Multi-Source Deep Domain Adaptation with Weak Supervision for Time-Series Sensor Data

Garrett Wilson
garrett.wilson@wsu.edu
Washington State University
Pullman, WA, USA

Janardhan Rao Doppa
jana.doppa@wsu.edu
Washington State University
Pullman, WA, USA

Diane J. Cook
djcook@wsu.edu
Washington State University
Pullman, WA, USA

ABSTRACT

Domain adaptation (DA) offers a valuable means to reuse data and models for new problem domains. However, robust techniques have not yet been considered for time series data with varying amounts of data availability. In this paper, we make three main contributions to fill this gap. First, we propose a novel *Convolutional deep Domain Adaptation model for Time Series data (CoDATS)* that significantly improves accuracy and training time over state-of-the-art DA strategies on real-world sensor data benchmarks. By utilizing data from multiple source domains, we increase the usefulness of CoDATS to further improve accuracy over prior single-source methods, particularly on complex time series datasets that have high variability between domains. Second, we propose a novel *Domain Adaptation with Weak Supervision (DA-WS)* method by utilizing weak supervision in the form of target-domain label distributions, which may be easier to collect than additional data labels. Third, we perform comprehensive experiments on diverse real-world datasets to evaluate the effectiveness of our domain adaptation and weak supervision methods. Results show that CoDATS for single-source DA significantly improves over the state-of-the-art methods, and we achieve additional improvements in accuracy using data from multiple source domains and weakly supervised signals.

CCS CONCEPTS

• **Computing methodologies** → **Transfer learning**; *Unsupervised learning*; *Adversarial learning*; *Neural networks*; • **Mathematics of computing** → *Time series analysis*.

KEYWORDS

transfer learning, domain adaptation, time series, human activity recognition, weak supervision

ACM Reference Format:
Garrett Wilson, Janardhan Rao Doppa, and Diane J. Cook. 2020. Multi-Source Deep Domain Adaptation with Weak Supervision for Time-Series Sensor Data. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining USB Stick (KDD '20), August 23–27, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3394486.3403228

1 INTRODUCTION

Time series sensor data abound in many real-world settings including human activity recognition [1], sleep stage classification [43], gesture recognition [26], speech recognition [17], and diagnosis and mortality prediction from medical data [32]. Labeling data in these situations is expensive and sometimes infeasible. One way to reduce labeling effort is to design unsupervised domain adaptation techniques that leverage the labeled data from one or more source domains and unlabeled data from a new target domain to build a classifier for the target domain [12, 41].

While unsupervised domain adaptation methods have been designed for image data, very limited work has focused on adaptation approaches for time series data [40]. A few time series methods have been introduced. However, these prior approaches utilize recurrent neural networks (RNNs) that can be very slow to train for reasonable-sized time series arising in real-world problems. While researchers have found that convolutional neural networks (CNNs) can achieve the same accuracy as RNNs while being trained and evaluated much faster [2, 28], previously-proposed domain adaptation network architectures are incompatible with time series data.

In this paper, we propose a new model: **Co**nvolutional deep **D**omain **A**daptation model for **T**ime **S**eries data (CoDATS). CoDATS couples technical principles from domain-invariant domain adaptation with a network design that is more efficient, accurate, time-series compatible, and extensible than prior work. The CoDATS architecture exhibits three important features. First, it leverages existing domain-invariant domain adaptation methods to operate on time series data. Second, it outperforms existing single-source time series adaptation models. Third, it is readily extensible to additional situations including when data from multiple source domains is available, which is particularly helpful for complex time series datasets having high variability between domains, and when the target-domain label distribution is available, which may be easier to collect than additional time series data labels.

While utilizing unlabeled target-domain data in unsupervised domain adaptation is one way of reducing labeling effort, another way is to use weakly-supervised information that is relatively easy to acquire. Obtaining labels for time series sensor data is more challenging than for image data. For example, cat vs. dog image classification labels can be easily obtained after data collection by having a person look at each image and determine if the image is of a cat or a dog. In contrast, for time series human activity recognition, it is much more difficult to identify what activity a human was performing by looking at raw accelerometer, gyroscope, and magnetometer sensor data. Thus, labels for time series sensor data are instead typically recorded while performing the activity [10, 29], which greatly limits the number of gathered labels because

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '20, August 23–27, 2020, Virtual Event, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7998-4/20/08...$15.00
https://doi.org/10.1145/3394486.3403228

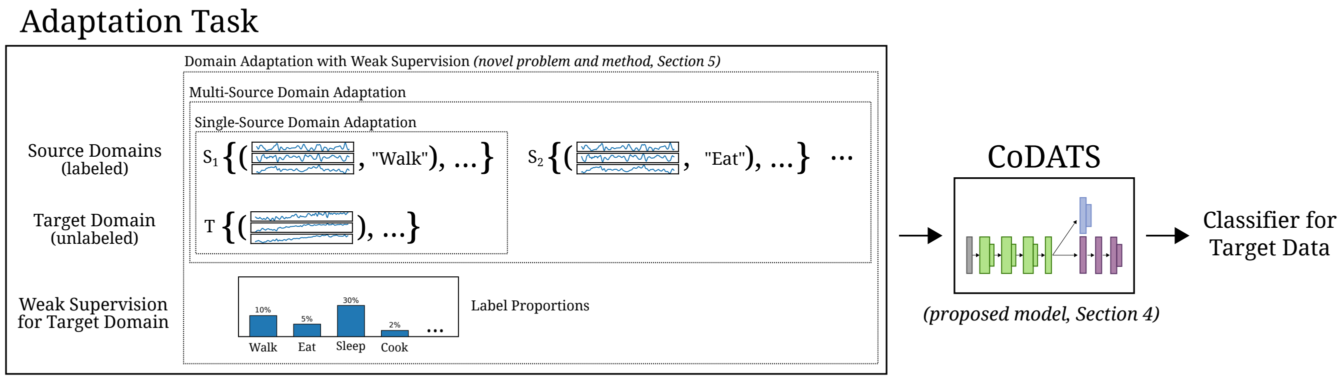

Adaptation Task
Domain Adaptation with Weak Supervision (novel problem and method, Section 5)
Multi-Source Domain Adaptation
Single-Source Domain Adaptation
Source Domains (labeled)
Target Domain (unlabeled)
Weak Supervision for Target Domain
S1 {(, "Walk"), ...}
S2 {(, "Eat"), ...} ...
T {( ), ...}
10%
5%
30%
2%
Walk
Eat
Sleep
Cook
...
Label Proportions
CoDATS
(proposed model, Section 4)
Classifier for Target Data

Figure 1: Illustration of our proposed CoDATS model that supports single- and multi-source domain adaptation in addition to our novel problem setting of domain adaptation with weak supervision.

of the additional burden from interrupting a person's activities. However, other possibilities exist for obtaining information in the form of weak supervision. For activity recognition, it may be easy for each participant to self-report what proportion of the time they performed each activity. For example, providing an estimate of how many hours a day they spend cooking is easier than labeling each data instance of cooking. We formulate this new problem setting of **D**omain **A**daptation with **W**eak **S**upervision (DA-WS) and develop a novel method to effectively utilize weak supervision in the form of label proportions. The key idea is to constrain the space of model parameters to those which approximately matches the label proportions on unlabeled data from the target domain.

To validate our proposed CoDATS model and weak-supervision method, we performed comprehensive experiments on diverse real-world time series benchmarks including gesture recognition and human activity recognition. We compare CoDATS with prior single-source time series methods and observe that CoDATS dramatically outperforms previous approaches to time series domain adaptation. Additionally, we demonstrate how CoDATS can further improve accuracy by utilizing data from multiple sources. We also find that coupled with our proposed CoDATS model, our DA-WS method yields additional improvements in accuracy.

