PREPARING SMART ENVIRONMENTS FOR LIFE IN THE WILD: FEATURE-SPACE AND MULTI-VIEW HETEROGENEOUS TRANSFER LEARNING

By

KYLE DILLON FEUZ

A dissertation submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY
Department of Electrical Engineering and Computer Science

May 2014

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of KYLE DILLON FEUZ find it satisfactory and recommend that it be accepted.

Diane J. Cook, Ph.D., Chair

Larry B. Holder, Ph.D.

Matthew E. Taylor, Ph.D.

Qiang Yang, Ph.D.

Acknowledgment

This dissertation could not have been possible were it not for the help of many individuals. First, I would like to thank my major professor, Dr. Diane Cook, whose help has been invaluable. She has put in many hours guiding me along the path as I delved into this research topic. Many times, I asked for feedback and guidance on a paper and she always responded quickly, with meaningful insight. It did not matter whether it was a weekday, a weeknight or a weekend, if there was a deadline approaching I could count on her to be there.

I would also like to thank my committee members, Dr. Larry Holder, Dr. Matthew Taylor, and Dr. Qiang Yang who each provided valuable contributions to this work. Their constructive criticism pushed me to achieve more. Dr. Aaron Crandall graciously agreed to fill in for Dr. Yang on the committee during the final defense and has been a wonderful person to work with over the last few years. The other students and faculty of the CASAS and IGERT programs have also been great people to work with. It is always nice to feel like you belong to a group and to share in that camaraderie.

Last and most importantly, I must acknowledge my family and my Heavenly Father. My wife has been there for me through thick and thin. In many ways, completing this dissertation has been harder on her than it has on me. My children provided the ultimate source of stress relief and helped me to see the joy in discovering life one day at a time. My parents have been my role models and their encouragement provided the extra boost I needed to complete this dissertation. Finally, the Lord has given me life itself and I owe all that I have and all that I am to Him.

PREPARING SMART ENVIRONMENTS FOR LIFE IN THE WILD: FEATURE-SPACE AND MULTI-VIEW HETEROGENEOUS TRANSFER LEARNING

Abstract

by Kyle Dillon Feuz, Ph.D.
Washington State University
May 2014

Chair: Diane J. Cook

With the ever-increasing abundance of sensing and computing devices embedded into our environments we have the opportunity to create personalized activity recognition ecosystems. Two key challenges must first be overcome, the new environment problem and the new sensing platform problem. The new environment problem is encountered every time a sensing platform is deployed to a new environment. The new sensing platform problem is encountered every time a new sensing platform is deployed into an environment with an existing sensing platform. We approach these problems as transfer learning problems with heterogeneous feature-spaces, referred to as heterogeneous transfer learning. We propose several novel algorithms for each setting. Additionally, some theoretical work on the accuracy bounds and the run-time of the algorithms is also presented.

Feature-Space Remapping (FSR) is proposed as a novel class of heterogeneous transfer learning algorithms which can be applied to the new environment problem. These algorithms are the first to perform heterogeneous transfer learning without requiring

explicit linkage data. We show how these algorithms are able to outperform learning a model generalized across different environments using relations between features as specified by a domain expert. We also show how FSR can be used in conjunction with ensemble learning to combine information from multiple datasets. This method outperforms the state-of-the-art by 10% to 20%.

Multi-view Transfer Learning is proposed as a solution to the new sensing platform problem. In multi-view transfer learning the same instance can be seen from multiple views or feature-spaces which facilitates transferring knowledge from one view to another. We develop several new multi-view learning algorithms for this problem. Using a well-trained view as a teacher, we show that the performance of new sensing platforms can be increased by as much as 20% through multi-view learning. The teacher can also be used to bootstrap a set of labeled training data for the new sensing platform which removes the need to manually annotate data when introducing new sensing platforms. We also provide bounds and an estimation of the learner's accuracy when the ground truth labeled data cannot be used to directly estimate the accuracy.

