Domain Adversarial Transfer Learning for Robust Cyber-Physical Attack Detection in the Smart Grid

Yongxuan Zhang

A Thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of
Master of Computer Science at
Concordia University
Montréal, Québec, Canada

December 2020

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By:	**Yongxuan Zhang**
Entitled:	**Domain Adversarial Transfer Learning for Robust Cyber-Physical Attack Detection in the Smart Grid**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________ Chair
Dr. Jinqiu Yang

______________________ Examiner
Dr. Serguei Mokhov

______________________ Examiner
Dr. Jinqiu Yang

______________________ Thesis Supervisor
Dr. Jun Yan

Approved by ______________________
Dr. Leila Kosseim, Graduate Program Director

December 17, 2020 ______________________
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Domain Adversarial Transfer Learning for Robust Cyber-Physical Attack Detection in the Smart Grid

Yongxuan Zhang

Thanks to the increasing availability of high-quality data and the success of deep learning algorithms, machine learning (ML)-based classifiers have become increasingly appealing and investigated against sophisticated attacks in complex cyber-physical systems like the smart grid. However, many of these techniques rely on the assumption that the training and testing datasets share the same distribution and class labels in a stationary environment. As such assumption may fail to hold when the system dynamics shift and new threat variants emerge in a non-stationary environment, the capability of trained ML models to adapt in complex operating scenarios will be critical to their deployment in real-world applications. Using cyber-physical attack detection in the smart grid as the targeted application, this research aims to leverage transfer learning-based strategies to improve the robustness of ML classifiers against variations in threat types, locations, and timing in a complex dynamic CPS.

To this end, this research investigates and develops domain-adversarial transfer learning schemes for robust intrusion detection against smart grid attacks.

The main contributions include: *(i)* A domain-adversarial transfer learning scheme with customized classifiers for attack detection based on realistic smart grid data collected from a hardware-in-the-loop testbed; *(ii)* A semi-supervised transfer learning to transfer the knowledge of limited known attack incidences to detect returning threats at a later time with different system dynamics; *(iii)* A divergence-based transferability analysis and a spatiotemporal domain-adversarial transfer learning scheme for robust detection against spatial and temporal variants. Experiments were conducted on standardized IEEE benchmarks, and the results have demonstrated the promising capability of domain adversarial transfer learning to improve ML robustness against system and attack variations.

Acknowledgments

I would like to first express my sincere gratitude and deepest appreciation to my supervisor Dr. Jun Yan from Concordia Institute for Information Systems Engineering, I am grateful for all the immense support, brilliant insights, instructive comments, and kind help throughout my two years' academic research. I won't be able to finish this work without his supervision. I would also like to show my sincere gratitude to my committee members, Dr. Serguei Mokhov and Dr. Jinqiu Yang, for taking their precious time to consider my work. Many thanks for being part of the examination committee for my thesis

In addition, I would like to thank Concordia Department of Computer Science and Software Engineering for the wonderful hardware facilities as well as the courses that prepared me for this research. This work is also supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants RGPIN-2018-06724 and RGPIN-2016-06691, by the Fonds de Recherche du Quebec - Nature et Technologies (FRQNT) under grant 2019-NC-254971, and Mitcas accelerate program with Ericsson.

I would like to acknowledge all my fellow labmates: Hang Du, Juanwei Chen, William Lardier, Luyang Hou, Moshfeka Rahman, Pengyi Liao, Quentin Varo, Yuanliang Li for all your help and knowledge we shared.

Finally, I am deeply grateful to my parents for supporting me spiritually throughout my life. And special thanks go to my girlfriend Ruoxuan Guo, for her support, accompany and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we first introduce the background of the research in Section 1.1. Then we describe the problem statement in Section 1.2. Finally, the structure of this research is presented in Section 1.4.

1.1 Background

1.1.1 Cyber-Physical Systems (CPS)

Cyber-Physical Systems (CPS) are the integration of computation, networking, and physical processes. The embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa [6]. The CPS, as illustrated in Fig. 1, include computing, communication, sensors, actuators, and storage, and are connected to human-machine interfaces and multiple systems [1, 7].

The rapid development of computing technology, communication technology, and control technology has induced profound changes in human social life. With the in-depth integration and development of informatization and industrialization, traditional single-point

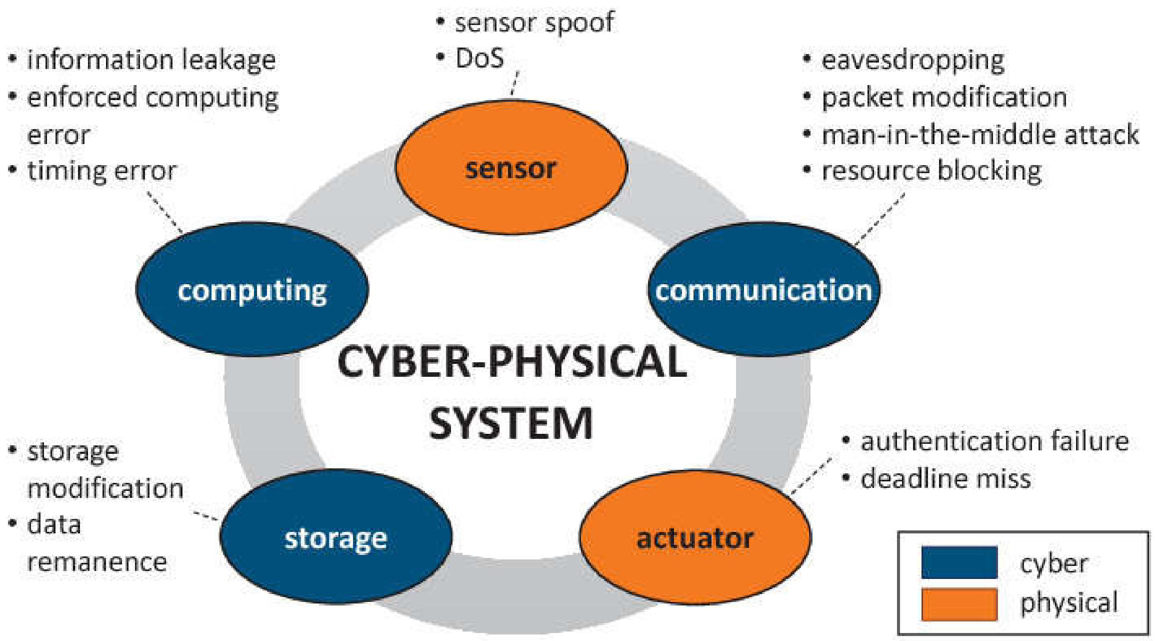


Figure 1: The cyber-physical system [1].

technology can no longer adapt to the demand for next-generation production equipment. In this context, CPS are regarded as the current frontier research direction in the field of automation. The emerging smart grid, being one of the most complex CPS ever built in history, witnesses such transformations during the ongoing integration of power and energy systems with information and communication technologies (ICTs) [8].

1.1.2 Smart Grid and Cyber Security Challenges

The smart grid has been recognized as the next-generation infrastructure that will power modern society with efficient, reliable, and sustainable electricity [9, 10, 11]. The digitization of power equipment and integration of communication networks establishes a cyber-physical infrastructure with ubiquitous automation and intelligence. Compared to traditional power grids, the smart grid will fully leverage high-speed two-way communications among utilities, regulators, and customers to better-informed and interactive electricity generation, transmission, distribution, and consumption, through new energy management capabilities such as wide-area monitoring, protection, and control (WAMPAC) [12], advanced

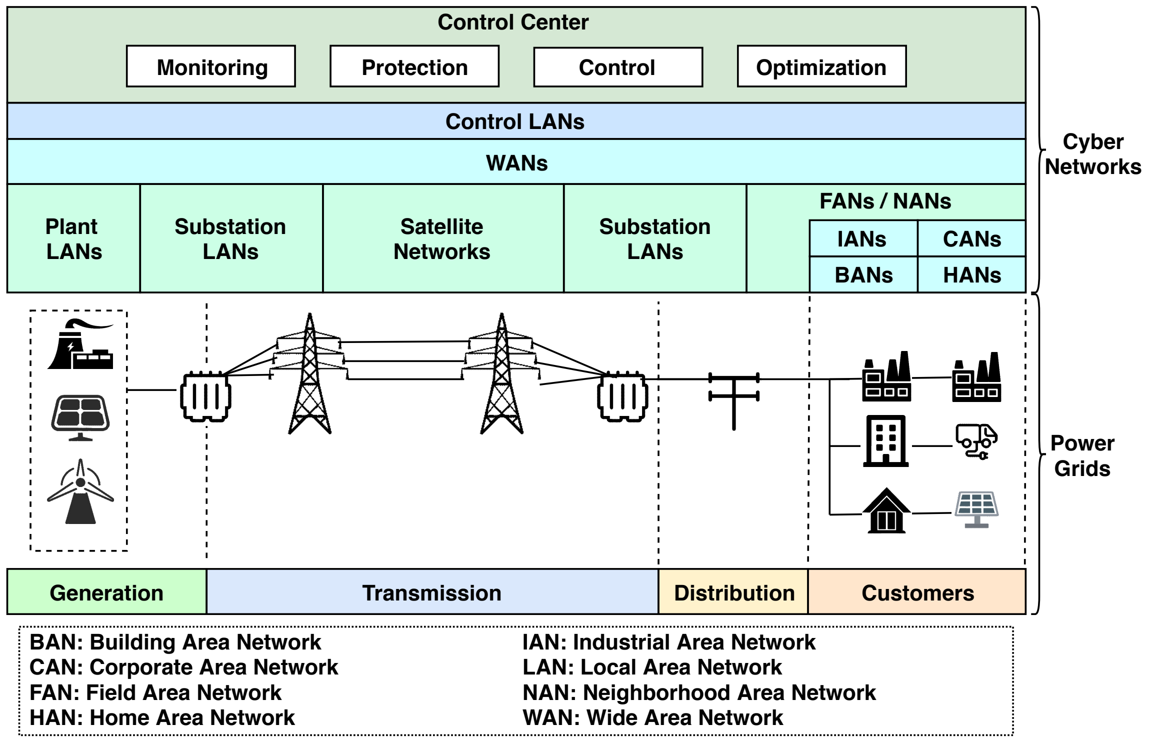


Figure 2: The cyber-physical architecture of the smart grid [2].

metering infrastructure (AMI) [13], and demand response [14], among others.

The resulting cyber-physical architecture, as illustrated in Fig. 2, will interconnect power and energy systems, intelligent field devices, substation automation systems, and control centers, among others. The smart grid communication networks will encompass:

- Wired and wireless networks of consumer-end smart meters;
- Local area networks in power plants, substations, and control centers;
- Field area networks over small regions;
- Wide area networks across regions and utilities;
- Satellite communications for time synchronization.

Millions of networked devices of different locations, manufacturers, standards, protocols, topology, and ownership will create massive complex information flow for situation analysis, decision making, and event logging.

Thanks to the cyber-physical integration, next-generation power and energy systems will empower modern society with more efficient, resilient, and sustainable electricity. However, the growing interconnection among billions of interacting systems, devices, and processes creates complex interdependence and vulnerabilities that will be inevitably exposed to cyber-attackers in the wild. The threat of a cyber-attack could be both sophisticated and disastrous, as demonstrated by recent research efforts [15, 16, 17], business studies [18], and real-world incidents [19]. From field devices to communication channels and control rooms, smart grid may expose many critical systems and processes that are both vulnerable and valuable to adversaries with terrorism, monetary, or other purposes.

The objectives and requirements of smart grid cyber security can be viewed as the following aspects: availability, integrity and confidentiality [20]. Where availability ensures timely and reliable access to the use of information. Integrity guards against improper information modification or destruction to ensure information non-repudiation and authenticity. And confidentiality protects personal privacy and proprietary information.

To meet the requirements of smart grid cyber security, proper mechanisms need to be designed and adopted, which have been commonly orchestrated in three stages: protection, detection, and mitigation [8].

1.1.3 Intrusion Detection System

Intrusion detection system (IDS) is a critical layer in the defense of smart grid security. Buczak *et al.* conducted a detailed study on data mining methods used for intrusion detection and identified three types of traditional IDS: signature-based IDS, the anomaly-based IDS and the hybrid-based IDS [21, 22].

- **Signature-based IDS:** Signature-based techniques are designed to detect known attacks by using signatures of those attacks [23, 24, 25, 26, 27]. They are effective for detecting known types of attacks without generating an overwhelming number of false alarms. They require frequent manual updates of the database with rules and signatures.

- **Anomaly-based IDS:** Anomaly-based techniques model the normal network and system behavior, and identify anomalies as deviations from normal behavior [28, 29, 30, 31, 32]. They are appealing because of their ability to detect zero-day attacks. Another advantage is that the profiles of normal activity are customized for every system, application, or network, thereby making it difficult for attackers to know which activities they can carry out undetected. Additionally, the data on which anomaly-based techniques alert (novel attacks) can be used to define the signatures for misuse detectors. The main disadvantage of anomaly-based techniques is the potential for high false alarm rates (FARs) because previously unseen (yet legitimate) system behaviors may be categorized as anomalies.

- **Hybrid IDS:** Hybrid techniques combine signature-based and anomaly detection. They are employed to raise detection rates of known intrusions and decrease the false positive (FP) rate for unknown attacks [33, 34, 35, 36, 37].

Traditional IDS need to adapt to emerging and diversifying patterns in the smart grid to effectively identify malicious attempts. However, due to the lack of attack records and challenges from system dynamics and attack variants, traditional IDS can face low detection rates in a changing environment. Such challenges call for advanced self-adaptive techniques like transfer learning. Transfer learning is designed to generalize models to adapt a new but related domain and has the potential to improve the robustness of IDS.

1.2 Problem Statement

The objective of this research is to design, implement and evaluate domain adversarial-based transfer learning intrusion detection schemes to identify unknown threats, data distribution shifts, and false data injection threats with spatial and temporal variations in the smart grid.

The highly changing system dynamics and unknown attack variants of smart grid post challenges to robust IDS. Among the existing intrusion detection methods, machine learning has been recognized as one of the most promising and reliable solutions. Nonetheless, with the assumption that training and testing data follow the same or similar data distribution, traditional machine learning models may not maintain the performance and suffer degraded detection accuracy when facing the lack of sufficient labeled data, system variations and attack variations. Transfer learning, in contrast, allows the distributions used in training and testing to be different. Transfer learning will leverage the knowledge learned from the source domain to generalize to a related target domain.

To this end, we investigated one of the state-of-the-art transfer learning frameworks, domain adversarial training of neural network (DANN) [38], and propose to modify the DANN framework according to the specific sub-problems for the smart grid IDS. Specifically, for detection threats with unseen types and locations, we leverage classic machine learning classifiers to replace the label predictor in DANN to achieve better accuracy.

However, in realistic smart grid systems, labeled normal data is sufficient but attack data is limited. This will create a class mismatching challenge for unsupervised domain adversarial transfer learning. In addition, temporal variants will introduce data distribution shifts. Aware of these challenges, we propose to formulate the intrusion detection in smart grid as a semi-supervised transfer learning problem by transferring the invariant representations between normal data of source and target domain.

To further improve the accuracy and robustness of our scheme, we propose a divergence-based transferability analysis for the smart grid to determine when to apply transfer learning. Then taken temporal and spatial information of smart grid data into consideration, we design a spatiotemporal feature extractor for the DANN framework.

1.3 Contributions

The major contributions of this research are as follows:

- We formulate a transfer learning problem for the intrusion detection in the smart grid;
- We propose a domain adversarial transfer learning-based intrusion detection scheme with customized classifiers for the problem;
- We set up different cases regarding the false data injection attack, different trends of power demand in the source domain as well as different time windows in target domain and evaluate the accuracy of the proposed semi-supervised domain-adversarial transfer learning scheme;
- We propose a divergence-based transferability analysis for CPS applications to determine when to apply transfer learning;
- We formulate a spatiotemporal multivariate time series transfer learning problem for the intrusion detection in CPS, considering the situation where attacks may happen at different time and/or locations during the power system operation, and tackle this problem by introducing a spatiotemporal domain-adversarial training scheme.

1.4 Thesis Structure

The thesis is organized into 6 chapters with an overall structure illustrated in Figure 3.

Chapter 1 presents the introduction and background of the thesis. Chapter 2 presents a literature review of intrusion detection and transfer learning for intrusion detection in the smart grid.

We proposed the transfer learning-based IDS to address unknown variants with different types and locations, temporal variations, and spatiotemporal variations within the following 3 chapters. The work of Chapter 3 and 4 reveal the potential of leveraging transfer learning to improve the robustness of IDS against threats with spatial and temporal variations, and the work of Chapter 5 considers the both. Chapter 3 presents an overview of the domain-adversarial training of neural network framework (DANN), implements and evaluates a robust detection method based on DANN with customized label predictor. The work of Chapter 3 has been published in SmartGridComm 2019 [2]. Chapter 4 presents the semi-supervised domain-adversarial training (SSDAT) for intrusion detection against false data injection in the smart grid with temporal variations. The work of Chapter 4 has been published in IJCNN 2020 [39]. Chapter 5 presents and evaluates the proposed divergence-based transferability analysis and spatiotemporal domain-adversarial training (STDAT) scheme for FDI detection. Chapter 6 summarizes the thesis and proposes future work.

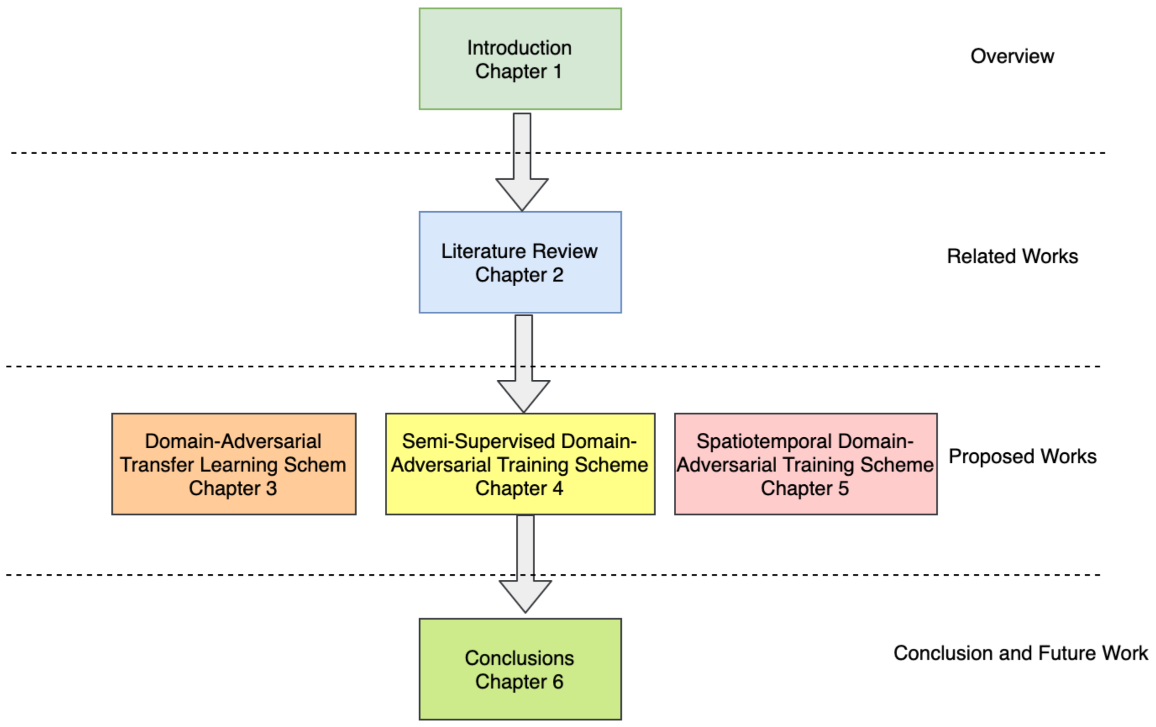


Figure 3: Overall structure of the thesis.

Chapter 2

Literature Review

2.1 Intrusion Detection in the Smart Grid

2.1.1 Non-learning-based Techniques

Generic detection methods were also developed and integrated into CPS control systems for smart gird. In this section, we will first review some existing studies that apply non-learning-based techniques for intrusion detection against attacks.

