.7 ± 2.1**	93.4 ± 2.1	93.4 ± 2.5
UCI HHAR	89.3 ± 3.9	89.4 ± 3.9	**89.8 ± 3.7**	88.7 ± 4.6	**89.8 ± 3.9**	89.4 ± 4.0
WISDM AR	79.0 ± 6.9	79.4 ± 7.7	80.2 ± 7.1	80.3 ± 6.9	80.0 ± 6.3	**81.4 ± 7.9**
WISDM AT	71.4 ± 8.2	71.0 ± 8.3	**72.1 ± 7.5**	71.2 ± 8.4	70.7 ± 6.5	71.0 ± 8.6
Myo EMG	83.0 ± 5.3	82.7 ± 6.1	**83.8 ± 5.6**	83.6 ± 5.4	83.4 ± 5.6	83.3 ± 5.3
NinaPro Myo	57.3 ± 3.6	56.9 ± 3.6	57.7 ± 3.8	55.9 ± 4.2	**58.0 ± 3.8**	56.0 ± 3.9
Synth InterT 0	93.7 ± 0.2	93.7 ± 0.1	93.8 ± 0.2	93.7 ± 0.2	93.7 ± 0.3	**93.9 ± 0.1**
Synth InterR 0	**94.2 ± 0.2**	**94.2 ± 0.1**	94.1 ± 0.1	94.1 ± 0.2	94.1 ± 0.1	94.0 ± 0.2
Synth IntraT 0	93.7 ± 0.2	93.8 ± 0.2	93.8 ± 0.3	**93.9 ± 0.2**	93.7 ± 0.3	93.6 ± 0.3
Synth IntraR 0	**94.1 ± 0.2**	**94.1 ± 0.2**	**94.1 ± 0.2**	94.0 ± 0.1	**94.1 ± 0.2**	93.9 ± 0.2
Synth InterT 10	69.8 ± 17.0	**70.8 ± 15.8**	69.4 ± 17.0	70.7 ± 15.0	68.4 ± 14.2	70.5 ± 14.5
Synth InterR 1.0	67.5 ± 12.5	69.9 ± 12.6	63.4 ± 10.2	**77.4 ± 8.1**	63.8 ± 10.3	76.3 ± 8.9
Synth IntraT 10	74.9 ± 6.4	73.6 ± 6.6	75.6 ± 8.4	75.9 ± 8.9	**77.9 ± 8.9**	76.2 ± 8.3
Synth IntraR 1.0	74.9 ± 8.3	74.3 ± 7.6	**77.5 ± 7.0**	76.0 ± 6.7	75.0 ± 7.9	76.9 ± 6.1
Real-World Avg.	79.7 ± 5.0	79.6 ± 5.4	**80.2 ± 5.0**	79.7 ± 5.3	79.9 ± 4.7	79.9 ± 5.4
Synth (No Shift) Avg.	93.9 ± 0.2	93.9 ± 0.1	**94.0 ± 0.2**	93.9 ± 0.2	93.9 ± 0.2	93.9 ± 0.2
Synth Avg.	71.8 ± 11.0	72.2 ± 10.7	71.5 ± 10.7	**75.0 ± 9.7**	71.3 ± 10.3	**75.0 ± 9.4**

on the real-world data and tied for best on the synthetic datasets, we select these two methods as the most-promising instantiations of our framework for subsequent experiments.

5.3 MS-UDA: CALDA vs. Baselines

To measure the CALDA's performance improvement, we compare the two most promising instantiations, CALDA-Any,R and CALDA-XS,H, with baselines and prior work. We include an approximate domain adaptation lower bound that performs no adaptation during training (*No Adaptation*). This allows us to see how much improvement results from utilizing domain adaptation. We include an approximate domain adaptation upper bound showing the performance achievable if we did have labeled target domain data available (*Train on Target*). For a contrastive domain adaptation baseline, we include the Contrastive Adaptation Network (*CAN*) [11] modified to employ a time-series compatible feature extractor. Finally, we include *CoDATS* [2] to see if our CALDA framework improves over prior multi-source time series domain adaptation work. The results are presented in Table 3.

We first examine the No Adaptation and Train on Target baselines, which train only on the source domain data or train directly on the target domain data respectively. The performance of the No Adaptation baseline can be viewed as an approximate measure of domain adaptation difficulty, where lower No Adaptation performance indicates a more challenging problem, i.e., with a larger domain gap between the source domains and the target domain. Accordingly, we can identify WISDM AR and WISDM AT as the most challenging activity recognition datasets and NinaPro Myo as the most challenging EMG dataset. In contrast, Train on Target performs well on all but one dataset. It does this well by "cheating", i.e., looking at the target domain labels and thereby eliminating the domain gap. However, the NinaPro Myo dataset is challenging enough that even with no domain gap, we cannot obtain near-perfect accuracy.