Contents

List of Tables

List of Figures

Dedication

This dissertation is dedicated to my wife and children

who remind me daily of the joy of life.

Chapter 1

Introduction

Learning and understanding observed activities is at the center of many fields of study. An individual's activities affect that individual, those around him, society, and the environment. The maturing of sensor and wireless network design has made it possible to automate activity recognition from sensor data. The number and diversity of projects that are utilizing activity recognition is exploding. Activity recognition is becoming an integral component of numerous solutions to real-world problems including health care, security surveillance, and context-aware services.

As the number of devices with sensing and computing capabilities increase a personalized activity recognition ecosystem may begin to emerge. Eventually, your smarthome, your smartphone, your smartvehicle and your smartoffice will all be working together to ensure your safety and satisfaction. However, if personalized activity recognition ecosystems are going to be successful, two interesting challenges must be overcome: the new environment problem and the new sensing platform problem. The new environment problem is encountered every time an activity recognition algorithm needs to be trained for a new environment or a new sensor layout. This situation occurs every time someone wants to install a new smart environment. The new sensing platform problem is encountered every time a new sensing platform is introduced. These two problems are not mutually exclusive as you could have a new sensing platform in a new environment.

Three approaches can be taken to solve these problems. First, labeled data for the

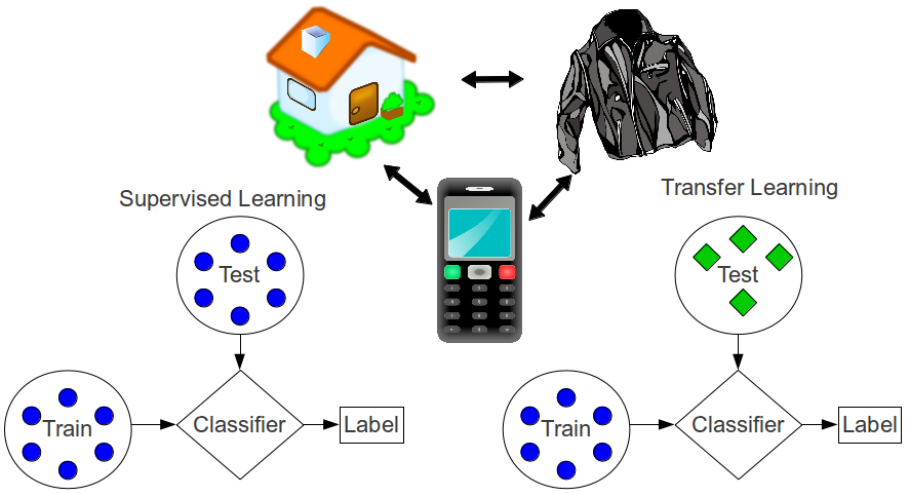


Figure 1: In traditional machine learning, the training and testing data come from the same domain and have similar distributions. In transfer learning, the training and testing data are related but no longer the same. In other words, transfer learning uses knowledge from a different but related problem to improve learning for the new problem. The home, phone and jacket represent three different sensor platforms that could be used in transfer learning.

new situation can be used to train a model for the specific situation. Second, labeled data from a variety of situations can be used to train a model which generalizes across the differences between the situations. Third, a new model, specific to the situation, can be adapted from one or more existing models. The first approach does not scale well due to the resource intensive nature of gathering labeled data. The second approach may also require large amounts of labeled data or in some situations may not even apply. For example, generalizing features across different sensor modalities may not be feasible. This leaves us with the third approach which is commonly referred to as transfer learning.

Transfer learning techniques have been proposed to specifically handle these types of situations. Transfer learning algorithms seek to apply knowledge learned from a previous task to a new, but related, task (See Figure 1). Heterogeneous transfer learning focuses

on transfer learning problems where the source and target domains are different because they have different feature spaces. The intuition behind transfer learning stems from the ability of humans to extend what has been learned in one context to a new context. In the field of machine learning, the benefits of transfer learning are numerous; less time is spent learning new tasks, less information is required of experts (usually human), and more situations can be handled effectively, making the learned model more robust. These potential benefits have led researchers to apply transfer learning techniques to many domains with varying degrees of success.