A rich line of model-based techniques capable of detecting integrity attacks on the sensors of a control system were investigated and discussed by Mo *et al.* [40]. Kyriakos *et al.* [41] proposed novel game-theoretic approaches to estimate a binary random variable based on a vector of binary sensor measurements that may have been corrupted by an attacker, also known as the Byzantine problem. Mitchell *et al.* [42] proposed a behavior-rule specification-based IDS technique for intrusion detection of physical devices. The utility of headends, distribution access points/data aggregation points and subscriber energy meters has been exemplified in their work. Lolo *et al.* [43] introduced a hybrid detection framework to detect anomalous and malicious activities by incorporating their proposed grid sensor placement algorithm with observability analysis to increase the detection rate. Simulations have shown that the network observability and detection accuracy can be improved utilizing grid-placed sensor deployment. Jokar *et al.* [44] proposed a layered specification-based IDS targeting the IEEE 802.15.4 standard. They have also introduced some known attacks against IEEE 802.15.4 and evaluated detection capabilities of our proposed IDS against them. Results have shown that their proposed IDS has the potential to successfully detect several known attacks. Faisa *et al.* [45] conducted a performance analysis experiment of some existing state-of-the-art data stream mining algorithms on a public IDS dataset. A run-time semantic analysis has also been developed to provide early warnings on altered control commands in the SCADA system by Lin *et al.* [46]With an efficient look-ahead power flow analysis, the semantic analysis simulates the execution consequence of control packets to issue alerts if the execution would result in unfavorable impacts such as line outages. A model-based IDS has been developed against the input attacks on the AGC system by Sridhar *et al.* [47]. The IDS utilizes RT load forecast to predict the ACEs over time, and their performances are compared with that of the actual ACEs obtained. With statistical and temporal characterizations of these performances, attack detector in the IDS is able to detect scaled and ramped inputs before they are sent into the AGC system.

While traditional non-learning-based IDS relies on system specifications[48] and attacks signatures to identify the adversaries, recent efforts [49] have developed advanced IDS based on machine learning and deep learning techniques to extract informative features and tackle system complexities in the detection and classification of cyber-physical attacks.

2.1.2 Learning-based Techniques

Machine learning (ML) can be categorized into mainly three types: unsupervised, semi-supervised, and supervised. For unsupervised learning, the assumption is that labeled data is missing and the task is to find out the hidden patterns from unlabeled data through learning. When partially labeled data is given during the training stage, the problem is transformed into semi-supervised learning. The addition of the labeled data greatly helps to solve the problem. When the training data are all labeled, the problem is called supervised learning and generally the task is to utilize a labeled dataset to build a model via the training process and evaluate its performance on another unlabeled set during the testing process.

Following the increasing availability of data and computing power, ML has been extensively investigated in contemporary literature as an intrusion detection and classification technique in communication/network security [49] as well as smart grid security [22, 50, 51]. This trend comes from the fact that cyber security has become more sophisticated and complex than before, and traditional approaches are no longer effective [52], and machine learning has non-linear analysis capabilities to detect stealthy cyber-attacks in complex systems [51].

Classic ML approaches have been applied in IDS. Le *et al.* [53] proposed a method to incorporate the FAIR's LEF into Bayesian Network (BN) to derive the numerical assessments to rank the threat severity. A Bayesian network [54, 55] is a model that encodes probabilistic relationships among important variables. This technique is generally used for intrusion detection in combination with statistical schemes. Jiang *et al.* [56] leveraged Hidden Markov Model (HMM) of real-time frequency and voltage variation features as well as other approaches to build a fault detection, identification, and location approach. Artificial neural network (ANN), designed to simulate the way the human brain analyzes and processes information, has demonstrated the powerful ability to reproduce and model nonlinear processes. ANN has found applications in many disciplines including IDS in smart grid [57, 58, 59].

To detect FDIA, general machine learning techniques such as Perceptron [60], k-Nearest Neighbor (kNN) [61] and Support Vector Machines (SVM) [62] were also investigated by

Ozay *et al.* [63]. The results showed that machine learning algorithms can detect attacks with higher performance than traditional non-learning methods. Manandhar *et al.* [64] adopted the Kalman filter to estimate the variables of a wide range of state processes in the model. In 2014, Oak Ridge National Laboratory has conducted a comprehensive evaluation of several classic machine learning approaches in binary classification tasks [65], which established a benchmark for different machine learning approaches on power system disturbance and cyber-attack discrimination. Subsequent studies based on neural networks and ensemble learning have reached over 95% overall accuracy [66, 67].

However, these existing approaches took the default assumption that training and testing datasets are from the same distribution and the same attacks have appeared in both datasets. While traditional machine learning can suffer degrading accuracy when the assumption fails to hold. In the context of smart grid, labeled attack data is rare and system variations will introduce different data distribution. The trained machine learning model may not be able to adapt such a situation.

2.2 Transfer Learning

2.2.1 Overview of Transfer Learning

Most of the machine learning methods assume that the distributions of the labeled training and unlabeled testing data follow the same distribution. Transfer learning (TL), in contrast, allows the domains, tasks, and distributions used in training and testing to be different.

TL aims at applying knowledge or patterns learned in a certain domain (or task) to a different but related domain (or task) [68]. The “transfer" is similar to the process where a graduate tries to apply what he/she learns in class to a practical problem, and the key is to find a mapping between the problems so that the learned knowledge can be adapted to the practical situation and resolve the problem. TL approaches can be grouped into three categories as illustrated in Fig. 4: inductive TL, transductive TL, and unsupervised TL. Now TL has been successfully applied in image recognition, document classification, sentiment classification, and language processing.

- **Inductive TL:** The target task is different from the source task, no matter when the source and target domains are the same or not [69, 70, 71, 72, 73]. In this case, some labeled data in the target domain are required to induce an objective predictive model function for use in the target domain.
- **Transductive TL:** The source and target tasks are the same, while the source and target domains are different [74, 75, 76, 77, 78]. In this situation, no labeled data in the target domain is available while labeled data in the source domain is available. In addition, according to different situations between the source and target domains, we can further categorize the transductive TL setting into two cases, either the feature space is different or the marginal probability distributions are different.
- **Unsupervised TL:** Similar to inductive TL setting, the target task is different from but related to the source task [79, 80, 81, 82]. However, the unsupervised TL focus on solving unsupervised learning tasks in the target domain, such as clustering, dimensionality reduction and density estimation. In this case, there are no labeled data available in both source and target domains in training.

2.2.2 Domain Adaptation in Transfer Learning

Domain adaptation is a transductive TL technique, where the marginal probability distributions of input data from the source and target domains are different while the feature spaces are the same [68]. Existing domain adaptation approaches can be categorized into unsupervised [83, 84, 38], semi-supervised [85, 86, 87], and supervised domain adaptation [88, 89, 90].

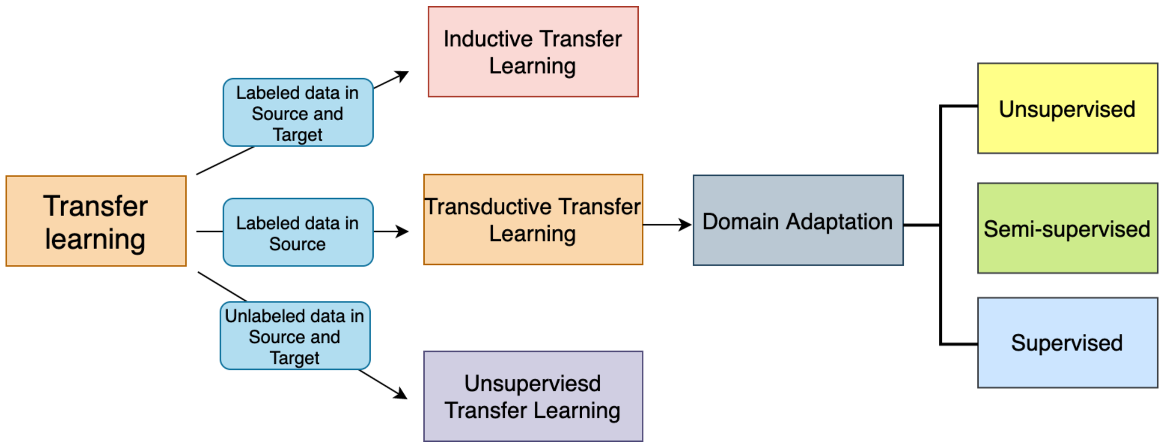

Transfer learning
Labeled data in Source and Target
Labeled data in Source
Unlabeled data in Source and Target
Inductive Transfer Learning
Transductive Transfer Learning
Unsuperviesd Transfer Learning
Domain Adaptation
Unsupervised
Semi-supervised
Supervised

Figure 4: An overview of transfer learning.

- **Unsupervised domain adaptation (UDA):** The UDA leverages labeled source domain data and unlabeled target domain data to decrease domain discrepancy or find domain invariant representations. Thus models trained on the source domain can generalize well on the target domain. Fang *et al.* [83] proposed to reduce distribution discrepancy and increases inter-class margins via Sphere Retracting Transformation. Ganin *et al.* [38] introduced adversarial training into domain adaptation to find domain invariant features. A domain classifier is utilized to distinguish data from two domains. A feature extractor is trained to confuse the domain classifier by reversing the gradient from the domain classifier. Li *et al.* [84] proposed category transfer to improve the adversarial domain adaptation. It iteratively estimates and minimizes Wasserstein distance between categories in multi-category structures to avoid negative transfer between different category data from source and target.

- **Semi-supervised domain adaptation (SSDA)**: SSDA assumes that limited labeled target data is available and can be used to support domain adaptation during the training stage. Wang *et al.* [87] proposed a transfer Fredholm multiple kernel learning approach. Fredholm integrals from two domains are calculated by labeled data to learn a kernel predictive model across two domains. Pereira *et al.* [86] made use of labeled data from two domains to minimize squared induced distance between instances from different classes and different domains and maximize squared induced distance between instances from different classes and different domains. Similarly Li *et al.* [85] considered to highlight the discrimination information of the labeled samples, which takes into account the class connections between source samples and target samples.

- **Supervised Domain Adaptation (SDA):** For SDA, the assumption is that only very few target labeled samples per class are available and the goal is to learn a prediction function from the source domain and generalize well on the target domain. Motiian *et al.* [90] exploited the Siamese architecture to learn an embedding subspace that is discriminative, and where mapped visual domains are semantically aligned and yet maximally separated. And under the supervised setting, they found that reverting to point-wise surrogates of distribution distances and similarities provides an effective solution. Garcia *et al.* [88] presented a comprehensive study on supervised domain adaptation of PLDA based i-vector speaker recognition systems, including fully Bayesian adaptation, approximate MAP adaptation, weighted likelihood and SPLDA parameter interpolation.

Most of the existing domain invariant adaptation approaches focus on aligning distributions through minimizing the divergence between domains or employ adversarial training. Sun *et al.* [91] used Correlation alignment (CORAL) to learn a nonlinear transformation that aligns correlations of layer activations in deep neural networks. Long *et al.* [92] proposed to minimize the joint maximum mean discrepancy (JMMD) criterion through a transfer network by aligning the distributions of multiple domain-specific layers across domains.

Domain-adversarial training is one of the latest domain adaptation techniques inspired by the highly-successful generative adversarial networks (GANs) [93], which trains a generative network and a discriminator network concurrently in a competitive/adversarial way.

The generative network tries to “fool" the discriminator network by learning to find a new feature set not distinguishable by the latter. This adversarial process allows the learning of a feature set that better captures the patterns in the data distribution for various tasks. Inspired by the highly-successful generative adversarial networks (GANs) [93], Tzeng *et al.* considered a similar setup for domain adaptation in [94], which improves the results by incorporating some labeled target examples and matching label distributions. Meanwhile, Ganin *et al.* proposed the domain-adversarial training based on neural networks (DANN) [38] as a UDA technique for image applications. DANN is one of the most studied UDA frameworks and has been recognized as a promising solution in computer vision.

Although an extensive literature of UDA has been applied to computer vision, only a few studies have considered the time series data in the content of domain adaptation. Purushotham *et al.* [95] proposed variational recurrent adversarial deep domain adaptation (VRADA) and recurrent domain adversarial neural network (R-DANN) with RNNs as feature extractors and tested on time series medical data. Wilson *et al.* [96] propose a novel Convolutional deep Domain Adaptation model for Time Series data. Existing works reveal the potential of incorporating time series analysis into domain adaptation.

2.3 Transfer Learning for Learning-Based Intrusion Detection

Researchers have recently started to introduce TL in cyber-security. Bartos *et al.* [97] computed a self-similarity measure of the network traffic logs for the domain adaptation problems with conditional shift in network security. Juan *et al.* proposed a feature-based TL framework that was able to boost classifier performance on the well-known NSL-KDD dataset of TCP traffic [98], demonstrating the potential merit for cyber-security analysis. D. Nahmias *et al.* [99] applied feature TL from pre-trained VGG19 neural network mode on malware detection.

For TL in IDS, different approaches were also investigated. Mathew *et al.* [100] leveraged a pretrained deep convolutional neural network and transfer it to the target domain for IDS on ATM surveillance dataset. Tariq *et al.* [101] proposed CANTransfer, an intrusion detection method using Transfer Learning for Controller Area Network bus, where a Convolutional LSTM based model is trained using known intrusion to detect new attacks. Singla *et al.* [102] investigated the viability of TL, compared the detection accuracy of a network IDS model trained using TL with a network IDS model trained from scratch, and showed that TL enables detection models to perform much better at identifying new attacks when there is relatively less training data available. Motivated by the existing efforts, we investigate several directions to further improve the detection accuracy of learning-based IDS.

Chapter 3

Domain-Adversarial Transfer Learning Scheme

3.1 Background

As mentioned in Section 2.2, TL has been widely adopted in various image/video applications [103, 104] as a promising solution to transfer a learned model into new but related domains or tasks. Given a source domain with labeled data and a target domain with new unlabeled data, TL methods are tasked to learn the mappings of both domains. By learning a new feature space where inner-class similarity and inter-class distance are preserved, TL methods allow machine learning-based classifiers to retain their performance on incoming data with new distributions. However, such capacity has not been extended or validated in complex cyber-physical systems like the smart grid to improve the robustness of IDS, where the common practice is still to manually select and test different feature subsets. In addition, as discussed in Section 1.2, traditional learning-based IDS assume that training and testing data follow the same or similar data distribution and they may not maintain the performance and suffer degraded detection accuracy when facing the lack of sufficient labeled data, attack variations and system variations.

Motivated by these gaps, this chapter proposes a domain-adversarial framework based on the latest deep adversarial learning technique [38] to extract novel features and transfer different machine learning classifiers for new operation scenarios and attack threats. A realistic smart grid security dataset, collected over hardware-in-the-loop simulators [3] with multiple attack and normal scenarios as well as cyber and physical system information, has been chosen to evaluate the performance of the proposed scheme. The dataset allows the study to leverage deep packet inspection (DPI) of the payloads (measurements), system logs, and snort alerts, instead of packet headers or traffic statistics, to utilize the cyber-physical information against unknown threats. The dataset assumes that the data is unencrypted. Similar DPI methods have also been adopted in [105], [106] to extract voltages and currents for detecting false data injection and fault localization. To validate the effectiveness of DANN on improving the robustness of IDS, we set up experiments on detecting unknown types of attack threats and same attacks with different locations. Ablation analysis has been performed to provide insights on the choice and influence of critical hyper-parameters. The results have shown that the domain-adversarial TL scheme can significantly improve classification accuracy against unknown threats of different types and locations. Discussions on hyper-parameters are also provided for practical uses.

3.2 Problem Formulation

In TL, a domain $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution *P(X)*. Given a specific domain $\mathcal{D}$= {$\mathcal{X}$,*P(X)*}, a task consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$ to be learned from the training data.

The source domain and target domain denote different data extracted from the control center. Labeled source domain data is used to train the classifiers. We collect unlabeled data in the target domain for domain adaptation so that the TL model can map the data of two domains into the same new feature space and decrease the distribution distance between projected source data and target data.

We model the attack detection of smart grid as a binary classification problem, that is, to classify the events as an attack or a natural event. The source domain $\mathcal{D}_S$ and target domain $\mathcal{D}_T$ have the same feature space $\mathcal{X}$ but different combination of attack types and normal events. The marginal probability distribution *P(X)* is thus different. We denote the labeled source domain as $\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), ..., (x_{S_{n_S}}, y_{S_{n_S}})\}$ and the unlabeled target-domain as $\mathcal{D}_T = \{(x_{T_1}, y_{T_1}), ..., (x_{S_{n_T}}, y_{S_{n_T}})\}$. The goal is to learn a predictive function $f(\cdot)$ from the training data that predicts labels in the target domain.

The following problems will be considered in this chapter:

- One threat appears only in $\mathcal{D}_S$ and another new threat appears only in $\mathcal{D}_T$;
- The same type of threats appears in both $\mathcal{D}_S$ and $\mathcal{D}_T$ but each domain contains attacks at different locations.

3.3 Methodology

3.3.1 Domain-Adversarial Training

The DANN architecture consists of three neural networks: a feature extractor, a domain classifier, and a label predictor. Note that each network can be implemented as a single or multi-layer perceptron, where the simplest structure was used in this chapter, as by Ganin *et al.* in [38].

The feature extractor learns a function $G_f : X \to R^D$ that maps an m-dimensional input into a D-dimensional feature:

$$G_f(\mathbf{x}) = sigmoid(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

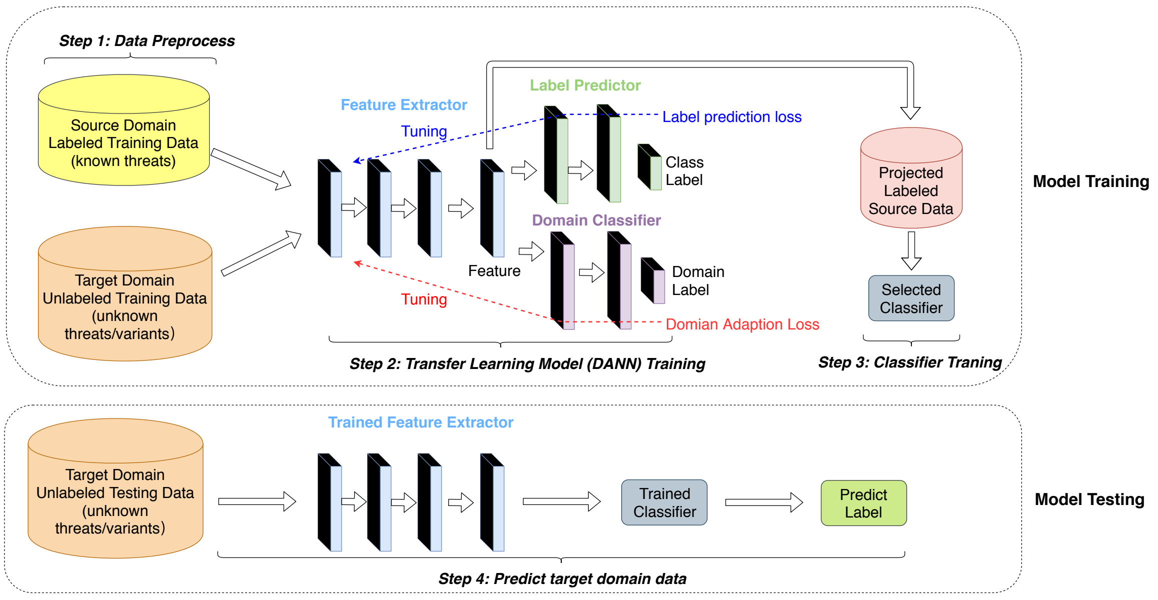

Step 1: Data Preprocess
Source Domain Labeled Training Data (known threats)
Target Domain Unlabeled Training Data (unknown threats/variants)
Feature Extractor
Tuning
Label Predictor
Label prediction loss
Class Label
Domain Classifier
Feature
Domain Label
Tuning
Domian Adaption Loss
Projected Labeled Source Data
Selected Classifier
Step 2: Transfer Learning Model (DANN) Training
Step 3: Classifier Traning
Model Training
Trained Feature Extractor
Target Domain Unlabeled Testing Data (unknown threats/variants)
Trained Classifier
Predict Label
Step 4: Predict target domain data
Model Testing

Figure 5: Domain-adversarial transfer learning for IDS in smart grid.

where $sigmoid(\mathbf{a}) = [\frac{1}{1+e^{-a_i}}]_{i=1}^{|\mathbf{a}|}$, $(\mathbf{W}, \mathbf{b}) \in R^{D\times m} \times R^D$ is a matrix-vector pair of the weights $\mathbf{W}$ and the bias $\mathbf{b}$ of the feature extractor. As with other deep learning techniques, there is no direct loss function associated with the feature extractor; the losses of the label predictor and domain classifier will be back-propagated to adjust the weights of the feature extractor.