Next, we compare CALDA-Any,R and CALDA-XS,H with the No Adaptation and CoDATS baselines. One of the two CALDA instantiations always performs best. Similarly, CALDA-Any,R and CALDA-XS,H significantly outperform both No Adaptation and CoDATS across all datasets ($p < 0.01$). On the real-world datasets, the largest improvement over CoDATS is 2.4% on WISDM AT for CALDA-Any,R and 3.4% on WISDM AR for CALDA-XS,H. The largest improvement over No Adaptation is 12.9% and 12.5%, respectively, on UCI HHAR. On average, we observe a 1.6% and 1.3% improvement of these two CALDA instantiations over CoDATS and a 6.3% and 6.0% improvement over No Adaptation, respectively. On the synthetic datasets, these

TABLE 3
Comparing target domain accuracy of the most-promising CALDA instantiations with baselines. Bold denotes CALDA outperforming baselines. Underline denotes highest accuracy in each row.

Dataset	No Adaptation	CAN	CoDATS	*CALDA-Any,R*	*CALDA-XS,H*	Train on Target
UCI HAR	88.8 ± 4.2	89.0 ± 3.7	92.8 ± 3.3	**93.1 ± 2.3**	**93.4 ± 2.5**	99.6 ± 0.1
UCI HHAR	76.9 ± 6.3	77.5 ± 5.3	88.6 ± 3.9	**89.8 ± 3.7**	**89.4 ± 4.0**	98.9 ± 0.2
WISDM AR	72.1 ± 8.0	61.3 ± 7.5	78.0 ± 8.4	**80.2 ± 7.1**	**81.4 ± 7.9**	96.5 ± 0.1
WISDM AT	69.9 ± 7.1	66.2 ± 9.6	69.7 ± 6.6	**72.1 ± 7.5**	**71.0 ± 8.6**	98.8 ± 0.1
Myo EMG	77.4 ± 5.2	74.1 ± 6.3	82.2 ± 5.4	**83.8 ± 5.6**	**83.3 ± 5.3**	97.7 ± 0.1
NinaPro Myo	54.8 ± 3.6	56.8 ± 3.2	55.9 ± 5.0	**57.7 ± 3.8**	56.0 ± 3.9	77.8 ± 1.3
Synth SW	62.2 ± 11.0	78.2 ± 7.7	63.8 ± 10.6	71.5 ± 10.7	75.0 ± 9.5	93.7 ± 0.2
Synth 1GMM	88.5 ± 3.4	90.1 ± 4.2	91.4 ± 2.8	**92.3 ± 3.7**	91.4 ± 3.4	98.4 ± 0.5
Synth 2GMM	86.3 ± 6.5	89.4 ± 5.7	91.8 ± 7.8	**93.6 ± 5.1**	**94.0 ± 4.7**	100.0 ± 0.0
Synth 3GMM	80.1 ± 9.0	83.8 ± 9.1	91.3 ± 8.1	**91.4 ± 6.9**	**92.2 ± 7.2**	100.0 ± 0.0
Real-World Avg.	73.9 ± 5.8	71.3 ± 6.0	78.6 ± 5.4	**80.2 ± 5.0**	**79.9 ± 5.4**	94.9 ± 0.3
Synth Avg.	79.3 ± 7.5	85.4 ± 6.7	84.6 ± 7.3	**87.2 ± 6.6**	**88.2 ± 6.2**	98.0 ± 0.2

improvements are even larger: 7.7% and 11.2% improvement over CoDATS and 9.3% and 12.8% improvement over no Adaptation. These experimental results across a variety of real world and synthetic time-series datasets confirm the benefit of utilizing cross-source information through the CALDA framework for time-series multi-source domain adaptation.