We propose several novel heterogeneous transfer learning algorithms which can be used to solve the new environment problem and the new sensing platform problem. We can divide these techniques into two different classes of transfer learning algorithms feature-space remapping (FSR) and multi-view transfer learning (MVTL). FSR reuses an existing trained classifier by remapping the target data onto the source feature-space. MVTL works by having data instances which are seen from multiple views or feature spaces. This co-occurrence data can then be used to transfer knowledge from one view to another. Most heterogeneous techniques require some form of linkage (co-occurrence data, dictionaries, or domain experts) between the source and target dataset. FSR, on the other hand, operates without any traditional linkage data. This makes it especially suitable for the new environment problem which is likely to have a different feature-space and to not have any instance-instance co-occurrence data. Multi-view transfer learning is not suitable for the new environment problem because it relies on instance-instance co-occurrence data. However, it is suitable for the new sensing platform problem when an existing sensing platform is already in the environment.

In conjunction with the goal of enabling the creation of personalized activity recognition ecosystems, we put forth the following hypothesis or objectives to be evaluated:

Objective 1.1 *Show the effectiveness of the FSR techniques in solving the new environment problem by comparing it against a model trained for the specific environment and by comparing it against a model which has been generalized for different environments.*

Objective 1.2 *Show the effectiveness of the FSR techniques utilizing multiple source domains in solving the new environment problem by comparing it against a model trained for the specific environment and by comparing it against a model which has been generalized for different environments.*

Objective 1.3 *Show the applicability of the FSR techniques to other domains by evaluating the performance of FSR for different document classification problems.*

Objective 1.4 *Show the effectiveness of the MVTL techniques in solving the new sensing platform problem when two sensing platforms are introduced simultaneously to the environment by comparing the MVTL techniques against models which have been trained using only labeled data available specifically to each sensing platform.*

Objective 1.5 *Show the effectiveness of the MVTL techniques in solving the new sensing platform problem when one sensing platform is introduced to an environment in which another sensing platform is already in place and recognizing activities by comparing the MVTL techniques against models which have been trained using only labeled data available specifically to each sensing platform.*

Objective 1.6 *Show the effectiveness of the MVTL techniques in solving the new sensing platform problem when three different sensing platforms are involved by comparing*

against the results of each MVTL technique when only two sensing platforms are involved.

Objective 1.7 *Show the applicability of the derived accuracy bounds by comparing the observed accuracies and agreements between each sensing platform and the expected, upper, and lower bounds of the accuracies of the sensing platforms.*

Meeting these objectives will be shown in Chapter 5 but we first review the applicable literature and describe our proposed approaches in detail. We present five FSR algorithms: Genetic Algorithms for Feature-Space Remapping (GAFSR), Greedy search for Feature-Space Remapping (GrFSR), Informed Similarity Feature-Space Remapping (ISFSR), and Uninformed Similarity Feature-Space Remapping (USFSR). We show how multiple source classifiers can be combined using Ensemble Learning with Feature-Space Remapping (ELFSR). We also present six MVTL algorithms: Co-Training (CoTrain), Co-Expectation Maximization (CoEM), Manifold Alignment (Man), Teacher-Learner (Teach), Personalized Ecosystem (PECO), and Personalized Ecosystem with Ensembles (PECO-E). The first four MVTL algorithms are extensions to existing multi-view learning techniques while the last two MVTL algorithms and all of the FSR algorithms are completely novel.

Chapter 2

Background

2.1 Introduction

Researchers in the artificial intelligence community have struggled for decades trying to build machines capable of matching or exceeding the mental capabilities of humans. One capability that continues to challenge researchers is designing systems which can leverage experience from previous tasks into improved performance in a new task which has not been encountered before. When the new task is drawn from a different population than the old, this is considered to be *transfer learning*. The benefits of transfer learning are numerous; less time is spent learning new tasks, less information is required of experts (usually human), and more situations can be handled effectively. These potential benefits have lead researchers to apply transfer learning techniques to many domains with varying degrees of success.