Similarly, the label predictor aims to learn a function G_y from the extracted features to the class labels:

$$G_y(\mathbf{x}) = softmax(\mathbf{V}G_f(\mathbf{x}) + \mathbf{c}) \tag{2}$$

where $softmax(\mathbf{a}) = [\frac{e^{a_i}}{\sum_{j=1}^{|a|} e^{a_j}}]_{i=1}^{|\mathbf{a}|}$, $(\mathbf{V}, \mathbf{c}) \in R^{L\times D} \times R^L$ is a matrix-vector pair of the weights $\mathbf{V}$ and the bias $\mathbf{c}$ of the label predictor.

Given a sample-label pair $(\mathbf{x}, y)$, the loss function of the binary classification problem is given as:

$$L_y(\mathbf{x}, y) = -y\log[G_y(\mathbf{x})] - (1-y)\log[1 - G_y(\mathbf{x})] \tag{3}$$

For the source domain, the domain-adversarial training can thus be formulated as the following optimization problem:

$$\underset{\mathbf{W},\mathbf{b},\mathbf{V},\mathbf{c}}{min}\left[\frac{1}{n_S}\sum_{\mathbf{x}\in\mathcal{D}_S} L_y(\mathbf{x},y)+\lambda\cdot R(\mathbf{W},\mathbf{b})\right] \tag{4}$$

where $R(\mathbf{W},\mathbf{b})$ is a regularizer weighted by a hyper-parameter λ. The regularizer determines the penalty for the feature extractor, which provides the mapping from the source/ target domains to a new feature space where the domain adaption loss, or the dissimilarity of samples mapped from the source and target domains, are minimized.

The domain classifier learns a logistic regression $G_D \rightarrow [0,1]$ that predicts whether an input $\mathbf{x}$ is from $\mathcal{D}_S$ or $\mathcal{D}_T$:

$$G_d(\mathbf{x}) = sigmoid(\mathbf{u}^T G_f(\mathbf{x}) + z) \tag{5}$$

where $(\mathbf{u}, z) \in R^D \times R$ is a vector-scalar pair of the weights $\mathbf{u}$ and the bias z of the domain classifier.

Given a sample $\mathbf{x}$, the domain adaption loss L_d is given as:

$$L_d(\mathbf{x}, d_\mathbf{x}) = -d_\mathbf{x}\log[G_d(\mathbf{x})] - (1-d_\mathbf{x})\log[1-G_d(\mathbf{x})] \tag{6}$$

where $d_\mathbf{x} = 0$ if $x \in \mathcal{D}_S$ and $d_\mathbf{x} = 1$ if $x \in \mathcal{D}_T$.

Subsequently, the regularizer $R(\mathbf{W},\mathbf{b})$ can be defined as:

$$R(\mathbf{W},\mathbf{b}) = -\frac{1}{n_S}\sum_{\mathbf{x}\in\mathcal{D}_S} L_d(\mathbf{x},d_\mathbf{x}) - \frac{1}{n_T}\sum_{\mathbf{x}\in\mathcal{D}_T} L_d(\mathbf{x},d_\mathbf{x}) \tag{7}$$

and the optimization problem in (4) can be rewritten as:

$$\begin{aligned}\underset{\mathbf{W},\mathbf{V},\mathbf{b},\mathbf{c},\mathbf{u},z}{min} \Bigg[\frac{1}{n_S} \sum_{\mathbf{x}\in\mathcal{D}_S} L_y(\mathbf{x}, y) - \frac{\lambda}{n_S} \sum_{\mathbf{x}\in\mathcal{D}_S} L_d(\mathbf{x}, d_{\mathbf{x}}) \\ - \frac{\lambda}{n_T} \sum_{\mathbf{x}\in\mathcal{D}_T} L_d(\mathbf{x}, d_{\mathbf{x}}) \Bigg]\end{aligned} \tag{8}$$

3.3.2 DANN with Customized Label Predictor

The overall architecture, as illustrated in Fig. 5, consists of four steps: (1) collect labeled source data and unlabeled target data; (2) train the model over both domains to obtain a new feature space in a domain-adversarial manner; (3) use the new feature representations of the labeled data in the source domain to train classifiers; (4) map the new data to the new feature space and predict their class labels.

The original DANN employed simple perceptron networks and a softmax layer as the label predictor/classifier. The design can be easily extended to integrate different alternative classifiers which may perform better in different domain-specific tasks [49]. For example, the k-Nearest Neighbour classifier is more suitable for data with cluster characteristics, low dimension, and no need for parametric assumptions. The decision tree is more advanced when the task requires intuitive knowledge expression and simple implementation.

Aware of this potential, we adopt a two-phase domain-adversarial training. The first phase consists of Steps 1 and 2 in Fig. 5, in which a rough training process is performed with the original DANN so that the errors can be back-propagated to the feature extractor based on both label prediction and domain adaption losses. The second phase consists of Step 3 in Fig. 5, in which the label predictor is replaced with an alternative ML-based binary classifier. The new classifier can be trained using the extracted features to exploit the knowledge transferred between the source and target domains. The final class label (normal vs. attack) will be given by the alternative classifier in Step 4 using the extracted features.

3.4 Experiments and Results

3.4.1 Attack Model

We choose the standardized smart grid Industrial Control System (ICS) Cyber Attack Datasets [3] (more details therein) for training and testing. The data is collected by the University of Alabama in Huntsville and Oak Ridge National Laboratory on a hardware-in-the-loop testbed. Illustrated in Fig. 6, the testbed implemented a high-level wide-area power grid abstracted into 3 buses, 2 generators (G1 and G2), and 4 PMU-embedded intelligent relays (R1-R4) connected to four circuit breakers (BR1-BR4). Measurements of the relays are sent through a simulated wide-area network to the phasor data concentrator (PDC) and then forwarded through a security gateway to the control room. Attack scenarios were built and simulated with the assumption that an actor had already gained access to the substation network and poses an insider threat by issuing commands from the substation switch.

Remote tripping command injection (RTCI): RTCI is an attack that sends a command to a relay which causes a breaker to open. It can only be done once an attacker has penetrated outside defenses. Attackers remotely send unexpected relay trip commands to closely mimic the line maintenance scenarios. The malicious trip command originates from another node on the communications network with a spoofed legitimate IP address. Such malicious attacks would cause breakers to unpredictable open, and attackers could appear as penetrating outsides defenses. The remote tripping command injection is injected into each single relay, or two relays on the same line (R1 and R2, or R3 and R4 respectively).

Data Injection (DI): For the DI attack, an attacker aims to manipulate the sensor measurements to induce an arbitrary change in the estimated value of state variables without being detected by the bad measurement detection algorithm of the state estimator. Hence, this attack could blind the operator and causes a blackout by changing values to parameters such as current, voltage, and frequency, among others. Here in the ICS dataset, a valid fault is imitated by changing values to parameters such as current, voltage, sequence components etc., in order to blind the operator and cause a blackout.

RTCI and DI attack are selected as the attack models in our experiments from this ICS dataset for two reasons: 1) The two attack model are both injection attack, which meet the formulation of source domain and target domain in TL; 2) Remote tripping command injection contains attacks with different locations and could help to investigate the case of detecting attacks at a different location.

3.4.2 Experiments Setup

The 128 features of this dataset are explained as follows. There are 29 types of measurements from each phasor measurement units (PMU). A PMU or synchrophasor is a device which measures the electrical waves on an electricity grid, using a common time source for synchronization. In the system, there are 4 PMUs which measure 29 features for 116 PMU measurement columns total. There are 12 features for control panel logs, snort alerts and relay logs of the 4 PMU.

As shown in Table 9, this chapter considers a binary classification problem between

Table 1: Summary of Classes and Scenarios Used in the Study

Classes	Scenarios	Descriptions
Normal	No Events	Normal operation with load variation
Attack	Data Injection (DI)	Attacker manipulates current, voltage, etc. to mislead controllers and/or operators into mal-operations.
	Remote Tripping Command Injection (RTCI)	Attacker sends a command to a relay and open a circuit breaker, directly causing a line outage.

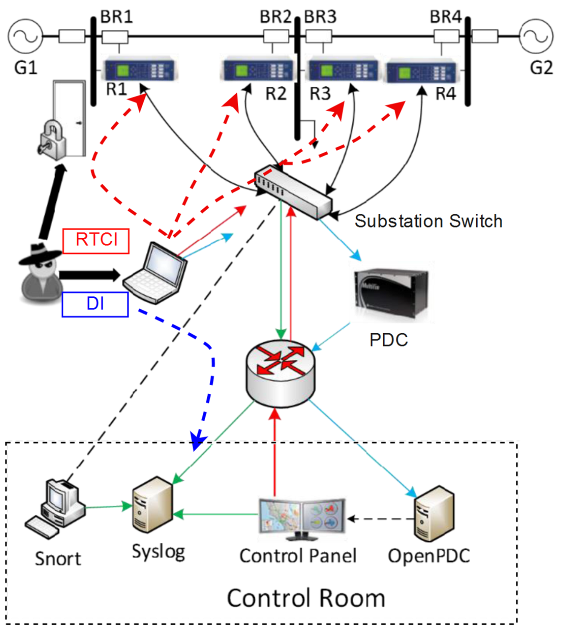


Figure 6: The data collection testbed and the attack threats [3].

normal and attack events; the attack class further consists of two threat scenarios, i.e., DI and RTCI. Each sample contains 128 features: 116 from four PMUs (29 each) and 12 from relay/control panel logs and the snort alert. We select two main attack categories DI, RTCI combined with Normal Events (Natural events and No events) as our domains.

Table 2 shows the cases of threats in the source domain (known/labeled) and threats in the target domain(unknown/unlabeled). The normal-class is present in all cases and accounts for 50% of both training and testing data, respectively. Classifiers in Cases 1 and 2 are trained on false data injection and faced by unseen remote tripping commands. The

Table 2: Case Setup for Domain Adaption.

Cases	**Threat in Source Domain**	**Threat(s) in Target Domain**
1	DI	RTCI
2	DI	DI and RTCI
3	RTCI	DI
4	RTCI	DI and RTCI
5	RTCI-15 (Relay R1)	RTCI-16 (Relay R2)
6	RTCI-17 (Relay R3)	RTCI-18 (Relay R4)

Table 3: Hyperparameter Setting

Classifiers	Hyperparameter
AdB	base learner = DecisionTree; learning rate = 1; algorithm = 'SAMME'.
kNN	N neighbors = 9
SVM	kernel = 'rbf'; C = 1.
RF	number of estimators = 100
CART	max depth = 5
ANN	architecture of hidden layers = 50-25-10

unlabeled target domain in Case 1 contains only RTCI threat while in Case 2 a mixture of both the known DI and the unknown RTCI threat; the combination of both cases will allow more comprehensive evaluation of the performance. Similarly, classifiers in Cases 3 and 4 are trained on RTCI threats are faced by unseen DI threats. Classifiers in Cases 5 and 6 are trained on RTCI threat on one relay and faced by unseen threats launched on the other one on the same line. The extractor will extract 60 features from the original 128; for other parameters, the domain adaptation regularizer is set as 0.5, the learning rate as 0.1, and the number of epochs as 1,000.

Similar to [65, 98], we choose AdaBoost (AdB), k-Nearest Neighbor (kNN) [61], Support Vector Machine (SVM) [107], Random Forest (RF) [108], Classification and Regression Tree (CART) [109], and Artificial Neural Network (ANN) [110] as the baseline classifiers and implement them with the Python Scikit-learn library [111]. The hyperparameters are shown in Table 3. The experiments are running on Windows 10 64 bit operating system with Intel Core i7-8700 CPU, 16GB ram and NVIDIA Quadro P620 GPU.

Table 4: Comparison of original and domain-adversarial machine learning classifiers against unknown threat variants

Cases	Methods	AdB	kNN	SVM	RF	CART	ANN
1	Original	72.0%	77.6%	57.9%	51.4%	53.2%	83.0%
	Domain-Adversarial	86.8%	86.0%	82.9%	88.2%	76.3%	84.5%
	Improvement	**+14.8%**	**+8.5%**	**+25.0%**	**+36.8%**	**+23.1%**	**+1.5%**
2	Original	77.3%	82.7%	71.0%	84.5%	76.7%	86.5%
	Domain-Adversarial	94.2%	90.4%	85.5%	95.2%	79.2%	87.8%
	Improvement	**+16.9%**	**+7.8%**	**+14.5%**	**+10.7%**	**+2.5%**	**+1.3%**
3	Original	73.0%	75.1%	64.8%	76.0%	61.3%	81.7%
	Domain-Adversarial	83.6%	82.2%	80.9%	84.5%	75.7%	83.6%
	Improvement	**+10.6%**	**+7.1%**	**+16.1%**	**+8.5%**	**+14.4%**	**+1.9%**
4	Original	71.2%	80.5%	66.7%	83.0%	69.5%	85.9%
	Domain-Adversarial	89.3%	88.2%	85.3%	90.0%	79.6%	87.6%
	Improvement	**+18.1%**	**+7.7%**	**+18.6%**	**+7.0%**	**+10.1%**	**+1.6%**

3.4.3 Detection of Unknown Threats Types

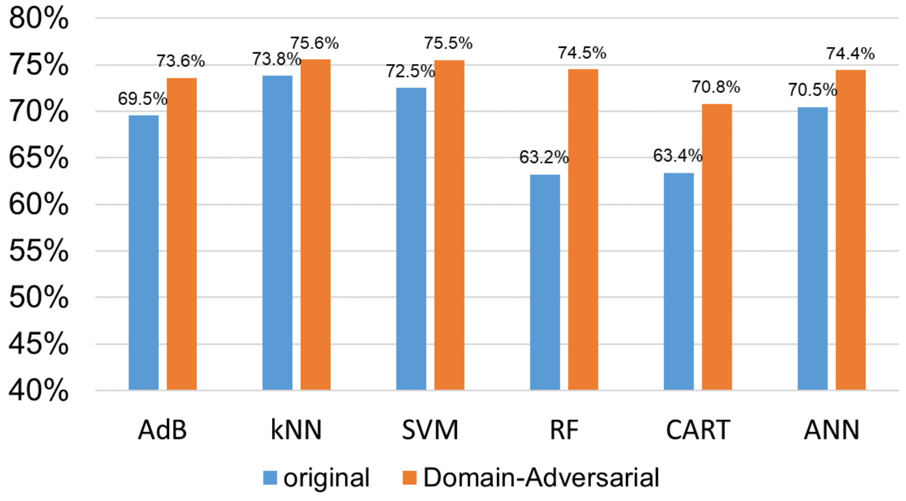
To evaluate the performance of detecting an unseen attack in the power system, we recreate the datasets from original ICS datasets based on the assumption that a new attack type exists in the target domain. Each dataset contains 3,000 samples. To make the dataset balanced, attack class and normal class account for 50% separately. The two attack classes are also set to 50% in combined attack class DI+RTCI. We first evaluate the case where the normal and attack classes are balanced, each accounting for 50% of the training and the testing data, respectively.

Table 4 shows the accuracy with the original and the domain-adversarial trained classifiers, along with the performance improvement, for Cases 1 to 4 with unknown threat types. It can be seen that while different baseline classifiers demonstrated accuracy among these cases, all have benefited from the domain-adversarial training to various extents. The significant improvements were observed on AdB, kNN, SVM, and RF classifiers, with improvements from 7% up to 36.8%; even for the less significant ones, the CART and the

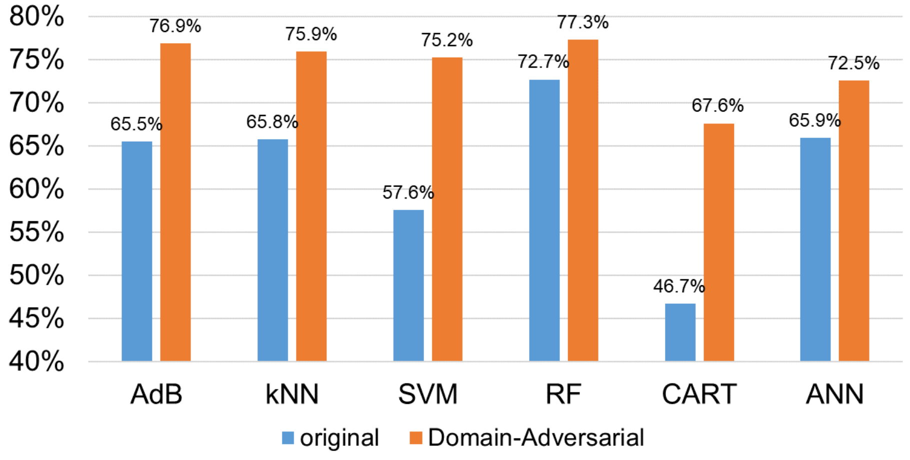
ANN, the average improvement is over 2.5%. Comparing between Cases 1 and 2, as well as between Cases 3 and 4, it can be seen that although all baseline classifiers demonstrated boosted accuracy (and less improvement) in the presence of known threats in the unlabeled target domain, they are still unable to handle unknown threats alone and would still benefit from the domain adaptability from domain-adversarial training. In practical smart grid communications where the threats are expected to evolve/revolve frequently, such capacity can allow a trained/deployed model to remain robust/resilient against the unknown threats for better situational awareness.

3.4.4 Detection of Unseen Threats Locations

Fig. 7a and Fig. 7b show the accuracy of Cases 5 and 6 where the same type of attacks are launched at two different locations, among which the classifiers are only partially trained for attacks on one location. The accuracy of all baseline classifiers mostly range only within 50% to 70%, which have all been improved to close to 80% after domain-adversarial training. It has been noted that such performance may not meet the practical requirement for deployment, and the limited number of samples (less than 2,000) for these two cases have been identified as the main reason. Nevertheless, the results have still demonstrated that the domain-adversarial training would allow trained models to extend their learned knowledge to attack events at different locations. In a large-scale network that expect more device connections, especially with increasing customer participation via smart meters, the adaptability provided by the TL framework can allow the learned models to handle known threats occurred on new devices and locations in the network, improving the resiliency against attacks while reducing the costs of an exhaustive point-by-point re-training.



(a)



(b)

Figure 7: Detection of threats on unseen locations in (a) Case 5 and (b) Case 6.

3.5 Discussions

3.5.1 Choice of Hyperparameters

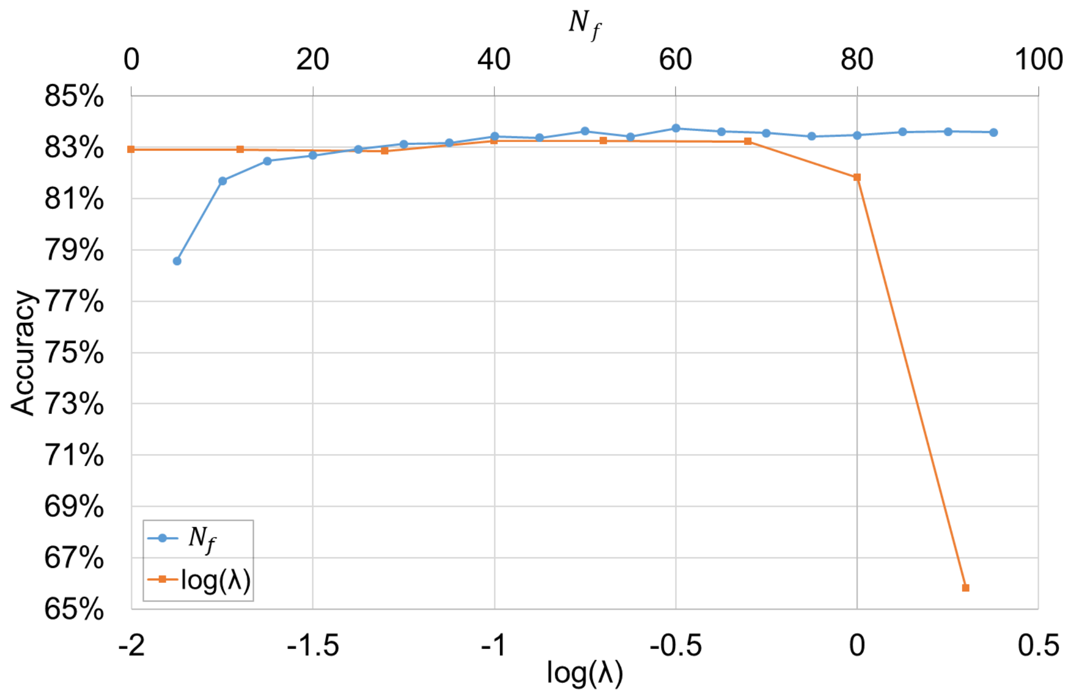


Figure 8: Accuracy with different N_f and λ.