Contributions. We make three key contributions, as summarized in Figure 1. **1)** We develop a new time-series compatible model referred as CoDATS to improve both accuracy and computational-efficiency when compared to prior work on single-source DA. CoDATS supports utilizing data from multiple sources to further improve accuracy. **2)** We formulate a new weak supervision problem to leverage target-domain label proportions when available and propose a novel method referred as DA-WS to effectively solve it. **3)** We perform comprehensive experimental evaluation on multiple challenging real-world benchmarks to show the efficacy of our CoDATS model and weak-supervision method over state-of-the-art.[1]

2 RELATED WORK

A large body of prior work has contributed numerous strategies for single-source domain adaptation of image data. These methods learn a domain-invariant feature representation [12, 22, 33] or perform domain mapping [4, 14, 16]. Some researchers explored using separate normalization statistics for each domain [25], building ensembles [9], or pushing decision boundaries into lower-density regions [23, 34]. While multi-source domain adaptation, in which multiple source domains are adapted to a single target, has not received the same level of attention, a few methods have been proposed. One straightforward method simply combines multiple domains with labeled data together as a single "source" domain [36]. More sophisticated methods take advantage of differences among source domains or the relationship between sources and target. Zhao et al. [42] include a separate domain classifier for each source domain and compute a loss based on the lowest domain error among these classifiers. Xie et al. [41] instead propose using a multi-class output for a single domain classifier, which scales better as the number of source domains increases. These previous single-source and multi-source adaptation algorithms cannot be directly applied to time series data without a time-series compatible model architecture, such as our proposed CoDATS model.

While less numerous, researchers have introduced a few time series adaptation methods. Specifically, variational recurrent adversarial deep domain adaptation (VRADA) and recurrent domain adversarial neural network (R-DANN) explore RNNs as feature extractors. These RNNs can be combined with an adversary to make a time series representation domain invariant [32]. R-DANN employs a long short-term memory (LSTM) network, and VRADA uses a variational RNN. These were tested on time series medical data, while a variation employing gated recurrent units (GRUs) predicted driving maneuvers [37].

A few domain adaptation methods have tackled specialized time series tasks. These include instance weighting for anomaly detection [38] and a domain-invariant feature learner for inertial tracking that combined an autoencoder, a generative adversarial network (GAN), and RNNs [5]. Another domain-invariant method combined CNNs and RNNs to perform domain generalization for classification of sleep stages from radio-frequency data [43]. In a related method, source datasets with differing label spaces were transferred to one target dataset using pre-training and fine-tuning [6].

We include the two most related approaches, VRADA and R-DANN, as baseline models in our empirical study validating our

[1] Code is available at: https://github.com/floft/codats

proposed CoDATS model since they were developed for time series classification and thus can be directly used for comparison. However, we do not include the others since they solve related but different problems or are incompatible with time series data.

Our proposed domain adaptation with weak supervision approach is inspired in part by the posterior regularization idea from Ganchev et al., in which learning is guided to meet certain constraints on the unlabeled data [11]. However, our work differs from that of Ganchev et al. in three key ways. First, they focus on developing methods for natural language processing, whereas we focus on developing appropriate methods and models compatible with time series. Second, they develop weak supervision for use alone or in tandem with some additional labeled data for semi-supervised learning, whereas we focus on coupling weak supervision with domain adaptation. Third, they develop a weak supervision method based on expectation maximization, whereas because we are using deep neural networks, we develop a weak supervision objective that instead takes the form of a differentiable regularization term added to the loss function. A related regularizer was used by Jiang et al. [20], but they estimate label proportions on the source domain data rather than using target-domain weak supervision.

A few other weak supervision methods have been proposed in different contexts. Hu et al. [18] learn human activity recognition models from video data that may have multiple, uncertain, or incomplete labels. We instead propose a form of weak supervision better suited to streams of raw sensor data that cannot be easily visually identified. Huang et al. [19] incorporate expert-provided biases into a hidden Markov model (HMM) for domain adaptation in a natural language processing context. The biases provide information about a specific task in an application domain (akin to feature engineering) rather than information about the target domain data, which may differ for each target domain. In the context of semantic segmentation of images, a variety of work has explored using weak supervision [15, 31]. However, such weak supervision takes the form of image labels or bounding boxes, which cannot be directly applied to time series sensor data. Despite the fundamental differences between these methods and the problem we are addressing, the performance gains suggest that weak supervision in a time series domain adaptation context may similarly yield an improvement in accuracy.

3 PROBLEM SETUP

Domain adaptation reuses labeled data from one domain to create a classifier for a different but related domain. In time series data settings such as human activity recognition, this is a common problem. For example, activity-labeled sensor data may exist for person 1 while only unlabeled data is available for person 2. Because of inherent differences (e.g., different activity patterns, sensor positions, or sampling rates), a classifier trained on person 1's data will likely not perform well for person 2. However, we can create a feature extractor that produces domain-invariant features. A classifier trained on these features will generalize better to person 2 because the features are similar in both domains.

Additionally, we may have other sources of information about person 2's activities. For example, they may self-report how much time they spend on each activity, which can be interpreted as label proportions for the target domain. This can be used as a constraint or regularizer for the learned model: learn a model that is domain invariant but also consistent with the known label proportions.

Below we provide definitions for two cases of unsupervised domain adaptation: single-source domain adaptation and multi-source domain adaptation. Following this, we formulate our novel problem setting of domain adaptation with weak supervision.

Single/Multi-Source Domain Adaptation. Formally, given input data X with L labels $Y = \{1, 2, \ldots, L\}$ and n source domains, then we have several distributions over the input and label space $X \times Y$: n source domain distributions $\mathcal{D}_{S_i}$ for $i \in \{1, 2, \ldots, n\}$ and a target domain distribution $\mathcal{D}_T$. During training, we draw labeled samples i.i.d. from each source domain distribution $\mathcal{D}_{S_i}$ and unlabeled samples i.i.d. from $\mathcal{D}_T^X$ (the marginal distribution of $\mathcal{D}_T$ over X). This gives us s_i source samples S_i for each source domain $i \in \{1, 2, \ldots, n\}$ and t target samples T:

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

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

Single-source domain adaptation corresponds to the case in which $n = 1$ and multi-source domain adaptation corresponds to the case where $n > 1$. During evaluation, we test the learned model on t samples T_{test} drawn i.i.d. from $\mathcal{D}_T$ and not seen during training (we need ground-truth labels to compute accuracy).

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

In this paper, we specifically investigate the case where X represents time series data. Time series can be either univariate or multivariate. If univariate, then $X = [x_1, x_2, \ldots, x_H]$ consists of H ordered real values. If multivariate, then $X = [X^1, X^2, \ldots, X^K]$ consists of K univariate time series, each of which contains H ordered real feature values [7]. For example, these features may be accelerometer x, y, and z values, each of which is a time series of real values.

Domain Adaptation with Weak Supervision. For weak supervision with known target-domain label proportions, the setup adheres to the above domain adaptation definition, except additional information is now available for training. Formally, we are provided with a discrete probability distribution $P(Y = y)$ giving the probability p_y that a target-domain example will have the label y:

$$Y_{true}(y) = P(Y = y) = p_y, \quad \forall y \in \{1, 2, \ldots, L\} \tag{4}$$

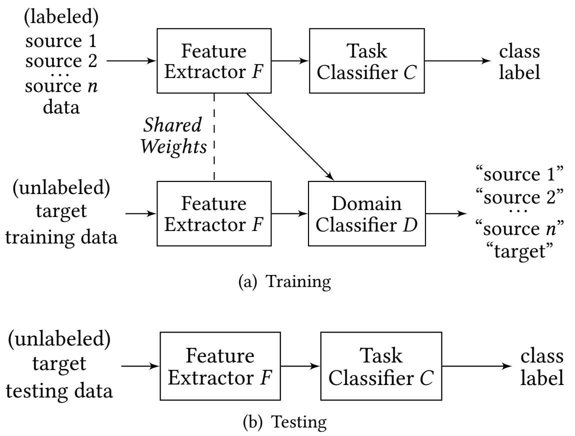
Note that because this is a probability distribution, $\sum_{y=1}^{L} p_y = 1$ and $p_y \geq 0$ for all $y \in \{1, 2, \ldots, L\}$.

4 CODATS FOR DOMAIN ADAPTATION

We propose a time series domain adaptation model to improve adaptation performance. Here, we first summarize the proposed adaptation method that supports both single-source and multi-source domain adaptation. Next, we detail the network architecture we develop that is compatible with time series data.

4.1 Adaptation Method

We propose a domain-invariant training method for CoDATS. The goal of domain-invariant methods is to learn a feature extractor network that produces domain-invariant features. Such features



(a) Training

(b) Testing

Figure 2: CoDATS training and testing setup. Its goal is to learn a domain-invariant feature representation during training that is then used during testing.

can be learned with a domain classifier (or "discriminator") acting as an adversary. If a classifier trained on these domain-invariant features performs well in the source domain, then this classifier may generalize well to the target domain due to the domain-aligned feature distributions. Ultimately, generalization also depends on the level of similarity between domains. If the domains are too distinct, learning domain-invariant features can increase error [3]. The most common domain-invariant method is the domain adversarial neural network (DANN) [12], which is the basis of the prior time-series adaptation works, VRADA and R-DANN. However, note that DANN only works for single-source domain adaptation.