One particularly interesting domain for transfer learning is human activity recognition. The goal of human activity recognition is to be able to correctly classify the current activity a human or group of humans is performing given some set of data. Activity recognition is important to a variety of applications including health monitoring, automatic security surveillance, and home automation. As research in this area has progressed, an increasing number of researchers have started looking at ways transfer learning can be applied to reduce the training time and effort required to initialize

new activity recognition systems, to make the activity recognition systems more robust and versatile, and to effectively reuse the existing knowledge that has previously been generated.

With the recent explosion in the number of researchers and the amount of research being done on transfer learning, activity recognition, and transfer learning for activity recognition, it becomes increasingly important to critically analyze this body of work and discover areas which still require further investigation. Although recent progress in transfer learning has been analyzed in [78, 101, 112] and several surveys have been conducted on activity recognition [2, 4, 14, 41] no one has specifically looked into the intersection of these two areas. This chapter, therefore, examines the field of transfer-based activity recognition and the unique challenges presented in this domain. For an overview of the chapter, see Figure 2 which illustrates the topics covered in this chapter and how they relate to each other.

2.2 Activity Recognition

Activity recognition aims to identify activities as they occur based on data collected by sensors. There exist a number of approaches to activity recognition [47] that vary depending on the underlying sensor technologies that are used to monitor activities, the alternative machine learning algorithms that are used to model the activities and the realism of the testing environment.

Advances in pervasive computing and sensor networks have resulted in the development of a wide variety of sensor modalities that are useful for gathering information about human activities. Wearable sensors such as accelerometers are commonly used