Hyperparameters are controlled variables to be preset for machine learning algorithms, which can significantly affect the performance of machine learning algorithms. We recognize the number of extracted features (N_f) and the value of domain adaptation regularizer (λ) as two such key hyper-parameters and conduct ablation analysis of their influence. As an example, 50 independent experiments were run on Case 3 with random initial weights and following values:

- N_f: from 5 to 95 with a step size of 5;
- λ: from 0.01 to 2.00 at the 1st, 2nd, and 5th decile (0.01, 0.02, 0.05, 0.1, ...).

The results are shown in Fig. 8, with two x-axes above and below the figure. The accuracy increases as N_f increases or λ decreases; the value peaked at 83.7% when $N_f = 60$ and at 83.6% for $\lambda = 0.5$, respectively, which were chosen as the best-performing parameter for the experiments. The comparison suggested that a relatively large N_f and a small λ may be the suitable choice for future studies.

3.5.2 Other Discussions

Results have shown that all baseline classifiers can benefit significantly from the domain-adversarial training and demonstrate robust performance against unseen types and locations of threats. Discussions on the hyper-parameters have also been provided for deployment in practice.

The power of adversarial domain adaptation inspires us of extending to complex scenarios and larger systems. The peak load in power grid can change dynamically over time. In this respect, the distribution of physical measurements could vary significantly. Domain adaptation may also help withstand the change of attack pattern. The limitation would be addressed in Chapter 4.

3.6 Summary

ML has been extensively investigated in IDS for the smart grid. However, the assumption of traditional machine learning, training data and testing data have identical data distribution may not always hold in practical systems.

To address this issue, this chapter has proposed a transfer learning scheme based on the domain-adversarial training framework [38] for intrusion detection against unknown threat variants. The scheme is capable of integrating different baseline machine learning classifiers to improve their detection accuracy against unknown threats of different types and locations. Results have shown that all baseline classifiers can benefit significantly from the domain-adversarial training and demonstrate robust performance against unseen types and locations of threats. Discussions on the hyper-parameters have also been provided for deployment in practice. The results of this chapter have been published in [2].

Chapter 4

Semi-Supervised Domain-Adversarial Training Scheme

4.1 Background

As mentioned in Chapter 1, many machine learning approaches presume that training and testing data will share the same feature sets and follow the same or similar distributions [63]. A potential solution to the challenge is TL. By learning a new feature space where inner-class similarity and inter-class distance are preserved, TL methods allow machine learning-based classifiers to retain their performance on incoming data with new classes. The effectiveness of applying TL on detecting unseen types of threats and attacks with different locations has been presented in Chapter 3.

However, in practice, the needs for transfer may come from the data distribution shift [112], which may occur when loads, topology, or other system dynamics change after a certain period of time. The shift can lead to bias in the training data and render the machine learning model intractable against a returning attack after the system dynamics change. TL is still needed to improve the robustness of IDS in such a case.

In addition, considering the real scenarios in the power grid, labeled attack data is often limited and normal data is sufficient. In the target domain, there may only exist normal class data and attack data is absent. The number of classes between the source domain and target domain may be different under such a situation. This doesn't meet the formulation of unsupervised TL, since it requires us to transfer between domains with a different number of classes. For a practical system, we may be able to label some of the normal data in the target domain because we know the system is under normal operation. This could help us to reformulate the unsupervised TL problem into semi-supervised TL.

The aforementioned issues create a strong incentive for effective TL algorithms to leverage the labeled normal data in the target domain and close up the gap for robust and adaptive intrusion detection against persistent threats with distribution shift. Based on the work of Chapter 3, we further extend the domain-adversarial TL to detect the widely studied false data injection (FDI) attack and propose a Semi-Supervised Domain-Adversarial Training (SSDAT) scheme, which extracts novel features to unify data distributions across normal data from two domains and improves classifier robustness against the shift. We tackle the challenge where the data of the attack incidence is rare compared to normal operations and address it by semi-supervised TL from a single-class target domain where the attack incidence is absent.

4.2 Problem Formulation

In this chapter, we focus on a scenario where two consecutive attacks targeted the same grid during different periods when load demands have changed. In TL, this suggests a data distribution shift, where the distribution $P(X)$ of inputs (samples) has changed but the conditional distribution $P(Y|X)$ of outputs (labels) remains the same. The concepts of domain and task in TL are the same as Section 3.2.

We still consider binary intrusion detection, which classifies the data as attack events or normal operations. In realistic scenarios, labeled attack data is limited and normal data is sufficient. The source training data will consist of labeled normal data and attack data. For target data, we could collect normal data to conduct domain adaptation with source data, and then test on unlabeled data from the target domain with similar data distribution. We assume that the source domain contains labeled normal and attack data, the target domain contains labeled normal data and unlabeled testing data to better match the realistic scenarios.

With the shift caused by time, the source domain $\mathcal{D}_S$ and target domain $\mathcal{D}_T$ will have the same feature space $\mathcal{X}$ but different data distributions *P(X)* caused by temporal variations. We denote the labeled source domain as $\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), ..., (x_{S_{n_S}}, y_{S_{n_S}})\}$, the labeled normal data from target domain as $\mathcal{D}_{T_l} = \{(x_{T_{l1}}, y_{T_{l1}}), ..., (x_{T_{ln}}, y_{T_{ln}})\}$, and the unlabeled testing data from target domain as $\mathcal{D}_{T_u} = \{(x_{T_{u_1}}, y_{T_{u_1}}), ..., (x_{T_{u_n}}, y_{T_{u_n}})\}$. The data within each domain are processed and trained non-sequentially and we did not apply strict time series analysis per sample. Nonetheless, the temporal variations between domains are counted to introduce data distribution shifts and the shifts will be alleviated during the training process.

The goal of TL in this chapter is to learn the predictive function $f(\cdot)$ from the source training data to predict labels in the unlabeled target testing data. To achieve this goal, we conduct domain adaptation with labeled normal data from source and target domains through semi-supervised domain-adversarial training, where the inter-class distance and inner-class similarity are both retained. The new feature space maps the shifting data distributions into a single one, where machine learning algorithms can be trained for better classification.

4.3 Methodology

The overall semi-supervised domain-adversarial TL scheme proposed in this chapter is illustrated in Fig. 9. It consists of three steps:

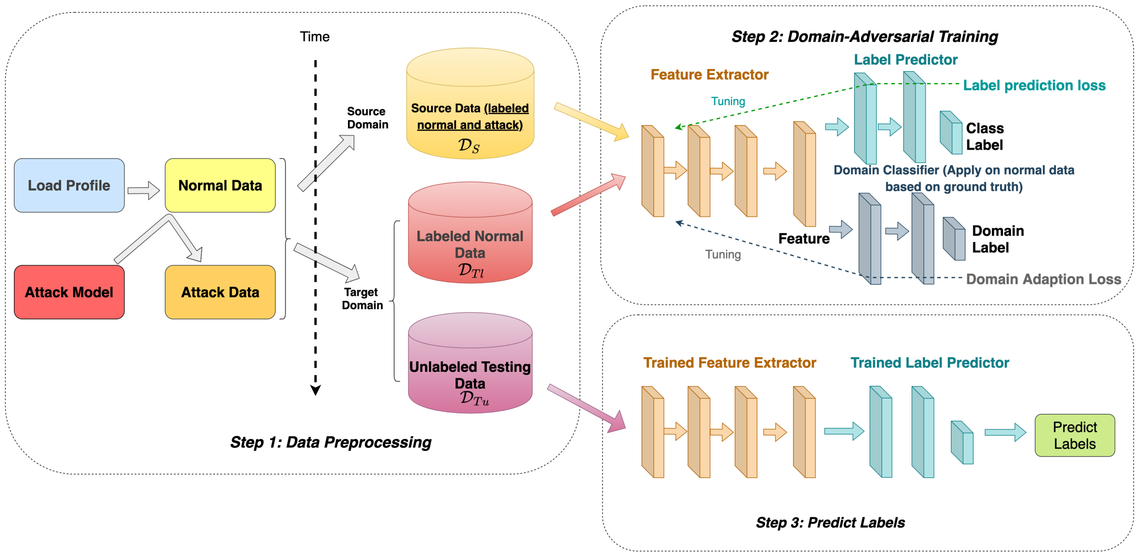


Figure 9: Semi-supervised domain-adversarial transfer learning for IDS in smart grid.

1. Given the rarity of labeled attack, we first collect labeled normal and attack data in the *source domain*, the labeled normal data in the *target domain* for adaptation, and the unlabeled target data in the *target domain* for testing;

2. Train the three networks with the source and target domains to obtain a new feature space via domain-adversarial training, which adapts the normal data over two domains;

3. Map the testing dataset to the new feature space and predict their class labels.

4.3.1 Data Preprocessing

Source Domain

Since the load demand of smart grid varies with time, and the physical measurements will also vary accordingly, we assume that the data distribution within a pre-defined time window is similar and thus could be treated as one domain. Then labeled attack data and normal data are combined to form the source domain, and different source domains could be defined according to the trends of load demand.

Target Domain

For the labeled target normal data and testing dataset, we choose the same size time window as source domain to collect the data. The labeled target normal data is collected one day before the testing data. It will serve as the adaptation data in the semi-supervised domain-adversarial training to help the model to adapt to target data distribution. The testing data will be collected online with the same time periods as labeled normal data so the data distribution will be similar.

4.3.2 Semi-Supervised Domain-Adversarial Training

The design of SSDAT is based on the consideration that in the context of cyber-physical security monitoring, labeled attack data can be extremely rare compared to the labeled normal data that are constantly sampled. We will have to face a rarity of positive class distribution problem in the target domain. The SSDAT is adopted for the following reasons:

- System dynamics will change over time and data distribution shift will occur for both normal data and attack data;
- The source domain contains labeled normal and attack data. The target domain contains labeled normal class data, the number of classes between source and target domains are unequal during the training process. Which creates a semi-supervised TL setting;
- As discussed in Section 4.1, normal operation data is always sufficient in smart grid while labeled attack data is limited because of its short occurrence. On one hand, the insufficient attack data will limit the model generalization ability. On the other hand,

normal data could provide out-of-distribution samples similar to unlabeled testing data and improve the robustness of TL.

To address this challenge, after the feature extraction, the normal data from the source domain and target domain will be fed into the domain classifier. The gradient from domain loss will be reversed when in the feature extractor to increase the domain similarity so that the data distribution shift can be mitigated. Both normal and attack data will be used for training the label predictor. Unlike the setting of unsupervised TL, where the source domain contains labeled data and the target domain only contains unlabeled data, our semi-supervised TL assumes that labeled normal data is available in the target domain.

During the training phase, the normal data from two domains is used in the domain-adversarial training to find the mapping of both domains. For the source domain, both labeled normal and attack data will be used for training the label predictor.

Following the same mechanism in Chapter 3, we select the random forest classifier [113] as the new label predictor in SSDAT, which will be trained on the embeddings obtained from the labeled source data through the feature extractor after the domain-adversarial training.

The detailed training steps are:

1. Collect source domain, target domain, testing data according to the specifications in Section 4.3.1;

2. Feed forward the batch of source domain data and target domain data in the feature extractor to extract the features;

3. Feed labeled source normal data and attack data as well as class labels into the label predictor to calculate the prediction loss, feed the source normal data and target normal data as well as domain labels into domain classifier to obtain the domain loss;

4. Back-propagate the prediction loss and domain loss in label predictor and domain classifier respectively, and reverse the gradient of domain loss at the last layer of feature extractor during the backpropagation;

5. Replace the label predictor with a customized classifier and train the feature extractor as well as the label predictor with extracted features and corresponding class labels.

4.4 Experiments and Results

4.4.1 Attack Model

Over the last two decades, various attack models have been developed to analyze and enhance the cyber-physical security of smart grid [17]. Among them, the false data injection (FDI) attack [114] stands out as one of the most studied threat models. As FDI attacks exploit a mathematical vulnerability in the residual-based bad data detector (BDD) to inject false data onto measurements without raising alarms, it posed a severe threat to power system state estimators (PSSE) and the energy management systems. Successful FDI attacks can introduce arbitrary errors into certain state variables and cause the system operator to perform misinformed control actions, which may result in physical damage and monetary loss [115]. Specifically, the FDI targets the DC state estimation is defined as [116]:

$$\mathbf{z} = \mathbf{Hx} + \mathbf{n} \tag{9}$$

where $\mathbf{z}$ is the known measurement, $\mathbf{H}$ is the Jacobian matrix of power grid topology and $\mathbf{x}$ is the unknown state variable. To identify corrupted measurements, the BDD utilizes statistical tests based on the residual between observed and estimated measurements: $\mathbf{r} = \mathbf{z} - \mathbf{H\hat{x}}$, where $\hat{x}$ is the estimated state variable solved by the weighted least square method. The normalized L_2-norm of $\mathbf{r}$ is then compared with a preset threshold τ to detect the bad data.

The FDI attack model can be written in the following form:

$$\mathbf{z_a} = \mathbf{z} + \mathbf{a} = \mathbf{Hx} + \mathbf{n} + \mathbf{a} \tag{10}$$

where $\mathbf{a}$ is the injected attack vector and $\mathbf{z_a}$ is the manipulated measurements. In the stealthy FDI attack, we assume the attacker has the knowledge of $\mathbf{H}$. In order to bypass the bad data detection and manipulate the states of buses, a targeted false state $\mathbf{x_a}$ is generated by $\mathbf{x_a} = \mathbf{x} + \mathbf{c}$, where $\mathbf{c} \sim N(0, \sigma_c^2)$ is the false state error injected into the system. The attack vector $\mathbf{a}$ is computed by $\mathbf{a} = \mathbf{Hc}$ and injected into the measurements $\mathbf{z}$ by $\mathbf{z_a} = \mathbf{z} + \mathbf{a}$. Let $\mathbf{r} = \mathbf{z} - \mathbf{Hx}$ be the remain residual for bad data detection. Then the new residual $\mathbf{r_a}$ will remain the same and bypass the residual-based bad data detection:

$$\begin{aligned} \mathbf{r_a} = \mathbf{z_a} - \mathbf{Hx_a} &= \mathbf{z} + \mathbf{a} - \mathbf{H}(\mathbf{x} + \mathbf{c}) \\ &= (\mathbf{z} - \mathbf{Hx}) + (\mathbf{a} - \mathbf{HC}) \\ &= \mathbf{z} - \mathbf{Hx} \end{aligned} \tag{11}$$

The pre-attack measurements are obtained from realistic load demands over multiple consecutive days, which represents the challenging data distribution shifts to classic machine learning and calls for TL against new attacks occurring at different hours of the day with different load demands. Some data samples for normal and attack data at the same time period are show in Fig. 10.

4.4.2 Experiments Setup

Normal Data Simulation

We chose the IEEE 30-bus system [4] for simulation and evaluation of the performance, whose topology is illustrated in Fig. 11. There are 30 buses and 41 branches with a total

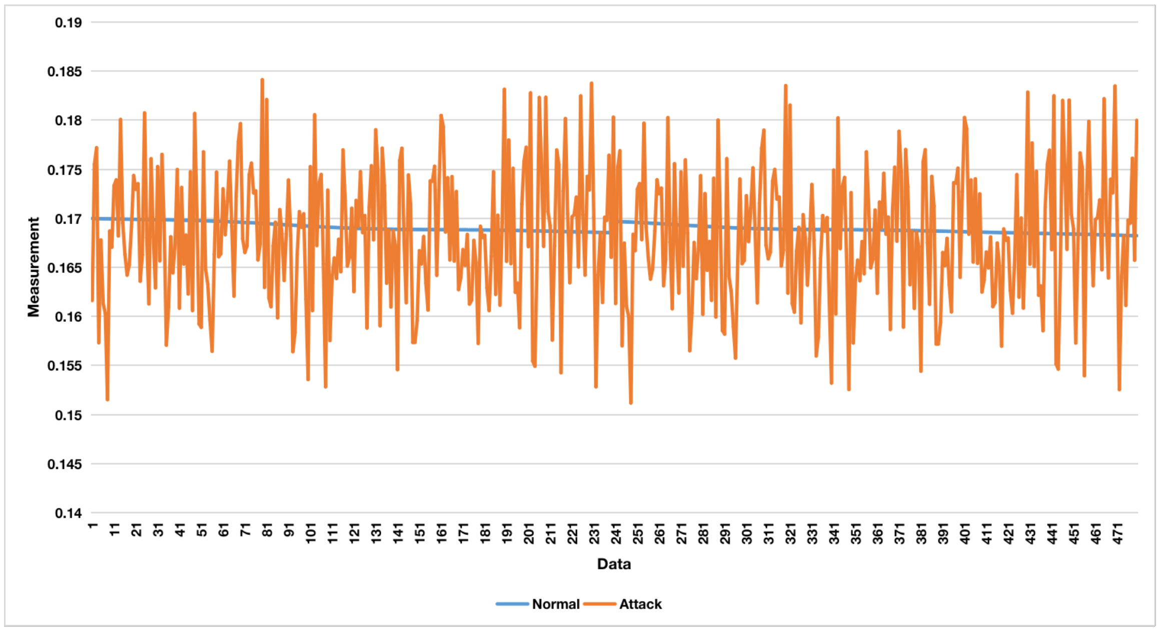


Figure 10: The normal and attack measurements.

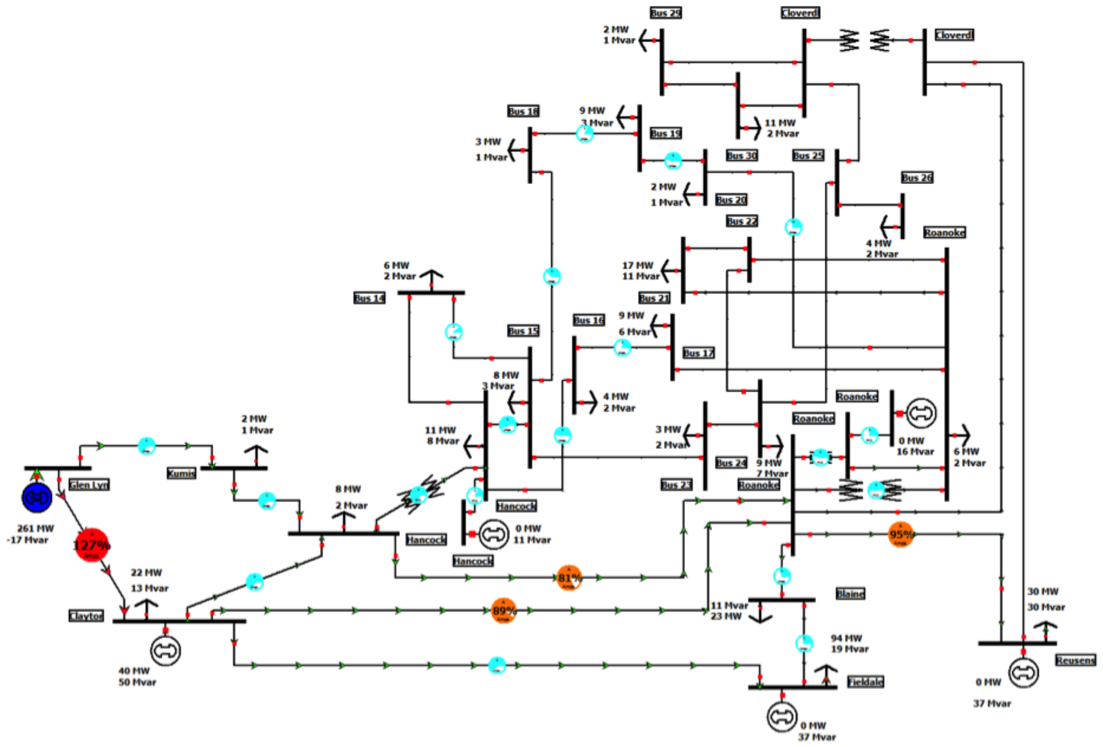


Figure 11: The IEEE 30-bus system by the Illinois Center for a Smarter Electric Grid (ICSEG) [4].