Xie et al. [41] extend DANN to a more general framework, supporting invariance from factors or traits other than binary domain labels. While their framework allows for multinomial variables, continuous variables, or variables with structure such as parse trees, for multi-source domain adaptation (or single-source, which is a special case) we are particularly interested in the case of multinomial variables. Thus, we base our adaptation method on this framework.

Our training setup illustrated in Figure 2 consists of three neural networks: a feature extractor F, a task classifier C, and a domain classifier D. The feature extractor is updated in an adversarial manner, with two competing objectives. The feature extractor and task classifier are updated such that the task classifier correctly classifies labeled source data. Similarly, the domain classifier is updated in such a way that it accurately classifies data as coming from the correct domain. At the same time, the feature extractor is also updated such that the domain classifier cannot classify which domain the data originated from (the adversarial component). The adversarial step is performed via a gradient reversal layer (GRL) [12] placed in the network between the feature extractor and domain classifier. The GRL flips the gradient during back propagation when updating the weights to achieve this effect.

Formally, given a discriminator $D(\cdot;\theta_d)$ with parameters θ_d, a feature extractor $F(\cdot;\theta_f)$ with parameters θ_f, a task classifier $C(\cdot;\theta_c)$ with parameters θ_c, labeled source domain data $\mathcal{D}_{S_i}$ from a source domain $i \in \{1, 2, \ldots n\}$, unlabeled target domain data $\mathcal{D}_T^X$, two multi-class cross entropy losses $\mathcal{L}_y$ and $\mathcal{L}_d$ for the labels and domains respectively, and domain labels for each example where the target domain is labeled $d_T = 0$ and the source domains are labeled $d_{S_i} = i$ for $i \in \{1, 2, \ldots n\}$, then the two competing objectives are:

$$\underset{\theta_f,\theta_c}{\operatorname{argmin}} \sum_{i=1}^{n} \mathbb{E}_{(\mathrm{x},y)\sim\mathcal{D}_{S_i}}\left[\mathcal{L}_y(C(F(\mathrm{x})),y) - \mathcal{L}_d(D(F(\mathrm{x})),d_{S_i})\right] - \mathbb{E}_{\mathrm{x}\sim\mathcal{D}_T^X}\left[\mathcal{L}_d(D(F(\mathrm{x})),d_T)\right] \quad (5)$$

$$\underset{\theta_d}{\operatorname{argmin}} \sum_{i=1}^{n} \mathbb{E}_{(\mathrm{x},y)\sim\mathcal{D}_{S_i}}\left[\mathcal{L}_d(D(F(\mathrm{x})),d_{S_i})\right] + \mathbb{E}_{\mathrm{x}\sim\mathcal{D}_T^X}\left[\mathcal{L}_d(D(F(\mathrm{x})),d_T)\right] \quad (6)$$

To summarize, Equation 5 updates the parameters of F and C such that C predicts the correct source task labels and D predicts the incorrect domain labels (the $\mathcal{L}_d$ terms are negated), while Equation 6 updates D to correctly predict the domain labels.

However, using the gradient reversal layer, Equations 5 and 6 can be combined together into one step. The gradient reversal layer can be represented as $\mathcal{R}(\mathrm{x})$ with different forward and backward propagation behavior, where I is the identity matrix and λ is a constant (possibly with a specified schedule during training):

$$\mathcal{R}(\mathrm{x}) = \mathrm{x}; \quad \frac{d\mathcal{R}}{d\mathrm{x}} = -\lambda \mathrm{I} \quad (7)$$

Then, the optimization step becomes:

$$\underset{\theta_f,\theta_c,\theta_d}{\operatorname{argmin}} \sum_{i=1}^{n} \mathbb{E}_{(\mathrm{x},y)\sim\mathcal{D}_{S_i}}\left[\mathcal{L}_y(C(F(\mathrm{x})),y) + \mathcal{L}_d(D(\mathcal{R}(F(\mathrm{x}))),d_{S_i})\right] + \mathbb{E}_{\mathrm{x}\sim\mathcal{D}_T^X}\left[\mathcal{L}_d(D(\mathcal{R}(F(\mathrm{x}))),d_T)\right] \quad (8)$$

Equation 8 yields an adaptation method capable of handling both single-source ($n = 1$) and multi-source ($n > 1$) domain adaptation.

4.2 Model

We now need to design time-series compatible network architectures capable of handling the shape and temporal nature of time series data. In contrast to image data which are typically represented by three dimensions (height, width, number of channels) having spatial relationships along two dimensions, time series data typically have two dimensions (time series length, number of features) with temporal relationships between sensor data across time. The developed network must be capable of learning dependencies along this time dimension.

Prior works used RNNs for time series because of their ability to handle sequential data. However, RNNs pose a number of challenges: long-term dependencies are inhibited because gradients backpropagated through time tend to vanish and training can fail if gradients explode. Gating mechanisms have been developed to partially address vanishing gradients [2], and exploding gradients can be partially resolved by gradient clipping [30]. Others have tried developing superior RNNs, but these have been found to not outperform LSTMs for many tasks [13, 21, 27]. An alternative direction for improving sequential models is instead employing feed-forward networks like CNNs, using 1D convolutions along the time axis. Stable RNNs have provably-good feed-forward approximations [28],

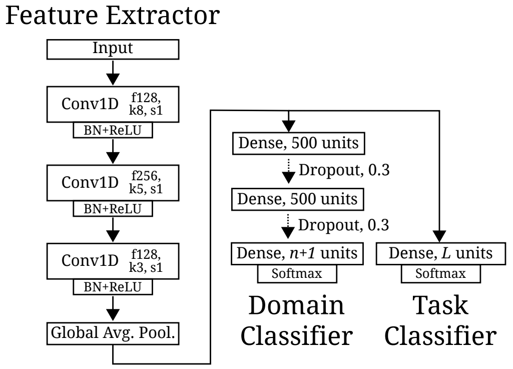


Figure 3: CoDATS model architecture. Each convolution layer uses same padding. fx indicates x filters, kx a kernel of size x, and sx a stride of x. There are $n+1$ domain classifier outputs due to n source domains and 1 target domain.

and researchers have empirically demonstrated the benefit of CNNs on sequential data, for example, performing better at long-term memory than RNNs [2].

While time-series adaptation traditionally employs RNNs, we hypothesize that both accuracy and computation cost can benefit from a CNN architecture. We propose the CoDATS networks summarized in Figure 3. The feature extractor consists of a fully convolutional network (FCN) [39] that is comparable to more computationally-expensive, state-of-the-art ensemble time series classification models [7, 39]. A similar FCN model has previously been used for other time series transfer learning tasks [6]. The single dense layer (the last layer) of the FCN then acts as the task classifier. The domain classifier acting as the adversary during training consists of a multi-layer perceptron (MLP) model [39]. We employ MLP rather than additional 1D convolutions since following the global average pooling layer in the feature extractor, there is no longer any time dimension. Thus, additional 1D convolutions along the time axis are not possible. Other works have similarly used multiple dense layers for a domain classifier [12, 32].

Similar to an RNN, the proposed network can handle variable-length time series data. RNNs handle variable-length data by processing each time step sequentially, whereas the proposed CNN network handles variable-length data with the global average pooling layer. While the data in our experiments are of a fixed length, the proposed model is not limited to such situations.

5 DA WITH WEAK SUPERVISION

We propose a novel method for domain adaptation with weak supervision (DA-WS) based on our CoDATS model. The general principle behind our learning approach is to use the weak supervision information as constraints to efficiently search for good model parameters. We can devise specific learning methods depending on the form of weak supervision by appropriately instantiating this general principle. DA-WS incorporates the known target-domain label proportion information during training by searching for model parameters whose predicted labels on unlabeled target domain data approximately matches with the given label proportions. Mathematically, this is done by introducing a differentiable regularization term to the training objective.