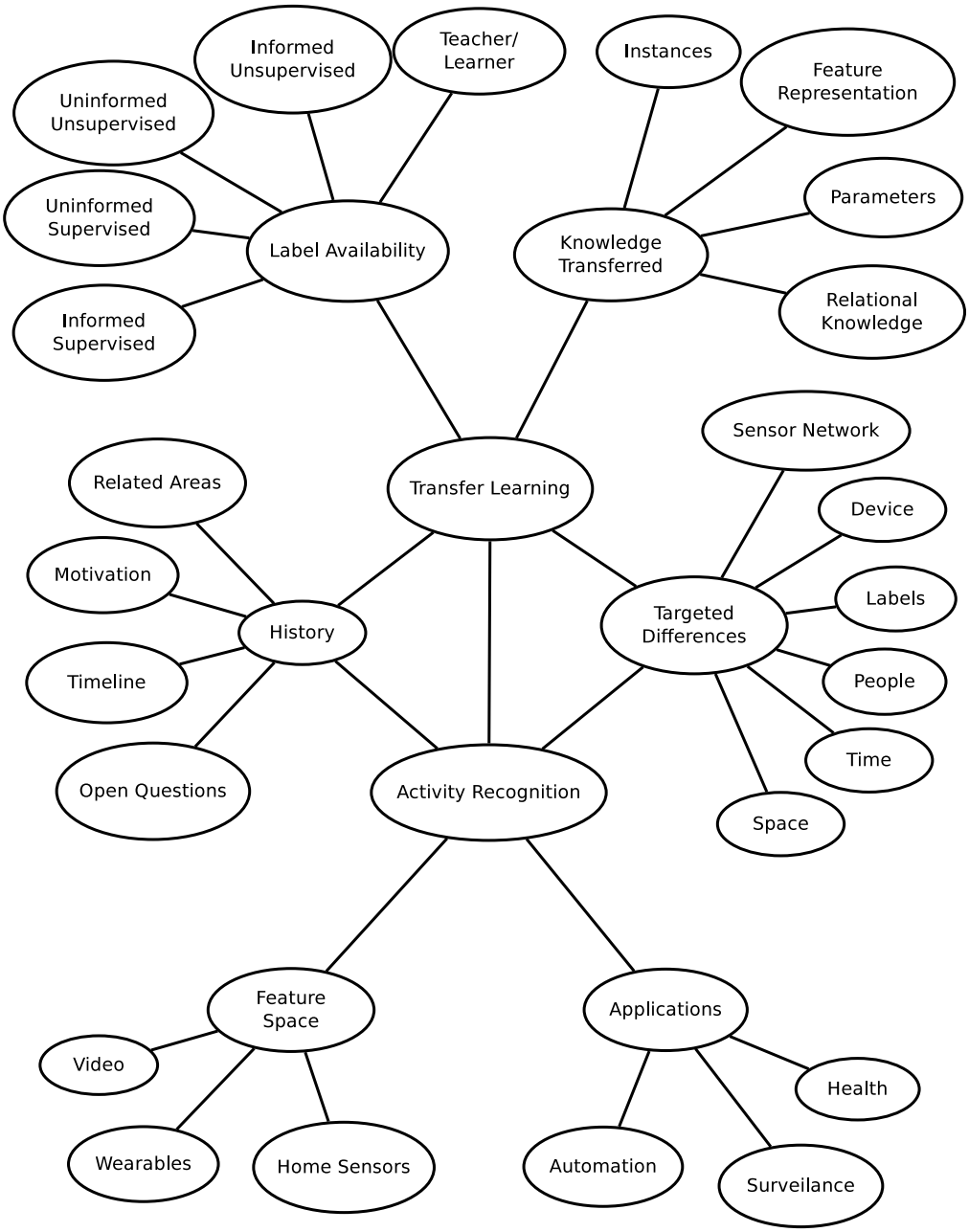


Figure 2: Content map of the transfer learning for activity recognition domain.

for recognizing ambulatory movements (e.g., walking, running, sitting, climbing, and falling) [52, 63]. More recently, researchers are exploring smart phones equipped with accelerometers and gyroscopes to recognize such movement and gesture patterns [54].

Environment sensors such as infrared motion detectors or magnetic door sensors have been used to gather information about more complex activities such as cooking, sleeping, and eating. These sensors are adept in performing location-based activity recognition in indoor environments [1, 62, 108] just as GPS is used for outdoor environments [59]. Some activities such as washing dishes, taking medicine, and using the phone are characterized by interacting with unique objects. In response, researchers have explored the usage of RFID tags and shimmer sensors for tagging these objects and using the data for activity recognition [71, 81]. Researchers have also used data from video cameras and microphones as well [1].

There have been many varied machine learning models that have been used for activity recognition. These can be broadly categorized into template matching / transductive techniques, generative, and discriminative approaches. Template matching techniques employ a kNN classifier based on Euclidean distance or dynamic time warping. Generative approaches such as naïve Bayes classifiers where activity samples are modeled using Gaussian mixtures have yielded promising results for batch learning. Generative probabilistic graphical models such as hidden Markov models and dynamic Bayesian networks have been used to model activity sequences and to smooth recognition results of an ensemble classifier [58]. Decision trees as well as bagging and boosting methods have been tested [63]. Discriminative approaches, including support vector machines and conditional random fields, have also been effective [15, 108] and unsupervised discovery and recognition methods have also been introduced [39, 92].

Along with a wide variety of machine-learning algorithms which have been applied to activity recognition there are also several different feature representation that are used. Some researchers have focused on pre-segmented activities [23, 59, 99, 115] while others perform activity recognition on unsegmented activity streams [22, 47]. For pre-segmented activities, features can be computed on the whole activity segment while for activity streams features are usually computed based upon a sliding activity window. In this work, we focus on two different feature representations both of which operate on activity streams rather than pre-segmented activities. The first feature representation is for event-based activity streams while the second feature representation is for continuous activity streams. The event-based feature representation uses fixed-length window based on the number of sensor events. The time of the sensor events as well as the frequency of each sensor are computed as features. The continuous activity stream feature representation uses a fixed-length window based upon the amount of time passed. The average values of the sensors are computed as features.