Finally, we compare CALDA-Any,R and CALDA-XS,H with the contrastive domain adaptation baseline CAN. Both CALDA instantiations significantly outperform CAN on all of the real-world time series datasets ($p < 0.01$). The largest improvements over CAN on the real-world datasets are 18.9% for CALDA-Any,R and 20.1% for CALDA-XS,H on WISDM AR. On average, we observe an 8.9% and 8.6% improvement in the two CALDA instantiations over CAN. On the synthetic datasets, CAN yields the strongest results on the simple synthetic data (SW). CAN relies on clustering for pseudo-labeling target domain data, which works well for the simple, synthetically-generated Gaussian domain shifts. CALDA outperforms the other methods for the more complex GMM scenarios ($p < 0.05$), and the amount of improvement grows with the data complexity. These results indicate that while CAN may be successful with some types of domain shifts such as those found in image datasets or clustered synthetic time series, we find that CALDA better handles the domain shifts found in complex real-world time series datasets.

5.4 MS-UDA with Weak Supervision

We additionally study whether our framework yields improved results for domain adaptation with weak supervision. First, we simulate obtaining target domain label proportions by estimating these proportions on the target domain training set and incorporate our weak supervision regularizer into each method to leverage this additional information. Following this, we determine the sensitivity of each method to noise in the estimated label proportions since, for example, if these label proportions are acquired from participants' self-reports, there will be some error in the reported proportions.

5.4.1 CALDA with Weak Supervision

We compare the two most promising instantiations of CALDA-WS (*CALDA-XS,H,WS* and *CALDA-Any,R,WS*) with *CoDATS-WS*. The results are shown in Table 4. Similar to no weak supervision, CALDA-Any,R,WS improves over both No Adaptation and CoDATS-WS across all datasets and CALDA-XS,H,WS in all except one case ($p < 0.01$). On average we observe a 4.1% and 3.6% improvement of the two CALDA instantiations over CoDATS-WS and a 9.8% and 9.3% improvement over No Adaptation on the real-world datasets. On the synthetic datasets, we observe a 12.3% and 16.5% improvement of CALDA over both CoDATS-WS and No Adaptation. These results demonstrate the efficacy of CALDA for the domain adaptation when incorporating weak supervision.

By comparing Table 3 with 4, we can measure the benefit of weak supervision. In all cases, the CALDA instantiations with weak supervision significantly improve over CALDA without weak supervision ($p < 0.01$). On the real-world datasets, this is also the case with CoDATS: CoDATS-WS improves over CoDATS ($p < 0.05$). On the real-world datasets, we observe a 3.5% and 3.3% improvement for the CALDA instantiations by including weak supervision. On the synthetic datasets, these differences are 3.0% and 3.7%. We observe the largest performance gains from utilizing weak supervision on the two unbalanced datasets: 5.0% and 3.3% improvements of the two instantiations on WISDM AR and 10.8% and 12.3% improvements on WISDM AT. Because these datasets are unbalanced, larger differences on these datasets are expected since our weak supervision regularization term capitalizes on label distribution differences among the domains. These gains demonstrate: (1) the benefit of leveraging weak supervision for domain adaptation when available, and (2) the observation that CALDA yields improvements over prior work, even for this related problem setting.

5.4.2 Sensitivity of Weak Supervision to Noise

By leveraging weak supervision, we were able to improve performance. However, in the above experiments, we simulated obtaining target domain label proportions by estimating those proportions exactly on the target domain training dataset. Now we perform additional experiments to determine how robust these methods are to noise in the estimated label proportions. Since weak supervision has the greatest effect when label distributions differ among domains, we compare

TABLE 4
Comparing target domain accuracy for domain adaptation methods utilizing weak supervision. Bold denotes CALDA outperforming baselines. Underline denotes highest accuracy in each row.

Dataset	No Adaptation	CoDATS-WS	*CALDA-Any,R,WS*	*CALDA-XS,H,WS*	Train on Target
UCI HAR	88.8 ± 4.2	92.9 ± 3.2	**94.8 ± 1.8**	**95.5 ± 2.1**	99.6 ± 0.1
UCI HHAR	76.9 ± 6.3	88.2 ± 4.6	**90.2 ± 3.8**	**89.8 ± 4.1**	98.9 ± 0.2
WISDM AR	72.1 ± 8.0	84.9 ± 7.2	**85.2 ± 6.9**	84.7 ± 7.0	96.5 ± 0.1
WISDM AT	69.9 ± 7.1	72.1 ± 10.3	**82.9 ± 7.3**	**83.3 ± 7.1**	98.8 ± 0.1
Myo EMG	77.4 ± 5.2	79.5 ± 5.7	**85.1 ± 4.6**	**84.7 ± 4.6**	97.7 ± 0.1
NinaPro Myo	54.8 ± 3.6	54.9 ± 4.2	**58.8 ± 3.8**	**56.0 ± 4.5**	77.8 ± 1.3
Synth SW	62.2 ± 11.0	62.2 ± 11.6	**74.5 ± 11.3**	**78.8 ± 9.2**	93.7 ± 0.1
Synth 1GMM	88.5 ± 3.4	90.4 ± 3.0	**91.7 ± 2.5**	**90.8 ± 3.7**	98.4 ± 0.5
Synth 2GMM	86.3 ± 6.5	86.4 ± 12.9	**92.3 ± 7.3**	**93.7 ± 7.2**	100.0 ± 0.0
Synth 3GMM	80.1 ± 9.0	87.7 ± 9.2	84.1 ± 13.5	**89.1 ± 10.9**	100.0 ± 0.0
Real-World Avg.	73.9 ± 5.8	79.6 ± 5.9	**83.7 ± 4.7**	**83.2 ± 4.9**	94.9 ± 0.3
Synth Avg.	62.2 ± 11.0	62.2 ± 11.6	**85.7 ± 8.7**	**88.1 ± 7.8**	93.7 ± 0.2