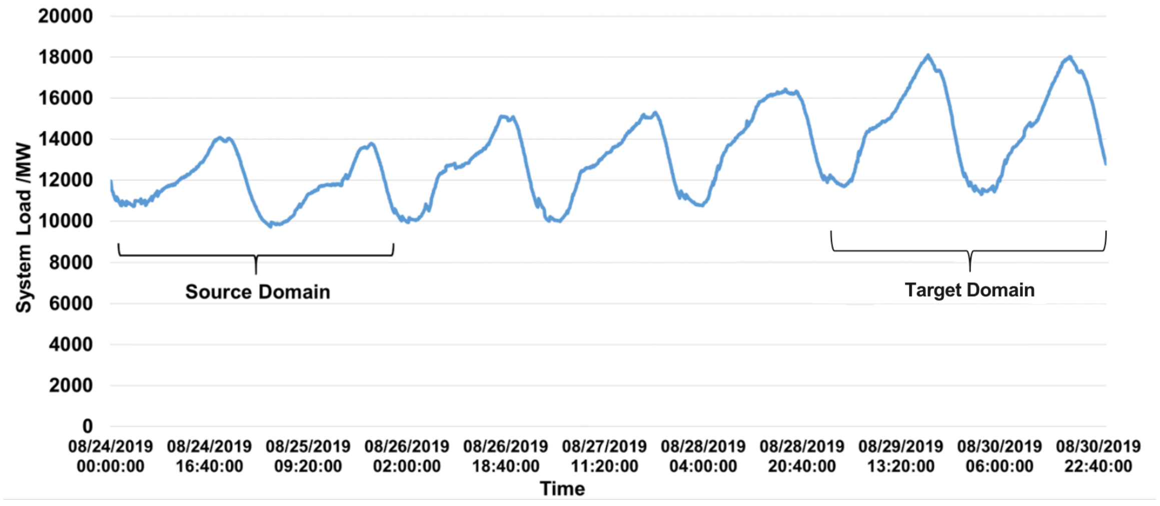


Figure 12: Load demand of ISO New England between Aug. 24 to 30, 2019 [5].

load demand of 189.2 MW. A total of 142 measurements are used to estimate 30 state variables under the DC model.

To set up realistic load variation on this static benchmark, we obtained public data from ISO New England [5] and synthesized the normal operating points (OPs) over a week. We selected one week demand from August 24 to 30, 2019, as shown in Fig. 12, to synthesize a typical weekly load curve for the IEEE 30-bus system. The demand was reported every 5 minutes, or 288 samples per day. By assuming the default load of the 30-bus system the peak load of the week (100%), we calculated all OPs over the 5-minute intervals using the DC optimal power flow (DC-OPF) solver in MATPOWER to collect the normal measurement data of the system.

As illustrated in Table 9, we followed the load variations at different hours of the day to create the source and target domains based on 4-hour time windows to best capture different patterns of data distribution in the 30-bus system. We assumed that the attack was launched on Day 0, and normal operations have been resumed on Day 1. We assumed that by Day 5 the data recording the attack period on Day 0 have been collected; the data recording the same period of normal operation on Day 1 have also been collected to form the labeled training set for the source domain. Then we assumed that day-to-day domain adaptation is performed until Day 6 when the attack was launched again but possibly at different hours. The target domain thus contains data recorded at corresponding periods on Day 5 when the last domain adaptation was performed.

Attack Data Generation

For the attack, we assumed that the attacker aims to manipulate the states with the least efforts. Since the number of compromised meters to manipulate the states varies between buses and depends on the topology $\mathbf{H}$. We searched the number of compromised measurements when attacking a single bus and identified three buses (Buses 11, 13, and 26) that require the minimal number of compromised measurements. The attack vectors were then generated by the FDI attack model $\mathbf{a} = \mathbf{Hc}$ and injected into the measurements $\mathbf{z}$. The false state $\mathbf{c}$ was set with a zero mean and a variance of $\sigma_c^2 = 0.1$.

Balanced Case

Considering that attacks may happen at a different time of the day when the load patterns can be distinctive, we defined 4 cases according to the variation of load demand: the valley, the ascending slope, the peak, and the descending slope. In each case, we assumed that the attack lasted for 4 hours on Day 0 before the system is restored. Once the attack period is located, we also extracted 4 hours of normal operation data on Day 1, recorded during the same 4-hour periods as the attack on Day 0, to create a balanced binary classification dataset in the source domain for SSDAT.

For the target domain, we also used the 4-hour time window but divided Days 5 and 6 into six intervals. On Day 5, we have only recorded normal operations, which contain natural data distribution shifts from load variation. The normal data from the target domain will be used for domain adaptation in SSDAT. For the fair comparison, target domain data will also be treated as labeled training data for baseline classifiers. On Day 6, we assumed

Table 5: Case 1 Setup

#	**Source Domain on Day 0 (attack) and Day 1 (normal)**		**Normal Data from Target Domain on Day 5**	**Testing Dataset from Target Domain on Day 6**
	Cases	Hours		
1	Valley	2–5	4-hour windows of normal operations at different time of the day.	2 hours of attack followed by 2 hours of normal operations.
2	Ascending	11–14		
3	Peak	17–20		
4	Descending	21–24		

that the attack last for 2 hours, which starts at the beginning of one of the six intervals, and the testing dataset is thus also a balanced set composed of 2 hours of attack data followed by 2 hours of normal data.

For each dataset, we have 142 features of physical measurements and 96 instances.

Imbalanced Case

Based on Case 1 we further investigate imbalanced cases to explore the performance when the attack data has different proportions in testing data. We keep the source domain and normal data from target domain the same as Case 1. For the testing dataset, we create cases by adjusting the percentage of 4-hour time window attack data as 25% and 75% separately. We shift the one attack hour for 25% cases and one normal hour for 75% cases to generate 4 sub-cases for each time window. In this chapter, We use the F_1 score to measure the accuracy of imbalanced cases [117].

Similar as Chapter 3, we have chosen four baseline classifiers to compare classic machine learning classifiers with the SSDAT: Artificial Neural Network (ANN) [110], Support Vector Machine (SVM) [107], Classification and Regression Tree (CART) [109], and Random Forest (RF) [113]. All classifiers are implemented in Scikit-learn [111] with manually optimized parameters. The hyperparameters are shown in Table 7.

Table 6: Case 2 Setup

#	**Source Domain on Day 0 (attack) and Day 1 (normal)**		**Normal Data from Target Domain on Day 5**	**Testing Dataset from Target Domain on Day 6**
	Cases	Hours		
1	Valley	2–5	4-hour windows of normal operations at different time of the day.	1 hour attack as 25% cases and 3 hours attack as 75% cases.
2	Ascending	11–14		
3	Peak	17–20		
4	Descending	21–24		

As illustrated in Table 6, the adaptation set consists of 4 hours normal data with the same hours on Day 5 as testing set. For data in the source domain, we set up 4 scenarios according to the variation of load demand: the valley, the ascending slope, the peak, and the descending slope. Each scenario contains 4 hours of data, evenly mixed with 4 hours attack data from Day 0 and 4 hours normal data with the same time period from Day 1.

The target domain contains measurements from Day 6, which is also divided into six 4-hour time windows evenly mixed normal and attack data for each. Day 5 is assumed to be all normal data, which is also divided into 6 slots with similar data distribution with day 6 slots as shown in Fig. 12. The sliding time window creates natural data distribution shifts in the target domain for the evaluation of detection performance with and without TL.

Similar to Chapter 3, we choose Support Vector Machine (SVM) [107], Random Forest (RF) [108], Classification and Regression Tree (CART) [109], and Artificial Neural Network (ANN) [110] as the baseline classifiers and implement them with the Python Scikit-learn library [111]. The hyperparameters are shown in Table 7. The experiments are running on Windows 10 64 bit operating system with Intel Core i7-8700 CPU, 16GB ram and NVIDIA Quadro P620 GPU.

Table 7: Hyperparameter Setting

Classifiers	Hyperparameter
SVM	kernel = ’rbf’; C = 1.
RF	number of estimators = 100
CART	max depth = 5
ANN	architecture of hidden layers = 50-25-10
SSDAT	domain adaptation regularizer = 0.5
	number of extracted features = 60

Table 8: Comparison of SSDAT and Machine Learning Classifiers against Returning Attacks at Different Hours

Cases	Source Hours	Target Hours	SSDAT	ANN	SVM	CART	RF	Best-Case Margin	Worst-Case Margin
1	2-5 (Valley)	1-4	**82.6%**	81.2%	81.3%	77.2%	81.9%	+5.4%	+0.7%
		5–8	**88.9%**	84.6%	87.5%	76.3%	81.0%	+12.6%	+1.4%
		9–12	**86.6%**	84.7%	81.3%	71.8%	76.2%	+14.8%	+1.9%
		13–16	**86.5%**	85.1%	72.9%	69.3%	72.0%	+17.2%	+1.4%
		17–20	**82.9%**	70.5%	64.6%	67.9%	66.0%	+18.3%	+12.4%
		21–24	**80.3%**	70.4%	68.8%	68.3%	69.0%	+12.0%	+9.9%
2	11-14 (Ascending)	1-4	95.3%	91.7%	79.2%	98.1%	**99.7%**	+16.1%	-4.4%
		5–8	**95.9%**	95.8%	87.5%	72.5%	87.3%	+23.4%	-0.1%
		9–12	**85.5%**	58.3%	81.3%	71.0%	79.0%	+27.2%	+4.2%
		13–16	**81.6%**	54.2%	70.8%	71.7%	78.1%	+27.4%	+3.5%
		17–20	**87.7%**	51.7%	70.8%	70.1%	74.7%	+36.0%	+13.0%
		21–24	**80.2%**	50.5%	68.8%	77.1%	79.8%	+29.6%	+0.3%
3	17-20 (Peak)	1-4	94.4%	93.7%	79.2%	91.2%	**95.4%**	+15.2%	-1.0%
		5–8	96.4%	91.7%	87.5%	97.2%	**97.7%**	+8.9%	-1.3%
		9–12	**85.3%**	75.6%	75.0%	66.8%	76.6%	+18.5%	+8.7%
		13–16	**91.1%**	70.7%	75.0%	66.9%	78.0%	+24.2%	+13.1%
		17–20	**85.1%**	68.9%	62.5%	62.9%	74.0%	+22.6%	+10.1%
		21–24	**94.3%**	74.4%	79.2%	83.0%	84.8%	+19.9%	+9.5%
4	21-24 (Descending)	1-4	94.9%	97.9%	85.4%	98.1%	**99.4%**	+9.5%	-4.5%
		5–8	96.9%	**100.0%**	91.7%	94.0%	96.5%	+5.2%	-3.1%
		9–12	**85.6%**	79.0%	60.4%	76.3%	83.4%	+25.2%	+2.2%
		13–16	**85.4%**	82.2%	60.4%	72.2%	83.4%	+25.0%	+2.0%
		17–20	**82.2%**	63.2%	58.3%	68.0%	78.6%	+23.9%	+3.6%
		21–24	**83.6%**	69.2%	56.3%	73.2%	80.8%	+27.3%	+2.8%

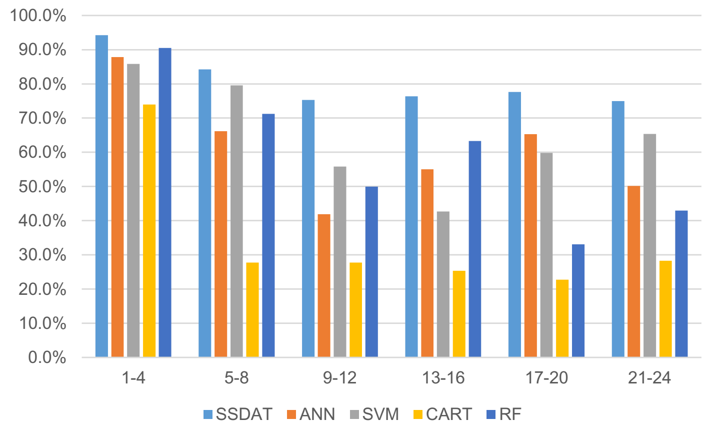
4.4.3 Results and Analysis

The detection accuracy for balanced cases of all classifiers over the 4 cases is shown in Table 8. Overall, the scheme shows robust performance in most of the cases and better than other baseline classifiers in 19 of the 24 sub-cases. The best-case improvement reaches +36.0% compared to ANN during Hours 17–20 in Case 2. The results suggested that SSDAT can retain high accuracy when the same attack occurs at different hours while the baseline classifiers fail to adapt.

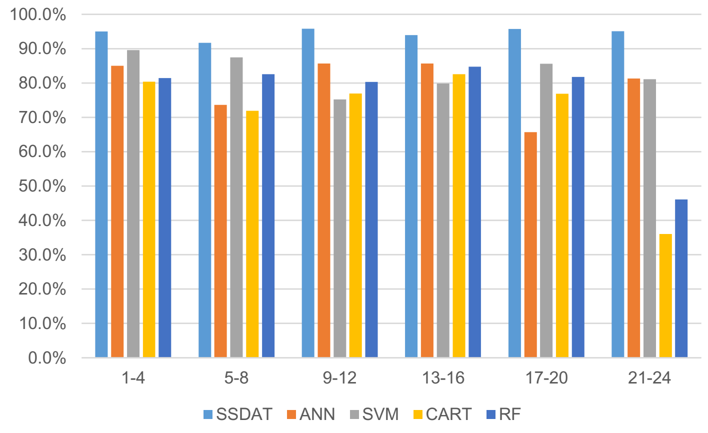
It is notable that during Hours 1–4 and 5–8 on Day 5, the performance of some baseline classifiers are better than the SSDAT. The reason is that most of the load demand of Days 0 and 1 in the source domain has a significant overlap with the Hours 1–4 and 5–8 on Days 5 and 6. The overlapping demand suggests limited distribution shifts, which contributes to the performance of baseline classifiers. Outside of these hours, however, SSDAT achieves better accuracy. The observation poses an interesting question on when the adaptation is indeed needed and how to identify such moments.

The averaged F_1 scores over 4 sub-cases of imbalanced cases with "valley" as source domain are illustrated in Fig. 13a and Fig. 13b. The results suggest that when the source domain is "Valley" data, SSDAT can outperform other baseline classifiers among 6 time windows. Especially when there is limited attack data, the SSDAT demonstrates significant improvements. The reason is that the source valley data has no overlap with the target domain. For other source domains not presented, there are still some cases in Hours 1–4 and 5–8 when some baseline classifiers achieve better accuracy. The reason is consistent with balanced cases.

The results have shown that the proposed scheme can effectively tackle the rarity of attack samples and achieve robust performance against data distribution shifts than classic machine learning classifiers.



(a)



(b)

Figure 13: F_1 scores for the "Valley" period as the source domain and (a) 25% attack data in testing and (b) 75% attack data in testing.

4.5 Discussions

The results have been published in [39], however, after the publication, we noticed that there is one limitation in the experiments, which is that regarding the use of labeled target data, there could be another case in the results to evaluate the performance of using the normal data from both source and target as well as the attack data from the source during the SSDAT, and present the results on the unlabeled target domain.

4.6 Summary

The smart grid faces increasingly sophisticated cyber-physical threats, against which machine learning (ML)-based intrusion detection systems have become a powerful and promising solution to smart grid security monitoring.

However, many ML algorithms presume that training and testing data follow the same or similar data distributions, which may not hold in the dynamic time-varying systems like the smart grid. The effectiveness of transfer learning on detecting unknown types of threats and attacks at different locations has been presented in Chapter 3.

In practical power systems, as operating points may change dramatically over time, the resulting data distribution shifts could lead to degraded detection performance and delayed incidence responses. Besides, unsupervised transfer learning requirements may not hold when the number of classes between source and target domain is different due to the absence of attack data in the target domain.

To address these challenges, this chapter has presented a semi-supervised scheme based on domain-adversarial training to transfer the knowledge of known attack incidences to detect returning threats at different hours and load patterns. Using normal operation data of the ISO New England grids, the proposed scheme leverages adversarial training to adapt learned models against new attacks launched at different times of the day. The effectiveness of the proposed detection scheme is evaluated against the well-studied false data injection attacks synthesized on the IEEE 30-bus system, and the results demonstrated the superiority of the scheme against persistent threats recurring in the highly dynamic smart grid. The results of this chapter have been published in [39]

Chapter 5

Spatiotemporal Domain-Adversarial Training Scheme

5.1 Background

As discussed in Chapter 3 and Chapter 4, TL aims at aligning the data distribution shift between different domains and could improve the robustness of machine learning models. However, most of the existing TL works focus on implementing sophisticated models to achieve better performance and little attention has been paid to the determination of when to transfer. Especially in the field of CPS, data distribution shift may or may not occur and the techniques of detecting such shifts will help to reduce the overhead of frequent transferring.

Since most real world data in CPS are rich in spatiotemporal information, domain adaptation TL may also need to consider both by extracting spatial and temporal features concurrently. The domain adversarial training technique can help in such an extraction process.

Motivated by the remaining gaps, this chapter proposes a Spatiotemporal Domain-Adversarial Training (STDAT) scheme based on DANN [38]. The proposed scheme is two-fold, the first step is to collect classification accuracy on known benchmark datasets, and leverage Kullback–Leibler (KL) divergence [118] and Maximum Mean Discrepancy (MMD) [119] as the divergence measurement metrics between training and testing datasets to build connections with computed accuracy, and perform transferability analysis by providing a detector to determine when to apply TL. Then the second step is to apply a spatiotemporal feature extractor in domain adversarial training to alleviate the data distribution shift between source and target domains and improve the detection accuracy of the classifier.

5.2 Problem Formulation

Same as Chapter 4, this chapter considers the covariate shift, where only the input distribution $P(X)$ changes and the conditional distribution $P(Y|X)$ remains the same. The covariate shifts often occur in CPS intrusion detection scenarios because the system variations and attack variations will have an influence on the normal and attack data distribution. The system variations may be caused by different load demands, normal operations, or topology changes. The attack variations could arise when either the load or the locations of attack launched differ from benchmark datasets. Such variations will inevitably lead to shifts in CPS data, which can be challenging for traditional learning-based IDS.

Aware of the aforementioned scenarios, this chapter investigates the attack detection in smart grid taking both temporal and spatial characteristics into consideration and propose the Spatiotemporal Domain-Adversarial Transfer Learning (STDAT) model to alleviate the impact of data distribution shift. We focus on the scenarios where two consecutive attacks targeted the same grid during different periods when load demands have changed and/or the injected bus of the two attacks vary.

Same as previous Chapters, the intrusion detection in smart grid is formulated in the context of TL as follows. A domain $\mathcal{D}$ consists a feature space $\mathcal{X}$ and a marginal probability distribution *P(X)*. Given a specific domain $\mathcal{D}$= {$\mathcal{X}$,*P(X)*}, a task consists of a label space $\mathcal{Y}$

and an objective predictive function $f(\cdot)$ from $\mathcal{X}$ to $\mathcal{Y}$ that will be learned from the training data.

This chapter considers the binary intrusion detection problem in smart grid, which classifies the multivariate time series measurements data as attack events or normal operations. The source domain $\mathcal{D}_S$ and target domain $\mathcal{D}_T$ will have the same feature space $\mathcal{X}$ but different data distributions *P(X)* with the shift caused by system or attack dynamics. We denote the labeled source domain as $\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), ..., (x_{S_{n_S}}, y_{S_{n_S}})\}$, and the unlabeled target domain as $\mathcal{D}_{T_s} = \{(x_{T_{s_1}}), ..., (x_{T_{sn}})\}$, Where $x \in R^{C \times T}$, C is the dimension of feature space, and T is the length of the time series. For the problem of intrusion detection in smart grids, we assume that source domain data consists of labeled normal operations and attack data. Unlabeled target domain contains normal data and attack data with different data distribution from the source domain.

Similar to the previous chapters, to generalize a classifier trained on source domain to a target domain, the main challenges are to determine when to apply transfer leaning and how to find the best-performing mapping for multivariate time series CPS data, which is tackled by the STDAT as follows.