From our formalization of the problem in Equation 4, we have the true label proportions $Y_{true}(y)$ available for the target domain. Now let us define the predicted target-domain distribution from the CoDATS task classifier. Let $C(F(\mathrm{x}))_y$ denote the element of the task classifier's softmax prediction corresponding to label y. Then, the predicted label distribution can be represented as a probability distribution $P(Y = y|X = \mathrm{x})$, indicating the probability that a target-domain instance $\mathrm{x} \sim \mathcal{D}_T^X$ will have label y:

$$Y_{pred}(y) = P(Y_{pred} = y) = C(F(\mathrm{x}))_y, \quad \forall y \in \{1, 2, \ldots, L\} \tag{9}$$

We do *not* want each individual prediction to have output probabilities matching $Y_{true}(y)$, which would regularize to always predict the majority class (the maximum probability in the $Y_{true}(y)$ discrete distribution). Rather, we want the expected value computed over a large number of predictions (e.g., a mini-batch) to follow this distribution. We want C to predict a particular class for each x, but on average the predictions should follow $Y_{true}(y)$. This gives us the estimated distribution $\tilde{Y}_{pred}(y)$ over multiple predictions:

$$\tilde{Y}_{pred}(y) = \mathbb{E}_{\mathrm{x} \sim \mathcal{D}_T^X}\left[Y_{pred}(y)\right] = \mathbb{E}_{\mathrm{x} \sim \mathcal{D}_T^X}\left[C(F(\mathrm{x}))\right] \tag{10}$$

During training, we can align the predicted distribution $\tilde{Y}_{pred}(y)$ with the known distribution $Y_{true}(y)$ with a distance measure such as Kullback-Leibler (KL) divergence. Thus, we propose adding the following weak supervision regularization term to the loss:

$$\begin{aligned} \mathcal{L}_{WS} &= D_{KL}\Big(Y_{true}(y) \,\Big\|\, \tilde{Y}_{pred}(y)\Big) \\ &= D_{KL}\Big(Y_{true} \,\Big\|\, \mathbb{E}_{\mathrm{x} \sim \mathcal{D}_T^X}\left[C(F(\mathrm{x}))\right]\Big) \end{aligned} \tag{11}$$

Combining our regularization term in Equation 11 with the CoDATS training objective in Equation 8, the entire training objective for DA-WS becomes:

$$\begin{aligned} \underset{\theta_f, \theta_c, \theta_d}{\operatorname{argmin}} \quad & \sum_{i=1}^{n} \mathbb{E}_{(\mathrm{x},y) \sim \mathcal{D}_{S_i}}\left[\mathcal{L}_y(C(F(\mathrm{x})), y) + \mathcal{L}_d(D(\mathcal{R}(F(\mathrm{x}))), d_{S_i})\right] \\ & + \mathbb{E}_{\mathrm{x} \sim \mathcal{D}_T^X}\left[\mathcal{L}_d(D(\mathcal{R}(F(\mathrm{x}))), d_T)\right] \\ & + D_{KL}\Big(Y_{true} \,\Big\|\, \mathbb{E}_{\mathrm{x} \sim \mathcal{D}_T^X}\left[C(F(\mathrm{x}))\right]\Big) \end{aligned} \tag{12}$$

Altogether, this objective finds model parameters that: 1) predict correct task labels using the labeled data from the source domains, 2) produce a domain-invariant feature representation among all domains using the unlabeled target-domain data, and 3) predict target-domain task labels that on average align well with the known target-domain label proportions.

Note that this regularizer only has an effect when the label distribution differs between the source and target domains. If the distributions are the same, then when training with empirical risk minimization, $C(F(\mathrm{x}))$ will already produce a predicted label distribution aligning with that of the source(s), making the KL divergence term zero. We only gain information from knowing the target-domain label distribution Y_{true} if it differs from the sources' label distribution, and thus only in such situations will we see DA-WS yield an improvement. Indeed, our experimental results corroborate this proposition.

6 EXPERIMENTAL SETUP

First, to evaluate our proposed model, we compare CoDATS with relevant prior methods and several baselines on diverse time series adaptation benchmarks. We hypothesize that CoDATS will yield an improvement in accuracy while also reducing training time. We choose accuracy as the evaluation metric to afford comparison with prior adaptation work. While we initially compare on single-source adaptation because this is the problem studied by prior work, we also demonstrate that our model supports multi-source adaptation, which we hypothesize can yield additional performance gains.

Second, we perform experiments utilizing our proposed domain adaptation with weak supervision (DA-WS) method. We hypothesize that DA-WS coupled with our CoDATS model will provide further improvements in accuracy.

6.1 Datasets

A requirement for selecting datasets to test domain adaptation is that the dataset includes a property for splitting the data into multiple domains. Such properties could include different data collection days or different generating processes (e.g., different persons in the case of human activity recognition data). The multi-variate time series datasets we select include a participant identifier, and we use this feature to split data into multiple domains. Thus, our adaptation problems consist of the realistic use-case adapting a model from one or more participants' data to another participant's data.

We include the Human Activity Recognition (HAR) dataset [1] due to its popularity in time series research. It contains accelerometer, gyroscope, and estimated body acceleration data from 30 participants. We also include the Heterogeneity Human Activity Recognition Dataset (HHAR) dataset [35], which is more diverse than HAR, including accelerometer data from 31 smartphones of different manufacturers, models, etc. positioned in various orientations. Additionally, we include 33 participants' accelerometer data from the WISDM activity recognition (WISDM AR) dataset [24]. Finally, we include a gesture recognition dataset (uWave) containing accelerometer data from 8 participants performing various hand gestures [26]. In the single-source experiments, we pick 10 random pairs of participants for each dataset. In the multi-source experiments, for each value of n, we pick 3 random subsets of source domains for each of the 10 random target domains. See the Appendix for more details.

6.2 CoDATS Model

Single-Source Domain Adaptation. In our evaluation, we compare CoDATS with several single-source domain adaptation baselines. First, we include no adaptation as an approximate lower bound (`No Adaptation`). The lower bound gives a rough estimate of the domain gap between the source and target distributions and indicates how hard it will be to perform domain adaptation. We expect that domain adaptation methods will exceed this lower bound, depending on the extent of the domain gap. Similarly, we include models trained directly on labeled target data as an approximate upper bound (`Trained on Target`). Assuming there is enough target data, then we expect these upper-bound models to perform well when trained on this labeled data. As explained by the theory of Ben-David et al. [3], given enough target labels, not using any of the source domain data is actually preferable because of the possibility of negative transfer. Finally, we include `R-DANN` and `VRADA` [32] as single-source time series adaptation baselines using their respective network architectures. We compare these baselines with our CoDATS model first for single-source domain adaptation to demonstrate the superiority of our proposed model for time series.

Multi-Source Domain Adaptation. As with single-source domain adaptation, we include the baselines `No Adaptation` and `Trained on Target` in the multi-source domain adaptation experiments. This set of experiments demonstrate the additional performance gains achievable by utilizing data from multiple people. Our proposed CoDATS model supports such multi-source domain adaptation, whereas prior methods do not.

6.3 Domain Adaptation with Weak Supervision

For the domain adaptation with weak supervision problem setting, we are given additional target-domain label proportion information. We simulate being given this information by estimating the label proportions of the target domain data from only its training set. We evaluate our `DA-WS` method with both a single source domain and also multiple source domains. Since this is a novel problem setting, we do not compare with prior works but rather compare with `No Adaptation`, `Trained on Target`, and `CoDATS`. By comparing with these domain adaptation methods that do not make use of the additional weak supervision information, we determine whether we can successfully utilize this additional information to improve model performance. Note that because the DA-WS method also uses the CoDATS model, we use the term `CoDATS` to refer to the experiments using the model without weak supervision and the term `CoDATS-WS` to refer to the experiments using the model with weak supervision.

7 RESULTS AND DISCUSSION

Here we present experimental results along multiple dimensions.

7.1 Single-Source CoDATS

Table 1 summarizes target domain classifier performance on all the benchmark datasets for single-source domain adaptation. To compare training cost, the training times are listed in Table 2.

Lower and Upper Bounds. As expected, the lower bound (`No Adaptation`) performs poorly, providing evidence that target domain data differ from source domain data. Though, two notable exceptions are uWave 1 to 8 and uWave 4 to 8, where the lower bound reaches 100%. In this case, adaptation should not yield a negative effect. On uWave 1 to 8, CoDATS does achieve 100% accuracy. However, on uWave 4 to 8, it is close but not quite perfect accuracy – though, it is far closer than either R-DANN or VRADA. This finding may indicate that CoDATS is more resistant to negative transfer than previous approaches (though further exploration is necessary). As for the upper bound, there is nearly always a sufficient amount of labeled target data to achieve close to 100% accuracy.