these methods with various noise budgets on the unbalanced WISDM datasets. A noise budget of 0.1 indicates that approximately 10% of the class labels can be redistributed. In the case of human activity recognition, if all hours of the day correspond with an activity, then this represents 10% of the day being attributed to an incorrect activity when self-reporting label proportions for weak supervision. The results are shown in Table 5. Note that label proportions are redistributed according to the noise budget and then renormalized so the proportions remain a valid distribution. In the table we provide the *True Post-Norm. Noise* column to validate that the true post-normalized noise on average is close to the desired noise budget.

CALDA typically outperforms CoDATS both with and without weak supervision on the WISDM AR dataset (the final row corresponds to methods without weak supervision). Similarly, the best method in each row is always one of the two CALDA instantiations. We observe that even with a noise budget of 0.1, CALDA-WS and CoDATS-WS perform better than CALDA and CoDATS without weak supervision. However, beyond this threshold, we find additional noise degrades performance on WISDM AR. From these results, we conclude the maximum acceptable noise level for weak supervision on WISDM AR is between 0.1 and 0.2. In the case of WISDM AT, the two CALDA instantiations outperform CoDATS both without weak supervision and with weak supervision regardless of the amount of noise. We find that it takes a noise budget of 0.1 before CoDATS-WS degrades to the performance of CoDATS without weak supervision. However, for CALDA-WS, the noise budget can be as large as 0.4 before it degrades to the performance of CALDA without weak supervision.

Overall, on these two datasets, we find that leveraging both weak supervision and cross-source label information can yield improved domain adaptation performance, even with some noise in the weak supervision label information. Though, the acceptable amount of noise depends on the dataset. On both datasets, CoDATS-WS requires a noise level of no more than approximately 0.1, and both CALDA-WS instantiations have similar limits on WISDM AR. However, on WISDM AT, the noise budget for either CALDA-WS instantiation can be as high as 0.4 – four times that of CoDATS-WS. Thus, we conclude that our CALDA framework improves over CoDATS with and without weak supervision and also that our CALDA framework can yield higher robustness to noise in the weak supervision label information on some datasets.

5.5 Validating Assumptions

Now we examine the validity of two key assumptions.

5.5.1 Importance of the Adversary

Using the CALDA framework, we investigated various design choices for how to use contrastive learning for domain adaptation. However, we made the assumption that adversarial learning is an important component for each of these instantiations. Here we illustrate why. For the two most-promising instantiations, we run experiments when excluding the adversary. The results on the real-world datasets are shown in Table 6. The methods with an adversary are far superior to those when we exclude the adversary ($p < 0.01$). This justifies our inclusion of the adversary.

5.5.2 Importance of Unlabeled Target Domain Data

Finally, in the problem of unsupervised domain adaptation, we have unlabeled target domain data available for use during training. Unsupervised domain adaptation methods typically make the assumption that these data are useful for improving target-domain performance. Here, both CoDATS and CALDA leverage this data via adversarial learning, which as observed above is vital to domain adaptation performance. However, another alternative is to only perform adversarial learning among the multiple source domains and exclude the target-domain unlabeled data, i.e., promote domain-invariant features among only the multiple source domains through the adversarial loss. This is related to the problem of domain generalization [51]. The results for the corresponding *CoDATS-DG*, *CALDG-XS,H*, and *CALDG-Any,R* methods are shown in Table 7. For comparison, we also include two domain generalization methods *Sleep-DG* [5] and *AFLAC-DG* [52]. Comparing Tables 3 and 7, on the real-world datasets we observe including the unlabeled data yields significantly higher accuracy of CoDATS, CALDA-Any,R, and CALDA-XS,H ($p < 0.01$). This is similarly true on the synthetic data,

TABLE 5
Weak supervision sensitivity to noise. Bold denotes higher accuracy than CoDATS. Underlining denotes best method in each row.