5.3 Methodology

5.3.1 Divergence-based Transferability Analysis

The transferability analysis is the approach to conduct analysis on the data and determine if there is a need to apply TL. Since when to transfer depends on whether there is a data distribution shift in target domain. Such analysis could lead to a data distribution shift detection problem.

The first step in our proposed scheme is to apply the data distribution shift detection between the target domain and the source domain.

Current approaches of detecting data distribution shift are mainly targeting identifying the out-of-distribution samples and ignore to build the connection between the distribution detection and answering when to apply TL. Rabanser *et al.* [120] investigated methods for detecting data distribution shift, and revealed that a two-sample-testing-based approach, using pre-trained classifiers for dimensionality reduction, performs best. Yet the comparison results are mostly reported on image and video datasets, whereas the characteristics of CPS data will vary significantly. Unlike the RGB values that have uniform color space and pixel connectivity, CPS data are multivariate and contain spatiotemporal information. This creates a strong incentive to adopt transferability analysis for CPS applications.

The objective of our data distribution shift detection is to not only identify the shift but also provide the intensity of divergence so we could utilize the empirical data to measure and test if the divergence is considerable enough to apply TL. To this end, we investigate the applicable divergence measurements and conduct experiments simulated multivariate CPS data to validate the performance.

We select two widely used divergence metrics, KL divergence and MMD, in this paper as the approaches for measuring the distance between two data distributions. In existing works, KL divergence is commonly used to calculate the divergence between two distributions [121] or as the regularizer in the loss function to minimize the distance between domains [122, 123]. The MMD-based method is one of the most fruitful lines for domain adaptation [124, 125, 126], which could also act as the regularizer in the objective function and help to decrease the distribution divergence between extracted features. In our transferability analysis, KL divergence and MMD will be mainly used to measure the distance between domains in the smart grid and provide insights to answer when to apply TL.

The KL-divergence, also known as the relative entropy, between two probability density functions $p(x)$ and $q(x)$,

$$D_{KL}(P||Q) = \int p(x) log \frac{p(x)}{q(x)} dx \tag{12}$$

Since directly calculating the KL divergence between two multivariate datasets is non-trivial, we choose two methods for KL divergence estimation.

Gaussian Mixture Models based KL estimation

For the first solution, we adopt the work of and J.R. Hershey *et al.* [127] and consider the two datasets are Gaussian Mixture Models (GMM). The marginal densities of $x \in R^d$ under p and q are

$$\begin{aligned} p(x) &= \sum_a \pi_a \mathcal{N}(x; \mu_a; \Sigma_a) \\ q(x) &= \sum_b \pi_b \mathcal{N}(x; \mu_b; \Sigma_b) \end{aligned} \tag{13}$$

To estimate $D(P||Q)$ for large values of d we could conduct Monte Carlo simulation. Using n i.i.d. samples $\{x_i\}_n^{i=1}$, we have:

$$D_{MC}(P||Q) = \frac{1}{n} \sum_{i=1}^{n} log \frac{p(x_i)}{q(x_i)} \rightarrow D(P||Q) \tag{14}$$

The variance of the estimation error could be decreased when $n \rightarrow \infty$.

k-Nearest-Neighbour based KL estimation

For the second solution, we choose the Fernando *et al.*'s work [128] and use kNN density estimation as an intermediate step to estimate the KL divergence. Given a set with n i.i.d. samples from $p(x)$ and m i.i.d. samples from $q(x)$, we can estimate the $D(P||Q)$

from a k-NN density estimate as follows:

$$D_k(P||Q) = \frac{1}{n}\sum_{i=1}^{n} log\frac{p_k(x_i)}{q_k(x_i)} = \frac{d}{n}\sum_{n}^{i=1} log\frac{r_k(x_i)}{s_k(x_i)} + log\frac{m}{n-1} \tag{15}$$

where

$$p_k(x_i) = \frac{k}{n-1} \times \frac{\Gamma(\frac{d}{2}+1)}{\pi^{\frac{2}{d}}(r_k(x_i))^d} \tag{16}$$

$$q_k(x_i) = \frac{k}{m} \times \frac{\Gamma(\frac{d}{2}+1)}{\pi^{\frac{2}{d}}(s_k(x_i))^d} \tag{17}$$

$r_k(x_i)$ and $s_k(x_i)$ are the Euclidean distances to the kth nearest-neighbour of x_i and $\pi^{\frac{2}{d}}/\Gamma(\frac{d}{2}+1)$ is the volume of the unit-ball in R^d.

Maximum Mean Discrepancy

MMD is a non-parametric distance estimate between distributions based on the Reproducing Kernel Hilbert Space (RKHS) proposed by Borgwardt *et al.* The empirical estimate of the distance between P and Q, as defined by MMD, is

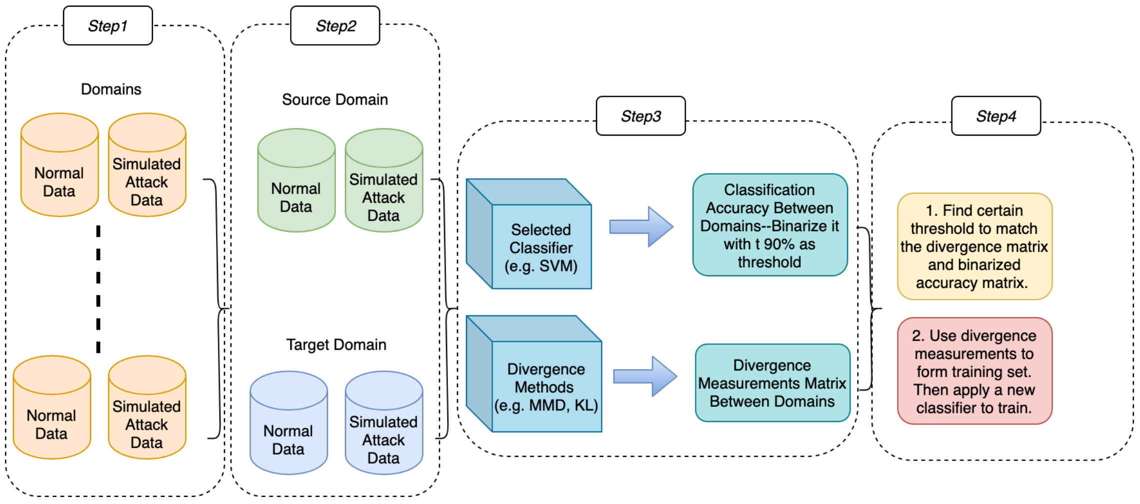
$$D_{MMD}(P||Q) = \left\| \frac{1}{n_1}\sum_{i=1}^{n_1}\varphi(x_i) - \frac{1}{n_2}\sum_{j=1}^{n_1}\varphi(x_j) \right\|_H \tag{18}$$

where $\varphi(x)$ is the feature space map from x to H.

Then we will leverage the selected metrics to calculate data distribution divergence and build the connections with classification performance. As illustrated in Fig 14, followings are the steps of our proposed data distribution detection method:

1. Define the domains: Sample normal data with specific sample rate, time period. Simulate attack data and combine it with normal data as a domain for binary classification. Generate several domains across different time periods.

2. Generate dataset pairs between domains: one domain as source training and another



Step1
Domains
Normal Data
Simulated Attack Data
Normal Data
Simulated Attack Data
Step2
Source Domain
Normal Data
Simulated Attack Data
Target Domain
Normal Data
Simulated Attack Data
Step3
Selected Classifier (e.g. SVM)
Classification Accuracy Between Domains--Binarize it with t 90% as threshold
Divergence Methods (e.g. MMD, KL)
Divergence Measurements Matrix Between Domains
Step4
1. Find certain threshold to match the divergence matrix and binarized accuracy matrix.
2. Use divergence measurements to form training set. Then apply a new classifier to train.

Figure 14: Divergence-based transferability analysis.

as target testing. Use the base model(e.g. SVM) to get the binary classification accuracy on target domain. Set certain accuracy (e.g. below 90%) as the threshold to binarize the accuracy matrix with 1 as accuracy above the threshold and 0 as accuracy below the threshold. The labels are the indicators to identify data distribution change;

3. Use Kullback–Leibler divergence and Maximum Mean Discrepancy (MMD) with different kernels to calculate the divergence between domains of each pair matching. Find the threshold t to binarize the divergence matrix and minimize the mismatching between binarized divergence matrix and binarized accuracy matrix. The divergence no greater than t will be the cases where data distribution is identical.

4. Use the metrics with high matching accuracy in Step 3 as features to form a new dataset for training and validation. Use the binarized accuracy matrix as labels and apply another base classifier to train a new detector.

5.3.2 Spatiotemporal Domain-Adversarial Training

As mentioned in previous chapters, domain invariant TL is a widely studied domain adaptation method, the goal is to learn and transfer the domain invariant features from

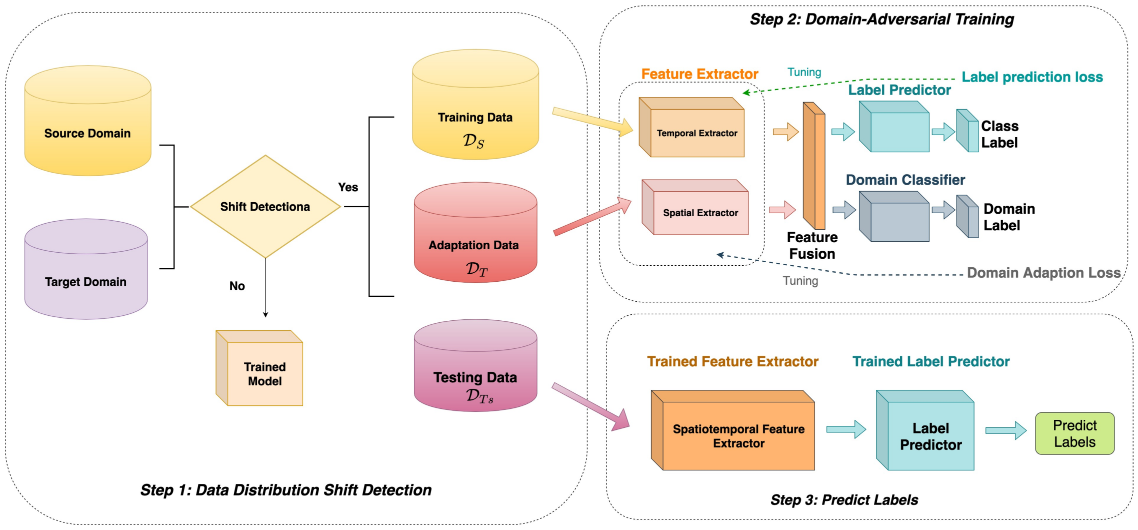


Figure 15: Spatiotemporal domain-adversarial transfer learning for IDS in smart grid.

source domain to target domain so the model trained on labeled source data could generalize well on unlabeled target domain data. The most commonly used domain-invariant method is the domain adversarial neural network (DANN), which usually consists of three components: a feature extractor, a domain classifier, and a label predictor. Each component consists of several layers of neural network in the original design.

As discussed in Section 2.2, only a few studies have considered the time series data in the content of domain adaptation. Purushotham *et al.* [95] proposed variational recurrent adversarial deep domain adaptation (VRADA) and recurrent domain adversarial neural network (R-DANN) and revealed the possibility to incorporate time series analysis into DANN. Wilson *et al.* [96] proposed a Convolutional deep Domain Adaptation model for Time Series data (CoDATS). They select three layers of fully convolutional network instead of RNN as the feature extractor and achieved state-of-the-art performance on Human Activity Recognition dataset. The CoDATS is also selected as our comparison method.

Our proposed STDAT differs from existing work in that STDAT considers both the spatial and temporal information of CPS data by customizing the feature extractor with two parallel fully convolutional networks

Motivated by the success of convolution neural networks on time series classification problems, in order to extract invariant spatiotemporal features, we proposed to leverage two fully convolution neural networks (FCN) [129] as parallel feature extractors to extract spatiotemporal features from the multivariate time series data on time and space dimensions. Specifically, each FCN block consists of 3 layers of 1-D CNN followed by a batch normalization layer [130] and a ReLU activation layer, where the kernel size of each layer is 8, 5, 3 without striding. The basic convolution block is

$$\begin{aligned} y &= W \otimes x + b \\ s &= BN(y) \\ h &= ReLU(S) \end{aligned} \tag{19}$$

Following is the global average pooling layer and fully connected layer. The extracted features are concatenated horizontally as the last fusion layer of the feature extractor.

After the feature extractor, same as Chapter 4, the label predictor, consisting of layers of neural networks, utilizes the labeled source data for back-propagation training to retain the inter-class distance. The divergence of data from two domains in the embedding space will be mitigated so the trained label predictor will not suffer severe impact of data distribution shift and achieve acceptable performance.

The overall STDAT proposed in this chapter is illustrated in Fig. 15. The scheme consists of four steps:

1. Follow the proposed data distribution shift detection method to obtain the detector;

2. For the new target domain data, determine if there is a data distribution shift where we need to apply the STDAT via data distribution shift detector;

3. Train the three networks with the source and target domains to obtain a new feature space via domain-adversarial training, which eliminates the divergence between

domains;

4. Map the testing dataset to the new feature space and predict their class labels.

The main difference of this scheme compared to Chapter 3 and Chapter 4 is that, consider the spatial and temporal information of smart grid, we propose a novel fully convolutional neural network-based spatiotemporal feature extractor to replace the original neural network-based feature extractor to process the multivariate time series CPS data.

5.4 Experiments and Results

In this section, we first introduce the attack model and then experiments setup on transferability analysis. Then we will elaborate our experiment setting to evaluate our proposed methods as well as other baseline models. We choose the IEEE-30 bus system for simulation and the real system load demand data to generate normal operations points to introduce time variations. And we study the False Data Injection (FDI) attack detection problem and demonstrate the binary classification accuracy on selected models. We also perform ablation studies of the proposed method with respect to random initial parameters.

5.4.1 Attack Model

For the attack model, similar to Chapter 4 we choose FDI attack in a multivariate time series context.

For the attack in different time cases, we assumed that the goal of the attacker is to alter the states with the least efforts and launch the attack vector at the same locations across different time periods. We conducted a search over different combinations of buses and identified three buses (Buses 11, 13, and 26) that require the minimal number of compromised measurements. Then the attack vectors were generated by $\mathbf{a} = \mathbf{Hc}$ and the false state **c** was set with a zero mean and a variance of $\sigma_c^2 = 0.1$. Then we select 10 minutes as the time step to transform the attack data into time series data.

For the attack in different locations cases, we choose the same mechanism to generate attack vector, but we assume that the attacker only inject 1 bus when launching the attack, hence for the IEEE 30 bus system we will have 30 source domains and 30 target domains.

5.4.2 Experiments Setup

Divergence-based Transferability Analysis

For the data distribution shift detection, in order to test the method on real CPS scenarios, we select 1 week real load demand (from Jan 1st to 7th 2020) from ISO New England [5] for data simulation and use 4 hours sliding time window to divide each day to 21 domains separately. Each domain consists of normal data and simulated attack data. The domains from day 1 are used for training and day 2 for validation. And data from other days are used for testing. For the domains on day 1, we train an SVM on 1 domain and test on the other domain to acquire the classification accuracy matrix. We use 90% as the threshold to binarize the matrix to 0 and 1 labels which indicate whether there is a data distribution shift that could have a negative effect on traditional machine learning models. Then we calculate the KL divergence and MMD with different kernels between domains as new features for training.

FDI Detection Dataset setup

This chapter selects the IEEE 30-bus system [4] as the simulation scenario and leverages Matlab toolbox MATPOWER to generate data. As illustrated in Fig. 11, the system consists of 30 buses and 41 branches with a total load demand of 189.2 MW. A total of 142 measurements are used to estimate 30 state variables under DC model.

To establish experiments based on realistic scenarios, we obtained public load demand

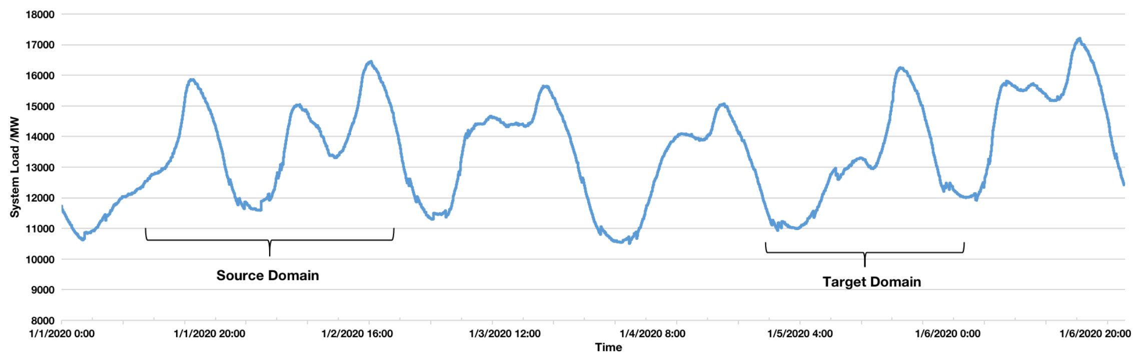


Figure 16: The load demand of ISO New England between Jan. 1 to 7, 2020 [5].

data from ISO New England [5] and synthesize the load demand from January 1 to 7, 2020 as shown in Fig. 16. The demand was reported every 5 minutes, in order to increase the sampling rate and maintain the trend of demand curve, load demand data is interpolated with 1-minute interval by Spline method in MATLAB. We calculated and collected all the normal measurement data over 1-minute intervals through the DC optimal power flow (DC-OPF) solver in MATPOWER, with the assumption that the default load of the 30-bus system is the peak load of the week (100%). To generate time series data, we conducted a sliding window with 10 minutes as the length size and 1 minute as stride size to transform the normal measurement data into sequential data.

To validate the effectiveness of our proposed model, we set up 3 categories of cases as follows. For each dataset of these cases, we have 142 features of physical measurements from the IEEE-30 bus system and 480 instances.

Temporal Variations Cases

Since the load demand and its patterns vary significantly throughout one day, we select the 4-hour time window data as the source domains and target domains to best capture the characteristics of data distributions. As illustrated in Table 9, the source data consists of attack data launched on Day 1 and normal data with the same hours collected on Day 2. Same as our previous work[2], considering that attacks may happen at a different time of the day when the load patterns can be distinctive, we defined 4 cases according to the variation of load demand: the valley, the ascending slope, the peak, and the descending slope.

For the target domain and testing dataset, without loss of generality, we assumed that the attack was launched on Day 5, we also used the 4-hour time window but divided Day 5 into six intervals.

Spatial Variations Cases

For spatial cases, we assume that the load demand pattern will be identical in source and target domains. We select the same 4 hours time window for source and target data whereas choosing different attack locations for the target data. The attack is injected only on one single bus for each domain. By conducting training and testing on 29×29 pairs experiment, the non-transfer methods perform worse when the target bus is 3, 5, 6, thus we select 1-30 buses as the source domain separately and bus 3, 5, 6 as target domain;

Spatiotemporal Variations Cases

For spatiotemporal cases, we assume that time and locations of attack in the target domain both vary from the source domain. To this end, we select "valley" as the source load demand pattern and "9-12" as the target load demand pattern. And we inject 1-30 buses for the source domains and 3, 5, 6 buses for the target domains.

To compare classic machine learning classifiers with the SSDAT, we have chosen four baseline classifiers: We choose Multilayer Perceptron (MLP) [110] and Support Vector Machine (SVM) [107] as the non-transfer machine learning methods. The two methods have demonstrated superior accuracy and computation efficiency in detecting FDI attack. Further, we select the state-of-the-art multivariate time series classification method FCN as a time series non-transfer baseline model. We also choose FCN and Convolutional

Table 9: Temporal Case Setup

#	**Source Domain on Day 0 (attack) and Day 1 (normal)**		**Target Domain**	**Testing Dataset**
	Cases	Hours		
1	Valley	2–5	2 hours of attack followed by 2hours of normal on Day 5.	Balanced data from target domain
2	Ascending	11–14		
3	Peak	17–20		
4	Descending	21–24		

Table 10: Hyperparameter Setting

Classifiers	Hyperparameter
SVM	kernel = ’rbf’; C = 1.
MLP	architecture of hidden layers = 50-25-10
FCN	Conv1D (filters, kernel size): (128, 8)-(256, 5)-(128, 3); padding = ’same’; activation = ’relu’.
CoDATS	domain adaptation regularizer = 0.5 number of extracted features = 128
STDAT	domain adaptation regularizer = 0.5 number of extracted features = 256

deep Domain Adaptation model for Time Series data (CoDATS) as the comparison method which only extracts features from the temporal dimension . All classifiers are implemented in Scikit-learn [111] and Keras with optimized parameters. The hyperparameters are shown in Table 10. The experiments are running on Windows 10 64 bit operating system with Intel Core i7-8700 CPU, 16GB ram and NVIDIA Quadro P620 GPU.