Prior Methods. R-DANN and VRADA exhibit lower performance on the uWave dataset than the others. While the other datasets contain only 128 time steps, uWave contains 315. This result may indicate that RNN models experience difficulty adapting long time

Problem	No Adaptation	R-DANN	VRADA	*CoDATS*	*CoDATS-WS*	Train on Target
HAR 2 → 11	**83.3 ± 0.7**	80.7 ± 5.2	64.1 ± 5.6	74.5 ± 4.5	74.5 ± 6.0	100.0 ± 0.0
HAR 7 → 13	89.9 ± 3.6	75.3 ± 5.8	78.3 ± 5.2	**96.5 ± 0.7**	**96.5 ± 0.7**	100.0 ± 0.0
HAR 12 → 16	41.9 ± 0.0	35.1 ± 2.9	61.7 ± 7.5	**77.5 ± 0.6**	75.2 ± 3.5	100.0 ± 0.0
HAR 12 → 18	90.0 ± 1.7	74.9 ± 0.6	74.4 ± 6.7	**100.0 ± 0.0**	**100.0 ± 0.0**	100.0 ± 0.0
HAR 9 → 18	31.1 ± 1.7	56.6 ± 6.4	59.8 ± 10.1	**85.8 ± 1.7**	76.7 ± 6.8	100.0 ± 0.0
HAR 14 → 19	62.0 ± 4.3	71.3 ± 2.4	64.4 ± 4.7	72.2 ± 27.2	**98.6 ± 1.1**	100.0 ± 0.0
HAR 18 → 23	**89.3 ± 5.0**	78.2 ± 6.4	72.9 ± 6.0	86.2 ± 0.6	**89.3 ± 1.1**	100.0 ± 0.0
HAR 6 → 23	52.9 ± 2.3	79.1 ± 2.7	78.2 ± 6.4	**94.7 ± 1.1**	94.2 ± 1.3	100.0 ± 0.0
HAR 7 → 24	94.4 ± 2.7	84.8 ± 6.9	93.9 ± 0.6	**100.0 ± 0.0**	99.1 ± 0.6	100.0 ± 0.0
HAR 17 → 25	57.3 ± 5.5	66.3 ± 5.8	52.0 ± 1.1	96.7 ± 1.5	**97.6 ± 1.0**	100.0 ± 0.0
HAR Average	69.2 ± 21.8	70.2 ± 14.0	70.0 ± 11.4	88.4 ± 10.1	**90.2 ± 10.1**	100.0 ± 0.0
HHAR 1 → 3	77.8 ± 4.4	85.1 ± 3.9	81.3 ± 10.6	**93.2 ± 1.6**	90.8 ± 2.0	99.2 ± 0.0
HHAR 3 → 5	68.8 ± 5.2	85.4 ± 1.4	82.3 ± 5.9	**95.6 ± 0.9**	94.3 ± 1.2	99.0 ± 0.1
HHAR 4 → 5	60.4 ± 3.0	70.4 ± 3.1	71.6 ± 3.1	94.2 ± 1.1	**94.7 ± 0.5**	99.0 ± 0.1
HHAR 0 → 6	33.6 ± 2.2	33.4 ± 1.8	35.6 ± 5.1	**76.7 ± 1.5**	74.2 ± 1.1	98.8 ± 0.1
HHAR 1 → 6	72.1 ± 3.9	81.7 ± 3.0	74.9 ± 7.2	90.5 ± 0.7	**90.8 ± 0.2**	98.8 ± 0.1
HHAR 4 → 6	48.0 ± 2.6	64.6 ± 5.6	62.7 ± 10.3	**93.7 ± 0.4**	85.3 ± 10.6	98.8 ± 0.1
HHAR 5 → 6	65.1 ± 6.9	54.4 ± 1.1	60.0 ± 2.8	90.7 ± 2.3	**91.7 ± 0.4**	98.8 ± 0.1
HHAR 2 → 7	49.4 ± 2.1	46.4 ± 3.0	45.0 ± 12.2	**58.1 ± 4.5**	56.6 ± 3.4	98.5 ± 0.5
HHAR 3 → 8	77.8 ± 2.1	82.8 ± 1.4	82.2 ± 1.7	93.4 ± 0.4	**94.3 ± 1.0**	99.3 ± 0.0
HHAR 5 → 8	95.3 ± 0.4	82.5 ± 2.6	87.5 ± 0.9	**97.1 ± 0.3**	95.8 ± 0.2	99.3 ± 0.0
HHAR Average	64.8 ± 16.9	68.7 ± 17.6	68.3 ± 16.4	**88.3 ± 11.4**	86.8 ± 11.8	99.0 ± 0.3
WISDM AR 1 → 11	71.7 ± 0.0	55.6 ± 6.4	55.0 ± 11.6	71.7 ± 0.0	**93.3 ± 0.0**	98.3 ± 0.0
WISDM AR 3 → 11	6.7 ± 4.9	28.9 ± 7.5	45.0 ± 4.9	**47.8 ± 0.8**	46.7 ± 0.0	98.3 ± 0.0
WISDM AR 4 → 15	78.2 ± 4.5	69.2 ± 5.7	**82.7 ± 2.7**	81.4 ± 8.9	75.6 ± 6.3	100.0 ± 0.0
WISDM AR 2 → 25	81.1 ± 2.8	57.8 ± 5.5	72.2 ± 10.3	90.6 ± 1.6	**97.8 ± 0.8**	100.0 ± 0.0
WISDM AR 25 → 29	47.1 ± 8.2	61.6 ± 5.4	81.9 ± 2.7	74.6 ± 7.4	**84.8 ± 1.8**	95.7 ± 0.0
WISDM AR 7 → 30	62.5 ± 0.0	41.7 ± 5.1	61.9 ± 4.7	**73.2 ± 16.2**	70.2 ± 9.9	100.0 ± 0.0
WISDM AR 21 → 31	57.1 ± 0.0	61.0 ± 8.8	68.6 ± 8.1	68.6 ± 4.0	**92.4 ± 1.3**	97.1 ± 0.0
WISDM AR 2 → 32	60.1 ± 9.1	49.0 ± 16.2	66.7 ± 4.2	67.3 ± 0.9	**68.6 ± 1.6**	100.0 ± 0.0
WISDM AR 1 → 7	68.5 ± 2.3	44.8 ± 5.6	63.0 ± 6.0	**70.9 ± 0.0**	66.1 ± 6.9	96.4 ± 0.0
WISDM AR 0 → 8	34.7 ± 9.3	13.3 ± 2.5	14.7 ± 8.1	54.0 ± 15.6	**62.0 ± 15.7**	99.3 ± 0.9
WISDM AR Average	56.8 ± 21.3	48.3 ± 16.0	61.2 ± 18.9	70.0 ± 11.6	**75.8 ± 15.4**	98.5 ± 1.6
uWave 2 → 5	86.3 ± 1.8	33.3 ± 12.0	18.5 ± 8.4	83.6 ± 12.1	**98.2 ± 1.9**	100.0 ± 0.0
uWave 3 → 5	82.7 ± 1.1	63.7 ± 5.3	32.4 ± 14.3	**93.8 ± 5.1**	92.9 ± 2.5	100.0 ± 0.0
uWave 4 → 5	83.3 ± 0.4	35.4 ± 19.2	12.8 ± 0.4	**99.1 ± 0.7**	90.2 ± 0.7	100.0 ± 0.0
uWave 2 → 6	86.0 ± 0.8	34.5 ± 13.1	25.3 ± 16.2	**93.8 ± 1.5**	91.4 ± 1.1	100.0 ± 0.0
uWave 1 → 7	95.2 ± 1.1	26.8 ± 13.1	29.2 ± 23.6	**98.5 ± 0.4**	91.1 ± 4.1	100.0 ± 0.0
uWave 2 → 7	85.1 ± 2.2	53.9 ± 27.4	12.2 ± 0.4	91.4 ± 6.6	**98.2 ± 0.7**	100.0 ± 0.0
uWave 3 → 7	95.5 ± 0.7	64.0 ± 4.9	30.4 ± 23.4	92.0 ± 8.8	**97.6 ± 1.5**	100.0 ± 0.0
uWave 1 → 8	**100.0 ± 0.0**	78.6 ± 9.1	11.0 ± 2.9	**100.0 ± 0.0**	93.8 ± 3.2	100.0 ± 0.0
uWave 4 → 8	**100.0 ± 0.0**	44.0 ± 25.6	12.5 ± 0.0	96.7 ± 1.1	93.2 ± 5.9	100.0 ± 0.0
uWave 7 → 8	**95.2 ± 0.4**	49.7 ± 20.4	12.5 ± 0.0	93.8 ± 4.8	**95.2 ± 2.3**	100.0 ± 0.0
uWave Average	91.0 ± 6.5	48.4 ± 15.8	19.7 ± 8.3	**94.3 ± 4.6**	94.2 ± 2.9	100.0 ± 0.0

Table 1: Target classification accuracy (source → target, mean ± std%) on 10 randomly-chosen problems for each dataset, adapting between users. The results of our proposed CoDATS model and DA-WS method are underlined when statistically significantly better (paired t-test with $p < 0.05$) than both R-DANN and VRADA. The highest accuracy in each row is bold.

series. Also, on average, R-DANN accuracy on WISDM AR and uWave is lower than that of the (approximate) lower bound. Similarly, VRADA accuracy on uWave is lower than the lower bound. These results indicate that even using the CoDATS model without applying any adaptation sometimes yields an improvement over prior work for time series sensor data.