Dataset	Weak Supervision	Noise Budget	True Post-Norm. Noise	CoDATS	*CALDA-XS,H*	*CALDA-Any,R*
WISDM AR	Yes	0.0	0.0	84.9 ± 7.2	84.7 ± 7.0	**85.2 ± 6.9**
		0.05	0.06	82.9 ± 7.4	**83.4 ± 6.2**	**84.0 ± 6.0**
		0.1	0.12	79.1 ± 8.4	**83.1 ± 7.7**	**81.8 ± 6.8**
		0.2	0.22	74.6 ± 8.8	**78.9 ± 6.7**	**78.4 ± 8.1**
		0.4	0.38	64.9 ± 9.7	**68.9 ± 8.8**	**69.9 ± 9.2**
WISDM AR	No	N/A	N/A	78.0 ± 8.4	**81.4 ± 7.9**	**80.2 ± 7.1**
WISDM AT	Yes	0.0	0.0	72.1 ± 10.3	**83.3 ± 7.1**	**82.9 ± 7.3**
		0.05	0.07	71.0 ± 11.9	**81.8 ± 6.6**	**82.4 ± 7.3**
		0.1	0.13	69.9 ± 14.3	**81.1 ± 7.9**	**82.7 ± 7.4**
		0.2	0.23	65.6 ± 11.4	**78.3 ± 8.3**	**79.2 ± 7.8**
		0.4	0.40	56.3 ± 12.8	**72.0 ± 8.6**	**71.3 ± 8.3**
WISDM AT	No	N/A	N/A	69.7 ± 6.6	**71.0 ± 8.6**	**72.1 ± 7.5**

TABLE 6
Ablation study comparing CALDA with or without an adversary. Bold denotes highest accuracy in each row.

Dataset	*CALDA-Any,R,NoAdv*	*CALDA-XS,H,NoAdv*	*CALDA-Any,R*	*CALDA-XS,H*
UCI HAR	89.9 ± 3.4	89.8 ± 3.5	93.1 ± 2.3	**93.4 ± 2.5**
UCI HHAR	74.0 ± 7.2	74.7 ± 6.8	**89.8 ± 3.7**	89.4 ± 4.0
WISDM AR	70.9 ± 6.8	72.1 ± 6.2	80.2 ± 7.1	**81.4 ± 7.9**
WISDM AT	70.9 ± 7.7	70.2 ± 7.9	**72.1 ± 7.5**	71.0 ± 8.6
Myo EMG	76.2 ± 5.2	76.2 ± 5.0	**83.8 ± 5.6**	83.3 ± 5.3
NinaPro Myo	53.4 ± 4.0	51.4 ± 4.4	**57.7 ± 3.8**	56.0 ± 3.9
Average	73.2 ± 5.8	73.1 ± 5.7	**80.2 ± 5.0**	79.9 ± 5.4

TABLE 7
Comparing domain adaptation performance excluding unlabeled target domain data during training. Bold denotes methods outperforming CoDATS-DG and No Adaptation baselines. Underline denotes highest accuracy.