2.2.1 Event-Based Features

For the event-based feature representation, we formulate the learning problem as that of mapping a sequence consisting of the most recent sensor events within a sliding window of length k to a label representing the activity to the last (most recent) event in the sequence. The sensor events preceding the last event define the context for this last event. Data collected in a smart home consists of events generated by the sensors. These are stored as a 4-tuple: (Date, Time, Sensor Id, Message) as shown in Table 1.

To perform activity recognition, we extract features from data point i, where the

Table 1: Sample of Sensor Events

Date	**Time**	**Sensor**	**Value**
2011-06-15	03:41:50.30088	M021	OFF
2011-06-15	03:41:50.402649	MA020	OFF
2011-06-15	03:44:50.862962	M021	ON
2011-06-15	03:44:51.929508	M021	OFF
2011-06-15	04:41:28.179357	M021	ON
2011-06-15	04:41:29.333803	M021	OFF
2011-06-15	05:33:44.024833	M021	ON
2011-06-15	05:33:45.118382	M021	OFF
2011-06-15	06:33:30.363675	M021	ON
2011-06-15	06:33:31.437863	M021	OFF
2011-06-15	06:33:33.878588	M021	ON
2011-06-15	06:33:35.956492	M021	OFF
2011-06-15	08:45:45.685723	M021	ON
2011-06-15	08:45:46.789252	M021	OFF
2011-06-15	08:46:03.646237	M021	ON
2011-06-15	08:46:03.817155	MA020	ON
2011-06-15	08:46:08.513192	M021	OFF
2011-06-15	08:46:08.712314	MA020	OFF
2011-06-15	08:46:09.87972	MA020	ON
2011-06-15	08:46:12.103082	MA020	OFF
2011-06-15	08:46:21.859339	MA020	ON
2011-06-15	08:46:22.752142	M021	ON
2011-06-15	08:46:23.885996	M021	OFF
2011-06-15	08:46:25.199775	MA020	OFF
2011-06-15	08:46:26.713111	MA020	ON
2011-06-15	08:46:27.590115	M019	ON
2011-06-15	08:46:29.876241	MA020	OFF
2011-06-15	08:46:30.760636	M019	OFF
2011-06-15	08:46:32.587806	M018	ON
2011-06-15	08:46:36.329587	MA013	ON
2011-06-15	08:46:37.117772	M018	OFF
2011-06-15	08:46:45.86861	MA013	OFF

Table 2: The feature vector describing a data point under the first feature representation.

Feature #	**Value**
1	Time of day of the latest sensor event in the sliding window
2	Day of week of the latest sensor event in the sliding window
3 to $n+3$	Number of occurrences of each sensor in within the current window (n sensors)

data point corresponds to a sensor event sequence of length k. The vector x_i includes values for the features summarized in Table 2. Each y_i corresponds to the activity label that is associated with the last sensor event in the sequence. A collection of data points x_i and the corresponding labels y_i are fed as training data to a classifier to learn the activity models in a discriminative manner. The classifier thus learns a mapping from the sensor event sequence to the corresponding activity label.

2.2.2 Continuous Features

For the event-based feature representation, we formulate the learning problem as that of mapping feature values within a sliding window of time-duration k to a label representing the most prevalent activity during that time period. Data collected in a smart home consists of events generated by the sensors. The event-based sensor data can be converted to continuously sampled data by maintaining a current state for each sensor and sampling the state values at the desired frequency.

To perform activity recognition, we extract features from data point i, where the data point corresponds to a time period of length k. The vector x_i includes the average value of each sensor over the given time period (OFF=0 and ON=1). Each y_i corresponds to the most prevalent activity label during that time period. A collection of data points x_i and the corresponding labels y_i are fed as training data to a classifier to learn the

activity models in a discriminative manner. The classifier thus learns a mapping from sensor values over a given time-period to the corresponding activity label.

2.2.3 Classification Algorithms

Using either of the previously describe feature representations any number of different supervised classification algorithms could be used to perform the activity recognition. We will focus on two such classification algorithms, a Naïve Bayes classifier and a Decision Tree classifier.