CoDATS vs. Prior Methods. We compare CoDATS with prior methods in terms of both accuracy and training-time efficiency. The CoDATS model improves in accuracy and consistency over the lower bound and prior methods. On average and on all but 4 of the 40 adaptation problems (one of which is a tie), CoDATS outperformed both of the prior works. On average and on all but 8 of the 40 adaptation problems (two of which are ties), CoDATS outperformed the lower bound. Since the prior baselines also use a domain-adversarial method for adaptation, the results indicate that the performance gains stem from our proposed model architecture. In addition to better performance, CoDATS typically exhibits greater consistency across random initializations as shown by the

Method	HAR 2⟶11 $H = 128$	HHAR 1⟶3 $H = 128$	uWave 2⟶5 $H = 315$
No Adaptation	0.016 ± 0.020	0.016 ± 0.026	0.036 ± 0.022
R-DANN	0.145 ± 0.029	0.143 ± 0.032	0.307 ± 0.057
VRADA	0.478 ± 0.050	0.452 ± 0.054	1.057 ± 0.048
CoDATS	**0.029 ± 0.032**	**0.029 ± 0.032**	**0.067 ± 0.031**
CoDATS-WS	**0.029 ± 0.039**	**0.029 ± 0.036**	0.068 ± 0.039

Table 2: Training times per iteration (seconds) on a Nvidia Tesla K80 on three of the datasets, arranged by time series length. The lowest training time in each column is bold.

lower standard deviations on average and for all but 14 of the 40 adaptation problems than both R-DANN and VRADA.

In addition to the improvements in accuracy, Table 2 indicates that CoDATS also greatly reduces training time. While the difference between CoDATS and the baseline methods depends to an extent on the number of time steps, CoDATS demonstrates faster training than R-DANN and VRADA on all datasets. We note that CoDATS requires only 20-22% of the R-DANN training time and only 6% of the VRADA training time for these datasets. Interestingly, CoDATS is less than twice the training time of the lower bound, indicating that adaptation can be performed with a large increase in accuracy while minimally impacting training time.

7.2 Multi-Source CoDATS

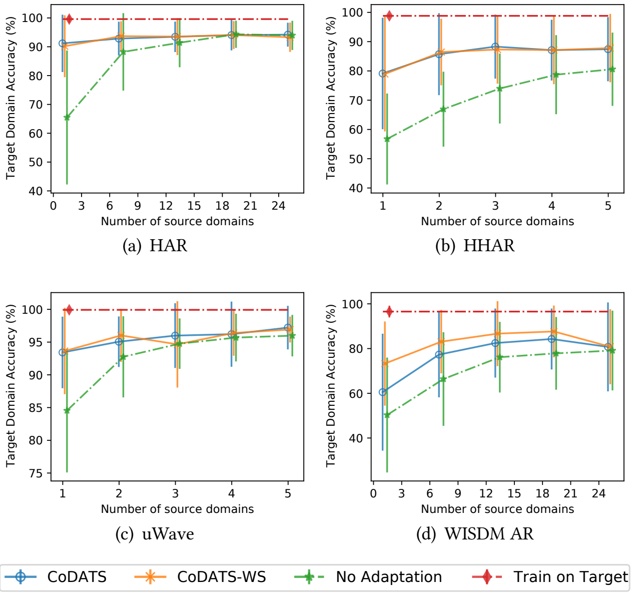
Figure 4 summarizes target domain classification performance as a function of n, the number of source domains. For each data point, we averaged over 10 random target domains for each data point in addition to 3 different random subsets of source domains.

Lower and Upper Bounds. In agreement with the single-source results, the lower bound performs poorly with one source domain. Given additional source domains, its performance improves, indicating that additional domains better cover the space of the possible ways to perform each activity/gesture and thus improve model performance. Also as in single-source adaptation, the upper bound again yields nearly 100% accuracy on all targets for all datasets.

CoDATS vs. Lower/Upper Bounds. The gap between the lower bound and CoDATS is much larger at small values of n, as the benefit of performing domain adaptation by utilizing target domain unlabeled data has diminishing returns once more labeled source domains become available. For the highest value of n on HHAR and uWave, domain adaptation still outperforms the lower bound.

The slight downturn of the CoDATS curve in HHAR and the more noticeable one in WISDM AR may stem from negative transfer. Additional domains better cover the space of possible ways of performing each activity/gesture, which yields a monotonically increasing lower bound. However, we are performing domain adaptation by learning a domain-invariant feature representation. A few people may perform activities highly different than others, meaning that if we force the representation to be invariant with respect to all of the domains, our representation may lose some ability to be discriminative. Note though that there is no downturn on the other two datasets, and the most significant drop is in the case of WISDM AR only after over 20 source domains. Thus, it is likely that in a human activity or gesture recognition settings where domain adaptation is applied, there may be fewer than 20 participants and this effect may not be noticed.

Regardless, performing multi-source adaptation on average always increases performance when compared with using only a single source – when looking from left to right, the CoDATS curves rise, always staying above the left-most point that represents single-source adaptation performance. Thus, not only does CoDATS improve over prior single-source work, but it also facilitates utilizing data from multiple sources to further improve accuracy.



(a) HAR (b) HHAR

(c) uWave (d) WISDM AR

Figure 4: Target classification accuracy when varying the number of source domains n (best viewed in color).

Number of Sources	No Adaptation	*CoDATS*	*CoDATS-WS*
$n = 1$	50.3 ± 25.7	60.5 ± 26.1	**73.3 ± 18.8**
$n = 7$	66.4 ± 20.9	77.3 ± 19.1	**83.1 ± 14.2**
$n = 13$	76.2 ± 15.8	82.5 ± 15.4	**86.7 ± 14.5**
$n = 19$	77.9 ± 16.3	84.3 ± 13.6	**87.7 ± 11.6**
$n = 25$	79.1 ± 17.8	80.7 ± 19.9	**80.8 ± 16.8**

Table 3: CoDATS-WS improvement over CoDATS for classification of WISDM. The highest accuracy in each row is bold.

7.3 Domain Adaptation with Weak Supervision

While domain adaptation with weak supervision (DA-WS) represents a different scenario than single/multi-source domain adaptation due to the availability of label proportions, we include the CoDATS-WS results in Table 1 and Figure 4 to facilitate comparing with CoDATS, which does not have label information available. A detailed view of the WISDM AR results from Figure 4 is shown in Table 3.

Single-Source Domain Adaptation. We expect that CoDATS-WS will offer limited benefit when the source and target domains are class-balanced as for the HAR, HHAR, and uWave datasets, because CoDATS-WS capitalizes on the target label distribution differing from the source. However, it should not yield a performance degradation. Additionally, it should increase in performance on datasets with high class imbalances in the target data and between domains as in the WISDM dataset (see the Appendix for plots of class balance). The largest performance difference is on WISDM, with a 5.8% increase. For the already-balanced datasets, CoDATS-WS yields a slight improvement on HAR (+1.8%), a slight degradation on HHAR (-1.5%), and nearly equal performance on uWave (-0.1%). This indicates that in situations where target-domain label proportions are available, CoDATS-WS can improve accuracy, with the same training efficiency (Table 2).

Multi-Source Domain Adaptation. While using multi-source domain adaptation yields an improvement over single-source domain adaptation, CoDATS-WS combined with multi-source domain adaptation performs even better. On WISDM AR, CoDATS-WS yields a model accuracy improvement over CoDATS for all values of n: an increase of 12.8% for n=1, 5.8% for n=7, 4.2% for n=13, 3.4% for n=19, and 0.1% for n=25, as shown in Table 3. Aligning with intuition, CoDATS-WS and CoDATS perform equally on the other three balanced datasets, as shown in Figure 4. Thus, it is evident that CoDATS-WS can yield accuracy improvements on datasets exhibiting non-uniform class distributions, which is common in real-world data.