Table 11: Results of Transferability Analysis

Metric	KL+GMM	KL+kNN	MMD (linear)	MMD (poly)	MMD (rbf)	MMD (linear+poly)+SVM	MMD (linear+poly)+RF
Train Accuracy	93.1%	56.3%	97.3%	97.0%	52.1%	**98.6%**	98.0%
Validation Accuracy	86.5%	51.7%	95.4%	93.7%	76.3%	**96.2%**	95.7%
Test Accuracy	81.4%	38.2%	91.6%	91.1%	64.4%	**95.0%**	94.8%

Table 12: Results of SVM for transferability analysis

Domains	1	2	3	4	5	6	7	8	9	10
1	**100.0%**	**100.0%**	**100.0%**	**99.6%**	**99.6%**	87.3%	74.8%	62.7%	50.2%	50.0%
2	**100.0%**	**100.0%**	**100.0%**	**99.6%**	**99.2%**	86.7%	74.2%	62.1%	50.0%	50.0%
3	**97.3%**	**99.8%**	**100.0%**	**99.6%**	**99.6%**	**91.7%**	79.2%	67.1%	54.6%	50.0%
4	81.5%	**94.0%**	**100.0%**	**100.0%**	**100.0%**	**99.8%**	87.9%	75.4%	62.9%	50.6%
5	78.5%	**91.0%**	**100.0%**	**100.0%**	**100.0%**	**99.8%**	**99.8%**	94.6%	82.1%	69.8%
6	73.8%	86.3%	**98.8%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**99.6%**	89.8%
7	68.1%	80.6%	**92.9%**	**99.4%**	**99.8%**	**99.8%**	**100.0%**	**100.0%**	**99.6%**	**99.6%**
8	50.0%	58.1%	70.4%	82.9%	**95.4%**	**99.8%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**
9	50.0%	50.0%	58.1%	70.4%	82.9%	**95.4%**	**99.8%**	**100.0%**	**100.0%**	**100.0%**
10	50.0%	50.0%	50.0%	51.9%	64.4%	76.9%	89.4%	**100.0%**	**100.0%**	**100.0%**

5.4.3 Results and Analysis

We first evaluate different data shift detection methods for transferability analysis on ISO New England data.

Transferability Analysis:

Partial classification results of SVM for transferability analysis are shown in Table 12. By changing the temporal variations through sliding time windows, the classification accuracy will degrade with the increasing temporal divergence between source and target domains. The results suggest that ml classifiers could retain the accuracy when data distribution divergence is not significant and the transferability analysis could help to avoid the overhead of applying transfer learning.

The results of transferability analysis are summarized in Table 11. From the table, we can observe that the MMD with linear and poly kernel have superior performance among the single divergence detection methods. And the combination of MMD linear and poly kernel with SVM as the shift detector could achieve the highest testing accuracy. The latter method takes the advantage of two superior metrics and leverages the SVM to find out the

Table 13: Comparison of STDAT and Machine Learning Classifiers against Returning Attacks at Different Hours

Cases	Source Hours	Target Hours	STDAT	CoDATS	FCN	SVM	MLP	Best-Case Margin	Worst-Case Margin
1	2-5 (Valley)	1-4	**97.0%**	90.3%	69.6%	67.9%	78.3%	+29.0%	+6.7%
		5–8	**100.0%**	90.1%	73.8%	66.7%	77.7%	+33.3%	+9.9%
		9–12	**100.0%**	90.0%	63.6%	65.4%	77.3%	+36.4%	+10.0%
		13–16	**98.5%**	95.4%	84.3%	65.0%	76.7%	+33.5%	+3.1%
		17–20	**90.0%**	83.8%	67.4%	64.2%	76.3%	+25.8%	+6.2%
		21–24	**86.8%**	82.1%	71.9%	62.7%	74.4%	+24.1%	+4.7%
2	11-14 (Ascending)	1-4	**100.0%**	97.7%	75.6%	79.0%	74.2%	+25.8%	+2.3%
		5–8	**100.0%**	94.6%	78.5%	77.7%	73.8%	+26.4%	+5.4%
		9–12	**82.7%**	80.0%	81.8%	76.5%	72.9%	+9.8%	+0.9%
		13–16	**90.0%**	71.4%	77.3%	74.4%	71.5%	+18.6%	+12.7%
		17–20	**100.0%**	100.0%	68.2%	72.5%	70.2%	+31.8%	+0.0%
		21–24	**90.0%**	83.3%	67.1%	71.0%	69.0%	+22.9%	+6.7%
3	17-20 (Peak)	1-4	**98.3%**	89.9%	77.7%	82.9%	73.8%	+24.5%	+8.4%
		5–8	**86.7%**	84.7%	67.5%	80.4%	71.5%	+19.2%	+2.0%
		9–12	**100.0%**	80.0%	72.7%	80.4%	71.5%	+28.5%	+19.6%
		13–16	**88.8%**	80.8%	85.8%	79.9%	70.2%	+18.6%	+3.0%
		17–20	**96.3%**	89.9%	77.3%	77.1%	69.0%	+27.3%	+6.4%
		21–24	**83.2%**	80.2%	75.6%	75.0%	68.1%	+15.1%	+3.0%
4	21-24 (Descending)	1-4	**97.5%**	92.7%	59.2%	90.0%	77.5%	+38.3%	+4.9%
		5–8	**90.5%**	85.5%	66.3%	89.8%	75.8%	+24.3%	+0.7%
		9–12	**90.0%**	80.2%	72.2%	89.4%	75.4%	+17.8%	+0.6%
		13–16	**87.8%**	85.2%	67.6%	87.5%	74.4%	+20.1%	+0.3%
		17–20	**87.2%**	85.0%	72.7%	86.3%	72.9%	+15.1%	+1.6%
		21–24	**89.5%**	85.1%	63.6%	85.0%	71.5%	+25.9%	+4.4%

nonlinear decision boundary for data distribution shift detection.

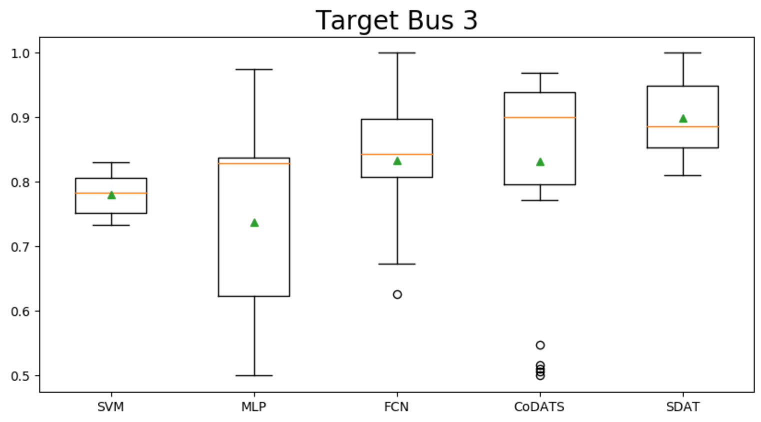
FDI Detection with Temporal Variations:

The detection accuracy of STDAT and 4 baseline methods over the 4 cases is illustrated in Table 13. Overall, the STDAT shows an advanced detection accuracy over other baseline classifiers in all of the cases. The best-case improvement reaches +38.3% compared to FCN during Hours 1–4 in Case 4.

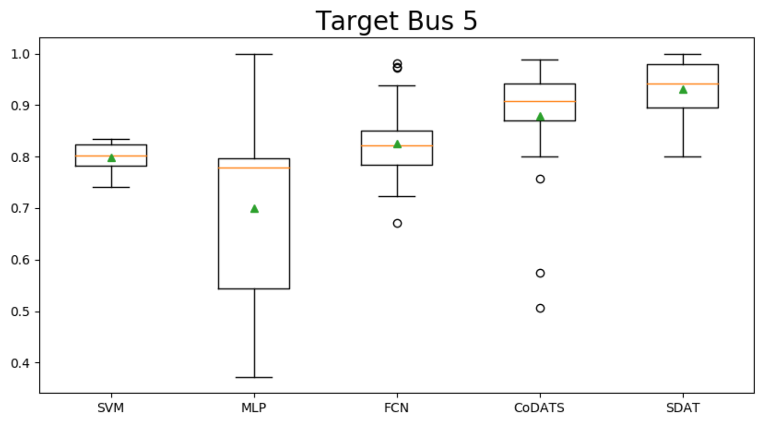
The results suggested that STDAT can extract domain-invariant and features and significantly improve the FDI detection performance while the non-transfer classifiers fail to adapt when there is a data distribution shift caused by variations in load demand. The STDAT achieves better accuracy than CoDATS via extracting discriminative spatial features in parallel to further improve the accuracy.

FDI Detection with Spatial Variations:

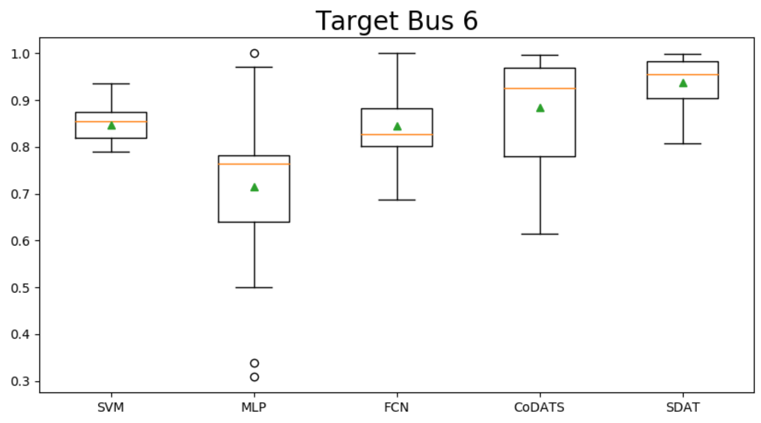
Then we further test out the classifiers on spatial cases. The results of classification



(a)



(b)



(c)

Figure 17: Box plot of accuracy with source attack launched on Buses 1-30 and (a) target attack on Bus 3 (b) target attack on Bus 5 and (c) target attack on Bus 6.

accuracy as well as the average score of accuracy are reported in Fig. 17a, Fig. 17b and Fig. 17c. For the spatial cases, since the data distribution of normal data is similar between source and target domain, the non-transfer methods can identify more attack samples and achieve higher accuracy compared to temporal cases. However, STDAT still demonstrates improvements in all of the cases.

The best-case and worst-case average improvements on non-transfer methods reach 23.1% compared to MLP when the target bus is 5 and 6.5% compared to FCN when the target bus is 5. For CoDATS, the best-case and the worst-case average improvements are 6.7% when the target bus is 3 and 5.3% when the target bus is 5 respectively. The results suggest that STDAT show a robust performance in spatial cases.

FDI Detection with Spatiotemporal Variations:

We further test out the classifiers on spatiotemporal cases. We report the best performance of SVM, MLP and FCN as comparison and demonstrate the average accuracy of 1-30 bus as the source domain and another bus on target domain for each method in Fig. 18.

For the spatiotemporal cases, the data distribution of both spatial and temporal dimensions further degrades the baseline methods and the proposed STDAT could demonstrate a better improvement. The averaged improvement is 17.4% and 5.6% compared to the best of non-transfer methods and CoDATS respectively. The best-case and worst-case average improvements on non-transfer methods reach 29.9% when the target bus is 28 and 4.7% when the target bus is 6. Through sorting by the margin of STDAT over the best of other methods, there are 7 out of the top 12 buses are between 20-30. The buses with the most improvements share similar location information. This indicates that the non-transfer methods perform worse on buses within this region while STDAT could maintain a higher accuracy. For CoDATS, the best-case and worst-case average improvements are 12.3% when the target bus is 27 and 0.2% when the target bus is 4. The results suggest that STDAT show a robust performance on detecting FDI attack with spatial and temporal

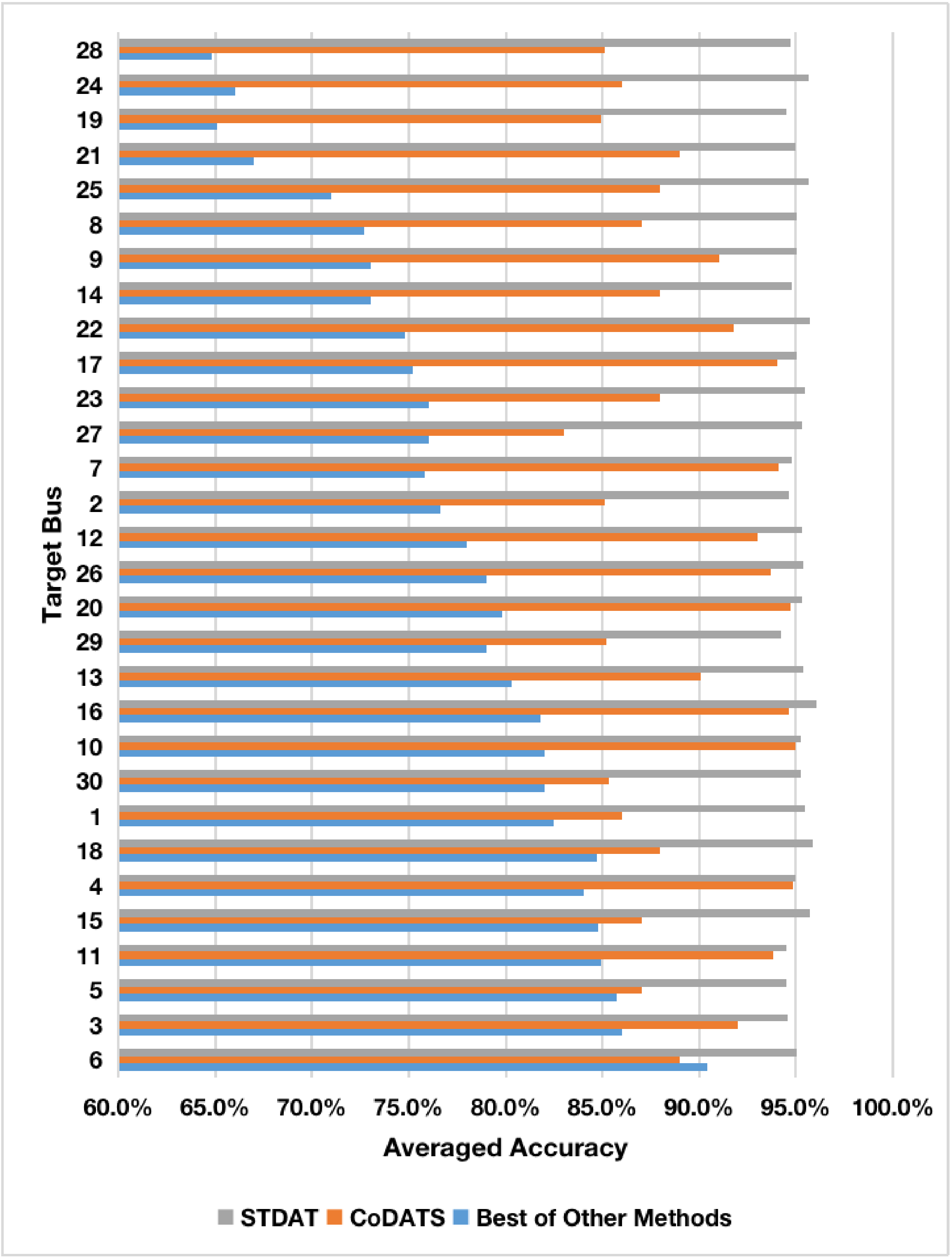


Figure 18: Spatiotemporal cases accuracy.

variants.

5.5 Discussions

The results have shown that the proposed method can effectively tackle the spatiotemporal variations and achieve robust performance against data distribution shifts than classic machine learning classifiers. However, the limitation of STDAT is that it still undergoes overhead of training time when transferring between domains. Meanwhile, it is time-consuming for the model to search initial parameters to converge, the reason may lie in that the complexity introduced by sophisticated design of the network.

5.6 Summary

The effectiveness of transfer learning in improving robustness has been seen in previous chapters. However, when to apply transfer learning and the spatial and temporal information of smart grid haven't been taken into consideration.

To address those challenges, this chapter has presented a spatiotemporal domain-adversarial training scheme to detect data distribution shifts and determine when to apply transfer learning, and transfer the invariant representations and detect returning threats at different hours or attack locations.

Considering the multivariate time series and spatial characteristics of false data injection in smart grid, this chapter has developed a spatiotemporal domain-adversarial training scheme to address the challenges of limited labeled attack samples and varying system loads and locations at the time of a second attack. To validate the effectiveness of the proposed method, this chapter selects the widely studied false data injection (FDI) attacks as the attack model and simulates data on IEEE-bus system with different system and attack variations. Compared with the state-of-the-art machine learning classifiers, the proposed solution has demonstrated effective accuracy improvements against returning attacks launched at different times and locations.

Chapter 6

Conclusions

The smart grid, empowered with the advancement of declining costs in communications, advanced sensors, and distributed computing technology, is connecting utilities and customers and providing efficiency, reliability and safety of power delivery. However, the growing number of interconnections among billions of cyber-physical devices creates complex interdependence and vulnerabilities that will inevitably raise the occurrence of cyber attacks in power systems. The influences of a cyber-attack could be grievous and disastrous.

Intrusion detection is of great importance for the smart grid. Among the state-of-the-art intrusion detection methods, machine learning, which undergoes the rigorous process of designing and implementing algorithms with expected performance, has acclaimed significant attention in the smart grid security research community. A rich line of ML approaches has significantly enhanced the cyber-physical security monitoring and situational awareness capacity [49, 50]. However, general machine learning approaches presume that the training and testing data are generated by identical independent distribution. This assumption may not hold in many real-world applications, especially for the Cyber-Physical Systems (CPS), since the system dynamics may alter the data distribution and thus fail the trained model. This poses a challenge to ML in the CPS scenarios where adaptive inference is required. In the power systems, the labeled real attack data is often rare and data distribution shift [112] may arise when system dynamics and attack patterns change. The trained model on limited data is brittle, and seemingly slight changes in the data distribution may lead to accuracy degradations. Such gaps call for effective and robust intrusion detection algorithms that could mitigate data distribution divergence and help to target advanced persistent threats.

In this work, we first propose a transfer learning scheme based on the domain-adversarial training framework [38] for intrusion detection against unknown threat variants. The scheme is capable of integrating different baseline machine learning classifiers to improve their detection accuracy against unknown threats/variants of different types and locations. Results have shown that all baseline classifiers can benefit significantly from the domain-adversarial training and demonstrate robust performance against unseen types and locations of threats. Discussions on the hyper-parameters have also been provided for deployment in practice.

Then we propose a self-adaptive intrusion detection scheme based on semi-supervised domain-adversarial training. The proposed scheme is capable of mapping data shifting distributions into a unified feature space to improve attack detection performance under dynamic change load demands. The results have shown that the proposed scheme can effectively tackle the rarity of attack samples and achieve robust performance against data distribution shifts than classic machine learning classifiers.

To answer when to apply transfer learning and take spatiotemporal information into consideration, we propose a spatiotemporal domain-adversarial training scheme to transfer the invariant representations and detect returning threats at different hours or/and attack locations. To validate the effectiveness of the proposed method, we select the widely studied false data injection (FDI) attack as the attack model and simulate data on IEEE-30 bus system with different system variations and attack variations. Compared with the state-of-the-art machine learning classifiers, the proposed solution has demonstrated effective detection accuracy improvements against returning attacks launched at different times or/and

Table 14: Comparison of proposed TL-based IDS schemes

IDS	Strengths	Limitations
Domain-Adversarial Transfer Learning	Improve the accuracy of detection threats with unseen types or locations.	Not consider the rarity of attack data.
Semi-Supervised Domain-Adversarial Training	Improve the accuracy of detection of the same threat with temporal variations.	Not consider the spatial variations of threats.
Spatiotemporal Domain-Adversarial Training	Improve the accuracy of detection of the same threat with spatiotemporal variations.	Require longer training time.

locations. The summary of proposed three methods is illustrated in 14.