A naïve Bayes classifier can learn a hypothesis by estimating $P(y)$ and $P(x|y)9;/$" based upon their observed frequencies and applying Bayes rule to estimate the posterior probability $P(y|x_i)$. The class y with the highest posterior probability is selected as the class label for q [68].

A decision tree classifier generates a tree where each interior node queries the value of an attribute and each leaf represents a value of the target variable (in this case, the activity label). Decision tree nodes are selected that maximize the reduction in entropy of the training data set [86].

2.2.4 Summary

In summary, researchers have investigated many of the various challenges involved in creating a personal activity recognition ecosystem, yet their efforts have remained largely disjoint and limited to a single class of sensors [17, 84]. Each of these research groups are making valuable contributions and a personal activity recognition ecosystem could not exist without these basic building blocks. However, we need to start finding ways

to successfully integrate these heterogeneous systems. The traditional approaches to activity recognition make the strong assumption that the training and test data are drawn from identical distributions. Many real-world applications cannot be represented in this setting and thus the baseline activity recognition approaches have to be modified to work in these realistic settings. Transfer based activity recognition is one conduit for achieving this.

2.3 Transfer Learning

The ability to identify deep, subtle connections, what we term *transfer learning*, is the hallmark of human intelligence. Byrnes [11] defines transfer learning as the ability to extend what has been learned in one context to new contexts. Thorndike and Woodworth [103] first coined this term as they explored how individuals transfer learned concepts between contexts that share common features. Barnett and Ceci provide a taxonomy of features that influence transfer learning in humans [5].

In the field of machine learning, transfer learning is studied under a variety of different names including learning to learn, life-long learning, knowledge transfer, inductive transfer, context-sensitive learning, and meta-learning [3, 34, 104, 105, 112]. It is also closely related to several other areas of machine learning such as self-taught learning, multi-task learning, domain adaptation, and co-variate shift. Because of this broad variance in the terms used to describe transfer learning it is helpful to provide a formal definition of transfer learning terms and of transfer learning itself which will be used throughout the rest of this work.

2.3.1 Definitions

We start with a review of basic definitions needed for discussions of transfer learning as it can be applied to activity recognition. Definitions for domain and task have been provided by Pan and Yang [78]:

Definition 2.1 (Domain) *A domain D is a two-tuple $(\chi, P(X))$. χ is the feature space of D and $P(X)$ is the marginal distribution where $X = \{x_1, ..., x_n\} \in \chi$.*

Definition 2.2 (Task) *A task T is a two-tuple $(Y, f())$ for some given domain D. Y is the label space of D and $f()$ is an objective predictive function for D. $f()$ is sometimes written as a conditional probability distribution $P(y|x)$. $f()$ is not given, but can be learned from the training data.*

To illustrate these definitions, consider the problem of activity recognition using motion sensors. The domain is defined by a feature space which may represent the n-dimensional space defined by n sensor firing counts within a given time window and a marginal probability distribution over all possible firing counts. The task is composed of a label space y which consists of the set of labels for activities of interest, and a conditional probability distribution consisting of the probability of assigning a label $y_i \in y$ given the observed instance $x \in \chi$.

Using these terms, we can now define transfer learning. In this chapter, we specify a definition of transfer learning that is similar to that presented by Pan and Yang [78] but we allow for transfer learning which uses multiple source domains.

Definition 2.3 (Transfer Learning) *Given a set of source domains $DS = D_{s_1}, ..., D_{s_n}$ where $n > 0$, a target domain, D_t, a set of source tasks $TS = T_{s_1}, ...T_{s_n}$ where $T_{s_i} \in TS$*

corresponds with $D_{s_i} \in DS$, and a target task T_t which corresponds to D_t, transfer learning improves the learning of the target predictive function $f_t()$ in D_t where $D_t \notin DS$ and $T_t \notin TS$.