8 CONCLUSIONS AND FUTURE WORK

In this paper, we introduced CoDATS, a model architecture that supports time series domain adaptation. From the experimental results, it is clear that this new time series model architecture improves over prior time series adaptation work in terms of both accuracy and training time efficiency. We also demonstrated that additional accuracy gains can be achieved by utilizing data from multiple source domains and by utilizing weak supervision data from known target-domain label proportions. Future work includes extending CoDATS to support heterogeneous feature sets, developing methods to handle additional forms of weak supervision relevant for time series data, and further model improvements such as incorporating insights from InceptionTime [8].

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. This research used resources from the Center for Institutional Research Computing at Washington State University.

REFERENCES

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

[2] S. Bai, J. Z. Kolter, et al. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271* (2018).

[3] S. Ben-David et al. 2010. A theory of learning from different domains. *Machine Learning* 79, 1 (2010), 151–175.

[4] K. Bousmalis et al. 2017. Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks. In *CVPR*.

[5] C. Chen, Y. Miao, C. X. Lu, and Xie. 2019. MotionTransformer: Transferring Neural Inertial Tracking between Domains. In *AAAI*. 8009–8016.

[6] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller. 2018. Transfer learning for time series classification. In *IEEE Big Data*. 1367–1376.

[7] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller. 2019. Deep learning for time series classification: a review. *DMKD* 33, 4 (2019), 917–963.

[8] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, et al. 2019. InceptionTime: Finding AlexNet for Time Series Classification. *arXiv:1909.04939* (2019).

[9] G. French et al. 2018. Self-ensembling for visual domain adaptation. In *ICLR*.

[10] J. Froehlich et al. 2007. MyExperience: A System for in Situ Tracing and Capturing of User Feedback on Mobile Phones. In *MobiSys*. 57–70.

[11] K. Ganchev, J. Gillenwater, B. Taskar, et al. 2010. Posterior regularization for structured latent variable models. *JMLR* 11, Jul (2010), 2001–2049.

[12] Y. Ganin et al. 2016. Domain-Adversarial Training of Neural Networks. *JMLR* 17, 59 (2016), 1–35.

[13] K. Greff et al. 2017. LSTM: A Search Space Odyssey. *IEEE Tr. Neur. Net. Learn. Sys.* 28, 10 (2017), 2222–2232.

[14] J. Hoffman et al. 2018. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In *ICML*, Vol. 80. 1994–2003.

[15] S. Hong, J. Oh, H. Lee, and B. Han. 2016. Learning Transferrable Knowledge for Semantic Segmentation With Deep Convolutional Neural Network. In *CVPR*.

[16] W. Hong, Z. Wang, M. Yang, and J. Yuan. 2018. Conditional Generative Adversarial Network for Structured Domain Adaptation. In *CVPR*.

[17] E. Hosseini-Asl, Y. Zhou, C. Xiong, and R. Socher. 2019. Augmented Cyclic Adversarial Learning for Low Resource Domain Adaptation. In *ICLR*.

[18] N. Hu, G. Englebienne, Z. Lou, et al. 2017. Learning to Recognize Human Activities Using Soft Labels. *IEEE Trans. Pat. Anal. Mach. Intell.* 39, 10 (2017), 1973–1984.

[19] F. Huang and A. Yates. 2012. Biased Representation Learning for Domain Adaptation. In *EMNLP-CoNLL*. 1313–1323.

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

[21] R. Jozefowicz, W. Zaremba, and I. Sutskever. 2015. An empirical exploration of recurrent network architectures. In *ICML*. 2342–2350.

[22] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann. 2019. Contrastive Adaptation Network for Unsupervised Domain Adaptation. *arXiv:1901.00976* (2019).

[23] A. Kumar et al. 2018. Co-regularized Alignment for Unsupervised Domain Adaptation. In *NeurIPS 31*. 9345–9356.

[24] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. 2011. Activity Recognition Using Cell Phone Accelerometers. *SIGKDD Explor. Newsl.* 12, 2 (2011), 74–82.

[25] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu. 2018. Adaptive Batch Normalization for practical domain adaptation. *Pattern Recognition* 80 (2018), 109 – 117.

[26] J. Liu, L. Zhong, and J. Wickramasuriya. 2009. uWave: Accelerometer-based personalized gesture recognition and its applications. In *PerCom*. 657 – 675.

[27] G. Melis, C. Dyer, and P. Blunsom. 2017. On the state of the art of evaluation in neural language models. *arXiv:1707.05589* (2017).

[28] J. Miller and M. Hardt. 2018. Stable recurrent models. *arXiv:1805.10369* (2018).

[29] B. D. Minor, J. R. Doppa, et al. 2017. Learning Activity Predictors from Sensor Data: Algorithms, Evaluation, and Applications. *IEEE TKDE* 29, 12 (2017), 2744–2757.

[30] R. Pascanu, T. Mikolov, and Y. Bengio. 2013. On the difficulty of training recurrent neural networks. In *ICML*. 1310–1318.

[31] D. Pathak, P. Krahenbuhl, and T. Darrell. 2015. Constrained Convolutional Neural Networks for Weakly Supervised Segmentation. In *ICCV*.

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

[33] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. 2018. Generate to Adapt: Aligning Domains Using Generative Adversarial Networks. In *CVPR*.

[34] R. Shu, H. Bui, H. Narui, and S. Ermon. 2018. A DIRT-T Approach to Unsupervised Domain Adaptation. In *ICLR*.

[35] A. Stisen et al. 2015. Smart Devices Are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition. In *SenSys*. 127–140.

[36] S. Sun, H. Shi, and Y. Wu. 2015. A survey of multi-source domain adaptation. *Information Fusion* 24 (2015), 84 – 92.

[37] M. Tonutti, E. Ruffaldi, A. Cattaneo, et al. 2019. Robust and subject-independent driving manoeuvre anticipation through Domain-Adversarial Recurrent Neural Networks. *Robotics and Autonomous Systems* 115 (2019), 162 – 173.

[38] V. Vercruyssen, W. Meert, and J. Davis. 2017. Transfer learning for time series anomaly detection. In *CEUR Workshop Proceedings*, Vol. 1924. 27–37.

[39] Z. Wang, W. Yan, and T. Oates. 2017. Time series classification from scratch with deep neural networks: A strong baseline. In *IJCNN*. 1578–1585.

[40] G. Wilson and D. J. Cook. 2019. A Survey of Unsupervised Deep Domain Adaptation. *arXiv:1812.02849* (2019).

[41] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig. 2017. Controllable Invariance through Adversarial Feature Learning. In *NeurIPS 30*. 585–596.

[42] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon. 2018. Adversarial Multiple Source Domain Adaptation. In *NeurIPS 31*. 8559–8570.

[43] M. Zhao et al. 2017. Learning Sleep Stages from Radio Signals: A Conditional Adversarial Architecture. In *ICML*, Vol. 70. 4100–4109.

A REPRODUCIBILITY

Here we provide additional details to aid in reproducing our results.