Dataset	No Adaptation	CoDATS-DG	Sleep-DG	AFLAC-DG	CALDG-Any,R	CALDG-XS,H	Train on Target
UCI HAR	88.8 ± 4.2	88.4 ± 3.7	87.0 ± 4.8	**89.3 ± 4.6**	**89.5 ± 3.8**	**90.0 ± 3.7**	99.6 ± 0.1
UCI HHAR	76.9 ± 6.3	76.0 ± 6.1	75.4 ± 6.7	76.6 ± 6.4	76.6 ± 6.4	76.2 ± 6.9	98.9 ± 0.2
WISDM AR	72.1 ± 8.0	66.9 ± 8.7	66.9 ± 8.9	70.9 ± 7.8	68.9 ± 8.8	70.2 ± 7.8	96.5 ± 0.1
WISDM AT	69.9 ± 7.1	69.7 ± 7.8	68.3 ± 8.9	69.7 ± 6.6	**70.7 ± 7.4**	**70.7 ± 7.7**	98.8 ± 0.1
Myo EMG	77.4 ± 5.2	73.0 ± 5.3	73.9 ± 5.9	74.3 ± 5.5	**78.4 ± 4.8**	76.8 ± 5.8	97.7 ± 0.1
NinaPro Myo	54.8 ± 3.6	50.6 ± 4.3	50.4 ± 4.7	51.1 ± 3.5	**55.1 ± 3.7**	49.8 ± 4.6	77.8 ± 1.3
Synth InterT 10	62.6 ± 18.8	68.4 ± 13.9	**68.7 ± 13.9**	67.6 ± 13.4	**69.8 ± 16.6**	**71.2 ± 15.9**	93.4 ± 0.2
Synth InterR 1.0	52.4 ± 7.8	53.3 ± 8.0	**53.7 ± 7.0**	**66.6 ± 7.7**	**65.4 ± 10.5**	**75.5 ± 8.1**	94.0 ± 0.0
Synth IntraT 10	70.6 ± 8.5	66.5 ± 10.8	68.2 ± 10.2	68.5 ± 7.5	**73.2 ± 8.7**	**74.5 ± 8.7**	93.7 ± 0.2
Synth IntraR 1.0	63.3 ± 9.0	59.9 ± 7.1	62.4 ± 6.3	**64.0 ± 7.8**	**77.3 ± 8.0**	**76.6 ± 6.1**	93.6 ± 0.2
Real-World Avg.	73.9 ± 5.8	71.4 ± 6.1	71.0 ± 6.7	72.7 ± 5.8	73.8 ± 5.9	73.1 ± 6.1	94.9 ± 0.3
Synth Avg.	62.2 ± 11.0	62.0 ± 9.9	**63.3 ± 9.3**	**66.7 ± 9.1**	**71.4 ± 11.0**	**74.5 ± 9.7**	93.7 ± 0.2

but the differences are not large enough to be significant. However, on both real and synthetic datasets, CALDG-Any,R and CALDG-XS,H are significantly better than CoDATS-DG, Sleep-DG, and AFLAC-DG ($p < 0.01$). On the synthetic datasets, they are similarly better than No Adaptation ($p < 0.01$). In contrast, on the real-world datasets, No Adaptation performs the best on average, though not significantly different than CALDG-Any,R. From these experiments, we conclude that the unlabeled target domain data makes a significant contribution to our proposed CALDA method. In addition, contrastive learning appears to benefit the problem of domain generalization as well as domain adaptation, though we leave a more detailed investigation to future work.

6 Conclusions and Future Work

We propose a novel time series MS-UDA framework CALDA, drawing on the principles of both adversarial and contrastive learning. This approach seeks to improve transfer by leveraging cross-source information, which is ignored by prior work time series work. We investigate design decisions for incorporating contrastive learning into multi-source domain adaptation, including how to select examples from multiple domains,

whether to include the target domain, and whether to utilize example difficulty. We observe that CALDA improves performance over prior work on a variety of real-world and synthetic time-series datasets both with and without weak supervision. In the weak supervision case, we additionally find the method is robust to label proportion noise. We also validated that both the adversary and unlabeled target domain data yield significant contribution to domain adaptation performance.

We observe the influence of hyperparameters on CALDA's performance. In a UDA setting, we do not rely on labeled source data to guide hyperparameter selection. Instead, methods such as those proposed by Saito et al. [34] and You et al. [35] can be leveraged to fine-tune CALDA's learned features toward a source density that maximizes discriminability.

In Section 3.1, we define time series as containing values that appear at uniform time intervals. In situations where time series data are unevenly spaced, such as electronic health record analysis or harmonizing multi-source data, data can be preprocessed to achieve uniformity [53]. Alternatively, the model can reason about the non-uniform time delays by incorporating them into the model as a separate parameter [54], [55].

Additional future work includes examining the use of CALDA for cross-modality adaptation [56]. We will also develop data augmentation compatible with time series domain adaptation and pseudo labeling techniques viable for the large domain gaps observed in time series to see if these yield further improvements in transfer.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1543656 and by the National Institutes of Health under Grant No. R01EB009675.

References

[1] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 5, Jul. 2020.

[2] G. Wilson, J. R. Doppa, and D. J. Cook, "Multi-source deep domain adaptation with weak supervision for time-series sensor data," in *KDD*, 2020, p. 1768–1778.

[3] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu, "Variational adversarial deep domain adaptation for health care time series analysis," in *ICLR*, 2017.

[4] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones." in *ESANN*, 2013.

[5] M. Zhao *et al.*, "Learning sleep stages from radio signals: A conditional adversarial architecture," in *ICML*, vol. 70, 2017, pp. 4100–4109.