For the future work, on one hand, we could extend to the current work to other smart grid systems and CPS scenarios such as IEEE 9-bus, IEEE 14-bus, IEEE 118-bus. [131, 132, 133], and we could further investigate larger scales and more detailed specifications of both cyber and physical systems, to extend the knowledge transfer ability among events like normal operations, system faults, and intentional attacks as well as scenarios of heterogeneous manufacturers, protocols, and networks. On the other hand, it will be interesting to investigate feature integration and fusion techniques to further improve the detection accuracy.

Bibliography

[1] M. M. H. Onik, C.-S. KIM, and J. Yang, “Personal data privacy challenges of the fourth industrial revolution,” 02 2019, pp. 635–638.

[2] Y. Zhang and J. Yan, “Domain-adversarial transfer learning for robust intrusion detection in the smart grid,” in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2019, pp. 1–6.

[3] Industrial control system (ICS) cyber attack datasets. https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets.

[4] Illinois Center for a Smarter Electric Grid (ICSEG). “IEEE 30-bus system". [Online]. Available: https://icseg.iti.illinois.edu/ieee-30-bus-system/

[5] “ISO New England - energy, load, and demand reports,” https://www.iso-ne.com/isoexpress/web/reports/load-and-demand/-/tree/dmnd-five-minute-sys, ISO New England, Tech. Rep., 2019.

[6] E. A. Lee, “Cyber physical systems: Design challenges,” in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*. IEEE, 2008, pp. 363–369.

[7] N. Y. Kim, S. Rathore, J. H. Ryu, J. H. Park, and J. H. Park, “A survey on cyber physical system security for IoT: Issues, challenges, threats, solutions.” *Journal of Information Processing Systems*, vol. 14, no. 6, 2018.

[8] H. He and J. Yan, “Cyber-physical attacks and defences in the smart grid: a survey,” *IET Cyber-Physical Systems: Theory & Applications*, vol. 1, no. 1, pp. 13–27, 2016.

[9] H. Farhangi, “The path of the smart grid,” *IEEE power and energy magazine*, vol. 8, no. 1, pp. 18–28, 2009.

[10] X. Fang, S. Misra, G. Xue, and D. Yang, “Smart grid—the new and improved power grid: A survey,” *IEEE communications surveys & tutorials*, vol. 14, no. 4, pp. 944–980, 2011.

[11] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "Smart grid technologies: Communication technologies and standards," *IEEE transactions on Industrial informatics*, vol. 7, no. 4, pp. 529–539, 2011.

[12] V. Terzija, G. Valverde, D. Cai, P. Regulski, V. Madani, J. Fitch, S. Skok, M. M. Begovic, and A. Phadke, "Wide-area monitoring, protection, and control of future electric power networks," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 80–93, Jan 2011.

[13] H. Sui, H. Wang, M. Lu, and W. Lee, "An AMI system for the deregulated electricity markets," *IEEE Transactions on Industry Applications*, vol. 45, no. 6, pp. 2104–2108, Nov 2009.

[14] M. LeMay, R. Nelli, G. Gross, and C. A. Gunter, "An integrated architecture for demand response communications and control," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, Jan 2008, pp. 174–174.

[15] M. Z. Alom and T. M. Taha, "Network intrusion detection for cyber security on neuromorphic computing system," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 3830–3837.

[16] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-based feature learning for cyber security applications," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 3854–3861.

[17] H. He and J. Yan, "Cyber-physical attacks and defences in the smart grid: a survey," *IET Cyber-Physical Systems: Theories & Applications*, vol. 1, no. 1, pp. 13–27, 2016.

[18] "Business blackout: The insurance implications of a cyber attack on the us power grid," Lloyd's and the University of Cambridge Centre for Risk Studies, Tech. Rep., 2015.

[19] "Cyber-attack against ukrainian critical infrastructure," https://www.iso-ne.com/isoexpress/web/reports/load-and-demand/-/tree/dmnd-five-minute-sys, The Industrial Control Systems Cyber Emergency Response Team (ICS-CERT), Tech. Rep., 2019.

[20] W. Wang and Z. Lu, "Cyber security in the smart grid: Survey and challenges," *Computer networks*, vol. 57, no. 5, pp. 1344–1371, 2013.

[21] H. Debar, M. Dacier, and A. Wespi, "A revised taxonomy for intrusion-detection systems," in *Annales des télécommunications*, vol. 55, no. 7-8. Springer, 2000, pp. 361–378.

[22] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[23] C. Kruegel and T. Toth, “Using decision trees to improve signature-based intrusion detection,” in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 173–191.

[24] F. Anjum, D. Subhadrabandhu, and S. Sarkar, “Signature based intrusion detection for wireless ad-hoc networks: A comparative study of various routing protocols,” in *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No. 03CH37484)*, vol. 3. IEEE, 2003, pp. 2152–2156.

[25] H. Wu, S. Schwab, and R. L. Peckham, “Signature based network intrusion detection system and method,” Sep. 9 2008, uS Patent 7,424,744.

[26] V. Kumar and O. P. Sangwan, “Signature based intrusion detection system using snort,” *International Journal of Computer Applications & Information Technology*, vol. 1, no. 3, pp. 35–41, 2012.

[27] W. Gao and T. H. Morris, “On cyber attacks and signature based intrusion detection for modbus based industrial control systems,” *Journal of Digital Forensics, Security and Law*, vol. 9, no. 1, p. 3, 2014.

[28] D. Yang, A. Usynin, and J. W. Hines, “Anomaly-based intrusion detection for scada systems,” in *5th intl. topical meeting on nuclear plant instrumentation, control and human machine interface technologies (npic&hmit 05)*, 2006, pp. 12–16.

[29] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” *computers & security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[30] M. Tavallaee, N. Stakhanova, and A. A. Ghorbani, “Toward credible evaluation of anomaly-based intrusion-detection methods,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 5, pp. 516–524, 2010.

[31] V. Jyothsna, V. R. Prasad, and K. M. Prasad, “A review of anomaly based intrusion detection systems,” *International Journal of Computer Applications*, vol. 28, no. 7, pp. 26–35, 2011.

[32] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, “Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model,” *Journal of Computational Science*, vol. 25, pp. 152–160, 2018.

[33] O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, “An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks,” *Expert systems with Applications*, vol. 29, no. 4, pp. 713–722, 2005.

[34] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, “A hybrid intrusion detection system design for computer network security,” *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, 2009.

[35] A. Herrero, E. Corchado, M. A. Pellicer, and A. Abraham, “Movih-ids: A mobile-visualization hybrid intrusion detection system,” *Neurocomputing*, vol. 72, no. 13-15, pp. 2775–2784, 2009.

[36] K. Yan, S. Wang, S. Wang, and C. Liu, “Hybrid intrusion detection system for enhancing the security of a cluster-based wireless sensor network,” in *2010 3rd international conference on computer science and information technology*, vol. 1. IEEE, 2010, pp. 114–118.

[37] H. Bostani and M. Sheikhan, “Hybrid of anomaly-based and specification-based IDS for internet of things using unsupervised opf based on mapreduce approach,” *Computer Communications*, vol. 98, pp. 52–71, 2017.

[38] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, Jan. 2016.

[39] Y. Zhang and J. Yan, “Semi-supervised domain-adversarial training for intrusion detection against false data injection in the smart grid,” in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–7.

[40] Y. Mo, R. Chabukswar, and B. Sinopoli, “Detecting integrity attacks on scada systems,” *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1396–1407, 2013.

[41] K. G. Vamvoudakis, J. P. Hespanha, B. Sinopoli, and Y. Mo, “Detection in adversarial environments,” *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3209–3223, 2014.

[42] R. Mitchell and R. Chen, “Behavior-rule based intrusion detection systems for safety critical smart grid applications,” *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1254–1263, 2013.

[43] C.-H. Lo and N. Ansari, “Consumer: A novel hybrid intrusion detection system for distribution networks in smart grid,” *IEEE Transactions on Emerging Topics in Computing*, vol. 1, no. 1, pp. 33–44, 2013.

[44] P. Jokar, H. Nicanfar, and V. C. Leung, “Specification-based intrusion detection for home area networks in smart grids,” in *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2011, pp. 208–213.

[45] M. A. Faisal, Z. Aung, J. R. Williams, and A. Sanchez, “Data-stream-based intrusion detection system for advanced metering infrastructure in smart grid: A feasibility study,” *IEEE Systems journal*, vol. 9, no. 1, pp. 31–44, 2014.

[46] H. Lin, A. Slagell, Z. T. Kalbarczyk, P. W. Sauer, and R. K. Iyer, "Runtime semantic security analysis to detect and mitigate control-related attacks in power grids," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 163–178, 2016.

[47] S. Sridhar and M. Govindarasu, "Model-based attack detection and mitigation for automatic generation control," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 580–591, 2014.

[48] G. Koutsandria, V. Muthukumar, M. Parvania, S. Peisert, C. McParland, and A. Scaglione, "A hybrid network IDS for protective digital relays in the power transmission grid," in *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Nov 2014, pp. 908–913.

[49] A. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.

[50] J. Yan, B. Tang, and H. He, "Detection of false data attacks in smart grid with supervised learning," in *2016 International Joint Conference on Neural Networks (IJCNN)*, July 2016, pp. 1395–1402.

[51] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander, and M. S. H. Sunny, "Application of big data and machine learning in smart grid, and associated security concerns: A review," *IEEE Access*, vol. 7, pp. 13 960–13 988, 2019.

[52] Y. Wang, M. M. Amin, J. Fu, and H. B. Moussa, "A novel data analytical approach for false data injection cyber-physical attack mitigation in smart grids," *IEEE Access*, vol. 5, pp. 26 022–26 033, 2017.

[53] A. Le, Y. Chen, K. K. Chai, A. Vasenev, and L. Montoya, "Incorporating fair into bayesian network for numerical assessment of loss event frequencies of smart grid cyber threats," *Mobile Networks and Applications*, vol. 24, no. 5, pp. 1713–1721, 2019.

[54] J. Pearl, "Bayesian networks," 2011.

[55] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[56] H. Jiang, J. J. Zhang, W. Gao, and Z. Wu, "Fault detection, identification, and location in smart grid based on data-driven computational methods," *IEEE Transactions on Smart Grid*, vol. 5, no. 6, pp. 2947–2956, 2014.

[57] A. M. Kosek, "Contextual anomaly detection for cyber-physical security in smart grids based on an artificial neural network model," in *2016 Joint Workshop on Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG)*. IEEE, 2016, pp. 1–6.

[58] L. Haghnegahdar and Y. Wang, "A whale optimization algorithm-trained artificial neural network for smart grid cyber intrusion detection," *Neural computing and applications*, vol. 32, no. 13, pp. 9427–9441, 2020.

[59] A. Shenfield, D. Day, and A. Ayesh, "Intelligent intrusion detection systems using artificial neural networks," *ICT Express*, vol. 4, no. 2, pp. 95–99, 2018.

[60] S. Kulkarni and G. Harman, *An elementary introduction to statistical learning theory*. John Wiley & Sons, 2011, vol. 853.

[61] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[62] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[63] M. Ozay, I. Esnaola, F. Yarman Vural, S. Kulkarni, and H. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1773–1786, Aug. 2016.

[64] K. Manandhar, X. Cao, F. Hu, and Y. Liu, "Detection of faults and attacks including false data injection attack in smart grid using kalman filter," *IEEE transactions on control of network systems*, vol. 1, no. 4, pp. 370–379, 2014.

[65] R. Borges Hink, J. Beaver, M. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in *2014 7th International Symposium on Resilient Control Systems (ISRCS)*, Aug. 2014, pp. 1–8.

[66] D. Wilson, Y. Tang, J. Yan, and Z. Lu, "Deep learning-aided cyber-attack detection in power transmission systems," in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, Aug. 2018, pp. 1–5.

[67] C. Hu, J. Yan, and C. Wang, "Advanced cyber-physical attack classification with extreme gradient boosting for smart transmission grids," in *2019 IEEE Power & Energy Society General Meeting (PESGM)*, in press.

[68] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[69] T. Croonenborghs, K. Driessens, and M. Bruynooghe, "Learning relational options for inductive transfer in relational reinforcement learning," in *International Conference on Inductive Logic Programming*. Springer, 2007, pp. 88–97.

[70] D. L. Silver and K. P. Bennett, "Guest editor's introduction: special issue on inductive transfer learning," *Machine Learning*, vol. 73, no. 3, p. 215, 2008.

[71] Z. Deng, K.-S. Choi, Y. Jiang, and S. Wang, "Generalized hidden-mapping ridge regression, knowledge-leveraged inductive transfer learning for neural networks, fuzzy systems and kernel methods," *IEEE transactions on cybernetics*, vol. 44, no. 12, pp. 2585–2599, 2014.

[72] J. Garcke and T. Vanck, "Importance weighted inductive transfer learning for regression," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 466–481.

[73] T. Scott, K. Ridgeway, and M. C. Mozer, "Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 76–85.

[74] A. Arnold, R. Nallapati, and W. W. Cohen, "A comparative study of methods for transductive transfer learning," in *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*. IEEE, 2007, pp. 77–82.

[75] B. Quanz and J. Huan, "Large margin transductive transfer learning," in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009, pp. 1327–1336.

[76] M. T. Bahadori, Y. Liu, and D. Zhang, "Learning with minimum supervision: A general framework for transductive transfer learning," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 61–70.

[77] L. Xie, Z. Deng, P. Xu, K.-S. Choi, and S. Wang, "Generalized hidden-mapping transductive transfer learning for recognition of epileptic electroencephalogram signals," *IEEE transactions on cybernetics*, vol. 49, no. 6, pp. 2200–2214, 2018.

[78] Q. Yao, H. Yang, A. Yu, and J. Zhang, "Transductive transfer learning-based spectrum optimization for resource reservation in seven-core elastic optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4164–4172, 2019.

[79] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012, pp. 17–36.

[80] B. Du, L. Zhang, D. Tao, and D. Zhang, "Unsupervised transfer learning for target detection from hyperspectral images," *Neurocomputing*, vol. 120, pp. 72–82, 2013.

[81] S. Li and Y. Fu, "Unsupervised transfer learning via low-rank coding for image clustering," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 1795–1802.

[82] H. Chang, J. Han, C. Zhong, A. M. Snijders, and J.-H. Mao, "Unsupervised transfer learning via multi-scale convolutional sparse coding for biomedical applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 5, pp. 1182–1194, 2017.

[83] Z. Fang, J. Lu, F. Liu, and G. Zhang, "Unsupervised domain adaptation with sphere retracting transformation," in *2019 International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–8.

[84] L. Li, H. He, J. Li, and G. Yang, "Adversarial domain adaptation via category transfer," in *2019 International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–8.

[85] L. Li and Z. Zhang, "Semi-supervised domain adaptation by covariance matching," *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[86] L. A. Pereira and R. da Silva Torres, "Semi-supervised transfer subspace for domain adaptation," *Pattern Recognition*, vol. 75, pp. 235–249, 2018.

[87] W. Wang, H. Wang, C. Zhang, and Y. Gao, "Fredholm multiple kernel learning for semi-supervised domain adaptation," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[88] D. Garcia-Romero and A. McCree, "Supervised domain adaptation for i-vector based speaker recognition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4047–4051.

[89] M. Abdelwahab and C. Busso, "Supervised domain adaptation for emotion recognition from speech," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5058–5062.

[90] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5715–5725.

[91] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *European conference on computer vision*. Springer, 2016, pp. 443–450.

[92] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *International conference on machine learning*. PMLR, 2017, pp. 2208–2217.

[93] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[94] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV '15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 4068–4076.

[95] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu, "Variational recurrent adversarial deep domain adaptation," 2016.

[96] G. Wilson, J. R. Doppa, and D. J. Cook, "Multi-source deep domain adaptation with weak supervision for time-series sensor data," *arXiv preprint arXiv:2005.10996*, 2020.

[97] K. Bartos and M. Sofka, "Robust representation for domain adaptation in network security," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2015, pp. 116–132.

[98] J. Zhao, S. Shetty, and J. Pan, "Feature-based transfer learning for network security," in *2017 IEEE Military Communications Conference (MILCOM)*, Oct. 2017, pp. 17–22.

[99] D. Nahmias, A. Cohen, N. Nissim, and Y. Elovici, "Trustsign: Trusted malware signature generation in private clouds using deep feature transfer learning," in *2019 International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–8.

[100] A. Mathew, J. Mathew, M. Govind, and A. Mooppan, "An improved transfer learning approach for intrusion detection," *Procedia computer science*, vol. 115, pp. 251–257, 2017.

[101] S. Tariq, S. Lee, and S. S. Woo, "Cantransfer: transfer learning based intrusion detection on a controller area network using convolutional lstm network," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 1048–1055.

[102] A. Singla, E. Bertino, and D. Verma, "Overcoming the lack of labeled data: training intrusion detection models using transfer learning," in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2019, pp. 69–74.

[103] A. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[104] Z. Cao, M. Long, J. Wang, and M. Jordan, "Partial transfer learning with selective adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[105] Z. Chu, J. Zhang, O. Kosut, and L. Sankar, "Unobservable false data injection attacks against pmus: Feasible conditions and multiplicative attacks," in *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, Oct 2018, pp. 1–6.

[106] M. Jamei, A. Scaglione, and S. Peisert, "Low-resolution fault localization using phasor measurement units with community detection," in *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, Oct 2018, pp. 1–6.

[107] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep 1995. [Online]. Available: https://doi.org/10.1023/A:1022627411411

[108] A. Liaw, M. Wiener *et al.*, "Classification and regression by random forest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[109] R. J. Lewis, "An introduction to classification and regression tree (CART) analysis," in *Annual meeting of the society for academic emergency medicine in San Francisco, California*, vol. 14, 2000.

[110] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[111] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[112] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence, *Dataset Shift in Machine Learning*, 2009.

[113] L. Brieman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[114] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 13:1–13:33, Jun. 2011.

[115] G. Liang, S. R. Weller, J. Zhao, F. Luo, and Z. Y. Dong, "The 2015 ukraine blackout: Implications for false data injection attacks," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3317–3318, 2016.

[116] A. Abur and A. Gomez-Exposito, *Power System State Estimation: Theory and Implementation*, 01 2004, vol. 24.

[117] Y. Sasaki, "The truth of the f-measure," *Teach Tutor Mater*, 01 2007.

[118] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997.

[119] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel method for the two-sample-problem," *Advances in neural information processing systems*, vol. 19, pp. 513–520, 2006.

[120] S. Rabanser, S. Günnemann, and Z. Lipton, "Failing loudly: An empirical study of methods for detecting dataset shift," in *Advances in Neural Information Processing Systems*, 2019, pp. 1396–1408.

[121] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 193–200.

[122] X. Cao, D. Wipf, F. Wen, G. Duan, and J. Sun, "A practical transfer learning algorithm for face verification," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3208–3215.

[123] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised representation learning with double encoding-layer autoencoder for transfer learning," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 9, no. 2, pp. 1–17, 2017.

[124] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International conference on machine learning*. PMLR, 2015, pp. 97–105.

[125] M. Baktashmotlagh, M. Harandi, and M. Salzmann, "Distribution-matching embedding for visual domain adaptation," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 3760–3789, 2016.

[126] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2272–2281.

[127] J. R. Hershey and P. A. Olsen, "Approximating the kullback leibler divergence between gaussian mixture models," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 4. IEEE, 2007, pp. IV–317.

[128] F. Pérez-Cruz, "Kullback-leibler divergence estimation of continuous distributions," in *2008 IEEE international symposium on information theory*. IEEE, 2008, pp. 1666–1670.

[129] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 1578–1585.

[130] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[131] Illinois Center for a Smarter Electric Grid (ICSEG). "IEEE 9-bus system". [Online]. Available: https://icseg.iti.illinois.edu/wscc-9-bus-system/

[132] Illinois Center for a Smarter Electric Grid (ICSEG). "IEEE 14-bus system". [Online]. Available: https://icseg.iti.illinois.edu/ieee-14-bus-system/

[133] Illinois Center for a Smarter Electric Grid (ICSEG). "IEEE 118-bus system". [Online]. Available: https://icseg.iti.illinois.edu/ieee-118-bus-system/