A.1 Experimental Setup

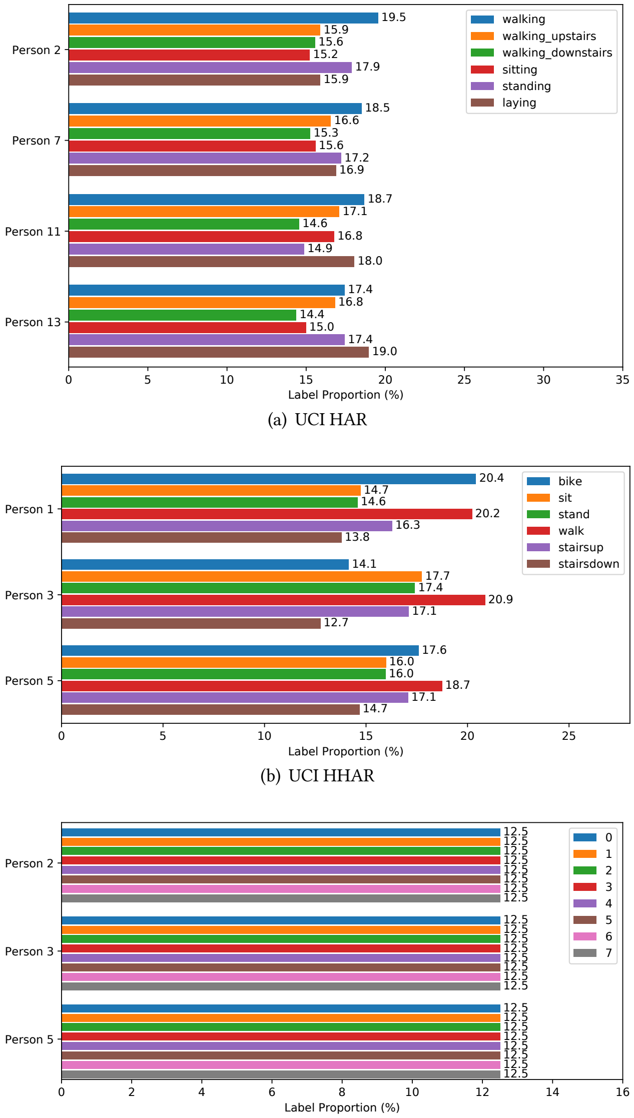
We train each model for 30,000 iterations using the Adam optimization algorithm with a learning rate of 0.0001 and a batch size of 128. We employ the DANN learning rate schedule [12] for adversarial training. Because VRADA and R-DANN have no publicly available code, we use our own implementation for each method as per the details in the original papers.

We split the data from each dataset into training, validation, and test sets. The training-test split was 80% and 20% respectively, and the training data was further split into training-validation with the same proportions. The datasets were stratified by the labels to maintain the same label proportions for training, validation, and testing sets. Each method we evaluate only has access to the training dataset: labeled data for the sources and unlabeled data for the target. The test data is only used for the final evaluation used to create our plots. All data is normalized to have zero mean and unit variance based on statistics computed from just the training set.

Each dataset consists of data from a number of participants. In the single-source experiments, we randomly select 10 of the possible adaptation problems between two domains (excluding adapting a domain to itself). For each data point, we average and compute standard deviation over three different random initializations of the network weights on the holdout test set. In the multi-source experiments, we vary the number of source domains when adapting to a separate target domain. For each data point, we average and compute standard deviation over three different random subsets of source domains on the holdout test set. Since these are trained separately, they also are trained with three different random initializations of the network weights. Then, we average over 10 different random target domains. We select more targets than source domain subsets because we expect there to be more variance among individual domains than among random subsets of multiple domains.

During training, we perform model selection by picking the model that performs best on the holdout validation set. We evaluate the model every 4,000 iterations in addition to at the end of training. The models are trained for a total of 30,000 iterations, meaning the best model is selected from a total of 9 models. The reported accuracies reflect evaluating models that performed best on this holdout validation set. Since unsupervised domain adaptation does not use any labeled target data, these results can be interpreted as an approximate upper bound on how well these methods can perform. However, labeled target data can be employed in this manner for tuning and model selection [23, 34, 40].

For single-source adaptation, DANN splits the batch size: half for the source domain and half for the target domain [12]. However, for multi-source domain adaptation, we have additional source domains. We divided the batch size evenly among all domains, thereby weighting domains uniformly. On the other hand, for domain adaptation with weak supervision (DA-WS), our method depends on estimating predicted label proportions in each batch, which requires a sufficient number of target domain predictions. For the DA-WS multi-source experiments, if we divide the batch size equally among all domains, the number of target domain predictions decreases with increasing n, yielding an extreme decrease in performance. Thus, we instead evenly split half of the batch size among the source domains and the other half for the target domain. Note another method that would fix this issue is gradient accumulation.



(a) UCI HAR

(b) UCI HHAR

(c) uWave

Figure 5: Label proportions for the participants of the relatively-balanced datasets, the first two single-source domain adaptation problems of each (best viewed in color)

A.2 Datasets

HAR [1] comes normalized and bounded to be between -1 and 1. Three-axis accelerometer, gyroscope, and estimated body acceleration were collected from 30 participants sampled at 50 Hz and come segmented into windows of 128 time steps.

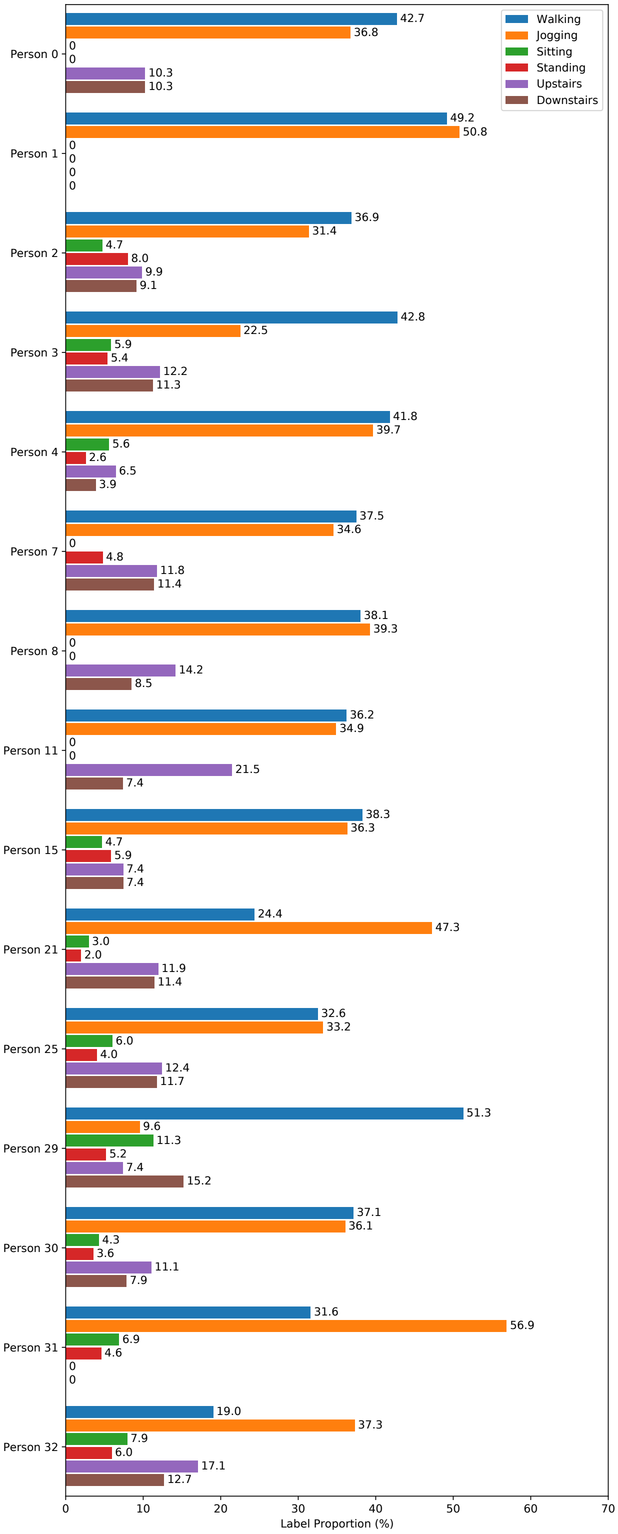


Figure 6: Label proportions for the participants of the imbalanced dataset WISDM AR, domains used for single-source domain adaptation (best viewed in color)

The authors providing HHAR only train models with either data from one sensor or the other (not both) and found models trained with the accelerometer data to have superior performance [35], so we similarly use the accelerometer data. The sensors were sampled at the highest rate each device would support, and we segment this data into non-overlapping windows of 128 time steps. We include the data collected from the 31 smartphones in our experiments.

For WISDM AR [24], we include data from the 33 participants who have enough labeled data to yield at least 30 examples for the test set (with an 80%-20% train-test split). The accelerometer is sampled at 20 Hz, and we segment this into non-overlapping windows of 128 time steps.

uWave data were collected from 8 participants over 7 days, and on each day each participant performed every gesture 10 times [26]. The data was sampled at 100 Hz. The maximum number of time steps for a gesture is 315, so we right zero-pad all gestures to this length so that all batches have a consistent size. We pad rather than segment this data because the gesture as a whole needs to be recognized, unlike in the other datasets where the activities typically consist of repetitious movements.

To show the differences in class balance, the label proportions from a few UCI HAR, UCI HHAR, and uWave participants are shown in Figure 5, which are relatively balanced between domains, and the label proportions from WISDM AR are shown in Figure 6, which have large differences between domains.