[6] Y. Ganin, E. Ustinova *et al.*, "Domain-adversarial training of neural networks," *JMLR*, vol. 17, no. 59, pp. 1–35, 2016.

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020, pp. 1597–1607.

[8] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, 2020, pp. 9729–9738.

[9] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot *et al.*, "Supervised contrastive learning," *arXiv:2004.11362*, 2020.

[10] C. Park, J. Lee, J. Yoo, M. Hur, and S. Yoon, "Joint contrastive learning for unsupervised domain adaptation," *arXiv:2006.10297*, 2020.

[11] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *CVPR*, 2019, pp. 4893–4902.

[12] J. Choi, M. Jeong, T. Kim, and C. Kim, "Pseudo-labeling curriculum for unsupervised domain adaptation," *arXiv:1908.00262*, 2019.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *CVPR*, 2015, pp. 815–823.

[14] T. Cai, J. Frankle, D. J. Schwab *et al.*, "Are all negatives created equal in contrastive instance discrimination?" *arXiv:2010.06682*, 2020.

[15] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial multiple source domain adaptation," in *NeurIPS*, 2018, pp. 8559–8570.

[16] C.-X. Ren, Y.-H. Liu *et al.*, "Multi-source unsupervised domain adaptation via pseudo target domain," *IEEE Transactions on Image Processing*, vol. 31, pp. 2122–2135, 2022.

[17] Y. Li, L. Yuan, Y. Chen, P. Wang, and N. Vasconcelos, "Dynamic transfer for multi-source domain adaptation," in *Computer Vision and Pattern Recognition*, 2021.

[18] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig, "Controllable invariance through adversarial feature learning," in *NeurIPS*, 2017, pp. 585–596.

[19] F. Ott, D. Rugamer *et al.*, "Domain adaptation for time-series classification to mitigate covariate shift," in *ACM International Conference on Multimedia*, 2022, pp. 5934–5943.

[20] Y. Shi, X. Ying, and J. Yang, "Deep unsupervised domain adaptation with time series sensor data: A survey," *Sensors*, vol. 22, p. 5507, 2022.

[21] R. Cai, J. Chen, Z. Li, W. Chen, K. Zhang, J. Ye, Z. Li, X. Yang, and Z. Zhang, "Time series domain adaptation via sparse associative structure alignment," in *AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6859–6867.

[22] A. Hussein and H. Hajj, "Domain adaptation with representation learning and nonlinear relation for time series," *ACM Transactions on Internet of Things*, vol. 3, no. 2, p. 12, 2022.

[23] Q. Liu and H. Xue, "Adversarial spectral kernel matching for unsupervised time series domain adaptation," in *International Joint Conference on Artificial Intelligence*, 2021, pp. 2744–2750.

[24] I. Ketykó, F. Kovács, and K. Varga, "Domain adaptation for sEMG-based gesture recognition with recurrent neural networks," in *IJCNN*, 2019, pp. 1–7.

[25] A. Ameri, M. A. Akhaee *et al.*, "A deep transfer learning approach to reducing the effect of electrode shift in emg pattern recognition-based control," *IEEE NSRE*, vol. 28, no. 2, pp. 370–379, 2019.

[26] U. Côté-Allard, C. L. Fall *et al.*, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 4, pp. 760–771, 2019.

[27] Y. Du, W. Jin *et al.*, "Surface emg-based inter-session gesture recognition enhanced by deep domain adaptation," *Sensors*, vol. 17, no. 3, p. 458, 2017.

[28] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognition*, vol. 80, pp. 109 – 117, 2018.

[29] K. Ganchev, J. Gillenwater, B. Taskar *et al.*, "Posterior regularization for structured latent variable models," *JMLR*, vol. 11, no. Jul, pp. 2001–2049, 2010.

[30] W. Jiang, C. Miao, F. Ma, S. Yao, Y. Wang, Y. Yuan *et al.*, "Towards environment independent device free human activity recognition," in *MobiCom*, 2018, pp. 289–304.

[31] N. Hu, G. Englebienne, Z. Lou, and B. Kröse, "Learning to recognize human activities using soft labels," *IEEE PAMI*, vol. 39, no. 10, pp. 1973–1984, Oct 2017.

[32] D. Pathak, P. Krahenbuhl, and T. Darrell, "Constrained convolutional neural networks for weakly supervised segmentation," in *ICCV*, Dec 2015.

[33] M.-C. Dinu, M. Holzleitner *et al.*, "Addressing parameter choice issues in unsupervised domain adaptation by aggregation," in *International Conference on Learning Representations (ICLR)*, 2023.

[34] K. Saito, D. Kim *et al.*, "Tune it the right way: Unsupervised validation of domain adaptation via soft neighborhood density," in *ICCV*, 2021.

[35] K. You, X. Wang, M. Long, and M. I. Jordan, "Towards accurate model selection in deep unsupervised domain adaptation," in *ICML*, 2019.

[36] E. P. Xing, A. Y. Ng *et al.*, "Distance metric learning with application to clustering with side-information," in *NeurIPS*, vol. 15, no. 505–512, 2002, p. 12.
[37] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, vol. 2, 2006, pp. 1735–1742.
[38] G. French *et al.*, "Self-ensembling for visual domain adaptation," in *ICLR*, 2018.
[39] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *Plos one*, vol. 16, no. 7, p. e0254841, 2021.
[40] M. Thota and G. Leontidis, "Contrastive domain adaptation," in *CVPR*, 2021, pp. 2209–2218.
[41] P. Su, S. Tang, P. Gao, D. Qiu, N. Zhao, and X. Wang, "Gradient regularized contrastive learning for continual domain adaptation," *arXiv preprint arXiv:2007.12942*, 2020.
[42] J. Guo, D. Shah, and R. Barzilay, "Multi-source domain adaptation with mixture of experts," in *EMNLP*, 2018, pp. 4694–4703.
[43] H. I. Fawaz, G. Forestier *et al.*, "Deep learning for time series classification: a review," *DMKD*, vol. 33, no. 4, pp. 917–963, 2019.
[44] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *IJCNN*, 2017, pp. 1578–1585.
[45] A. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv:1807.03748*, 2018.
[46] H. He, O. Queen, T. Koker, C. Cuevas, T. Tsiligkaridis, and M. Zitnik, "Domain adaptation for time series under feature and label shifts," 2023. [Online]. Available: https://arxiv.org/abs/2302.03133
[47] A. Stisen *et al.*, "Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition," in *SenSys*, 2015, pp. 127–140.
[48] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor. Newsl.*, vol. 12, no. 2, pp. 74–82, 2011.
[49] J. W. Lockhart, G. M. Weiss, J. C. Xue *et al.*, "Design considerations for the wisdm smart phone-based sensor mining architecture," in *SensorKDD*, 2011, pp. 25–33.
[50] S. Pizzolato, L. Tagliapietra, M. Cognolato, M. Reggiani, H. Müller, and M. Atzori, "Comparison of six electromyography acquisition setups on hand movement classification tasks," *PloS one*, vol. 12, no. 10, 2017.
[51] G. Blanchard, G. Lee, and C. Scott, "Generalizing from several related classification tasks to a new unlabeled sample," in *NeurIPS*, 2011, pp. 2178–2186.
[52] K. Akuzawa, Y. Iwasawa, and Y. Matsuo, "Adversarial invariant feature learning with accuracy constraint for domain generalization," *arXiv:1904.12543*, 2019.
[53] A. Bharambe and D. Kalbande, "Self-organizing data processing for time series using spark," *Mobile Computing and Sustainable Informatics*, vol. 534, pp. 239–248, 2021.
[54] T. Braun, C. N. Fernandez, D. Eroglu, A. Hartland, S. F. M. Breitenbach, and N. Marwan, "Sampling rate-corrected analysis of irregularly sampled time series," *Physical Review E*, vol. 105, no. 024206, 2022.
[55] J. Koss, S. Tinaz, and H. D. Tagare, "Hierarchical denoising of ordinal time series of clinical scores," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 7, pp. 3507–3516, 2022.
[56] S. Deldari, H. Xue, A. Saeed, D. V. Smith, and F. D. Salim, "Cocoa: Cross modality contrastive learning for sensor data," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 6, pp. 1–28, 2022.

Garrett Wilson received his BS in Engineering from Walla Walla University in 2016. He received a PhD in Computer Science from Washington State University. His research interests include transfer learning, generative adversarial networks, and time series.

Janardhan Rao Doppa is a Huie-Rogers Endowed Chair Associate Professor at Washington State University. He received his PhD from Oregon State University. His research interests include machine learning with a focus on small-data setting and robustness.

Diane J. Cook is a Regents and Huie-Rogers Professor at Washington State University. She received her PhD from the University of Illinois. Her research interests include machine learning, pervasive computing, and design of automated strategies for health monitoring and intervention.