This definition of transfer learning is broad and encompasses a large number of different transfer learning scenarios. The source domains can differ from the target domain by having a different feature space, a different distribution of instances in the feature space, or both. The source tasks can differ from the target task by having a different label space, a different predictive function for labels in that label space, or both. The source data can differ from the target data by having a different domain, a different task, or both. However, all transfer learning problems rely on the basic assumption that there exists some relationship between the source and target areas which allows for the successful transfer of knowledge from the source to the target.

2.3.2 Scenarios

To further illustrate the variety of problems which fall under the scope of transfer-based activity recognition, we provide illustrative scenarios. Not all of these scenarios can be addressed by current transfer learning methods. The first scenario represents a typical transfer learning problem solvable using recently developed techniques. The second scenario represents a more challenging situation that pushes the boundaries of current transfer learning techniques. The third scenario requires a transfer of knowledge across such a large difference between source and target datasets, that current techniques only scratch the surface of what is required to make such a knowledge transfer successful.

Scenario 1

In one home which has been equipped with multiple motion and temperature sensors, an activity recognition algorithm has been trained using months of annotated labels to provide the ground truth for activities which occur in that home. A transfer learning algorithm should be able to reuse the labeled data to perform activity recognition in a new setting. Such transfer will save months of man-hours annotating data for the new home. However, the new home has a different layout as well as a different resident and different sensor locations than the first home.

Scenario 2

An individual with Parkinson's disease visits his neurosurgeon twice a year to get an updated assessment of his gait, tremor, and cognitive health. The medical staff perform some gait measurements and simulated activities in their office space to determine the effectiveness of the prescribed medication, but want to determine if the observed improvement is reflected in the activities the patient performs in his own home. A learning algorithm will need to be able to transfer information between different physical settings, as well as time of day, sensors used, and scope of the activities.

Scenario 3

A researcher is interested in studying the cooking activity patterns of college students living in university dorms in the United States. The research study has to be conducted using the smart phone of the student as the sensing mechanism. The cooking activity of these students typically consists of heating up a frozen snack from the refrigerator in the microwave oven. In order to build the machine learning models for recognizing

these activity patterns, the researcher has access to cooking activities for a group of grandmothers living in India. This dataset was collected using smart home environmental sensors embedded in the kitchen and the cooking activity itself was very elaborate. Thus the learning algorithm is now faced with changes in the data at many layers; namely, differences in the sensing mechanisms, cultural changes, age related differences, different location settings and finally differences in the activity labels. This transfer learning from one setting to another diverse setting is most challenging and requires significant progress in transfer learning domain to even attempt to solve the problem.

These scenarios illustrate different types of transfer that should be possible using machine learning methods for activity recognition. As is described by these situations, transfer may occur across several dimensions. We next take a closer look at these types of transfer and use these descriptors to characterize existing approaches to transfer learning for activity recognition.

2.4 Dimensions of Analysis

Transfer learning can take many forms in the context of activity recognition. In this discussion we consider four dimensions to characterize various approaches to transfer learning for activity recognition. First, we consider different sensor modalities on which transfer learning has been applied. Second, we consider differences between the source and target environments in which data is captured. The third dimension is the amount and type of data labeling that is available in source and target domains. Finally, we examine the representation of the knowledge that is transferred from source to target.

2.4.1 Modality

One natural method for the classification of transfer learning techniques is the underlying sensing modalities used for activity recognition. Some techniques may be generalizable to different sensor modalities, but most techniques are too specific to be generally applicable to any sensor modality other than that for which they are designed to work with. This is usually because the types of differences that occur between source and target domains are different for each sensor modality. These differences and their effect on the transfer learning technique are discussed in detail in Section 2.4.2. In this section we consider only those techniques which have empirically demonstrated their ability to operate on a given sensor modality.

The classification of sensor modalities itself is a difficult problem and indeed creating precise classification topology is outside of the scope of this work. However, we roughly categorize sensor modalities into the following classifications, video cameras, wearable devices, and ambient sensors. For each sensor modality, we provide a brief description of the types of sensors which are included and a summary of the research works performing transfer learning in that domain. In this section, we do not describe the transfer learning algorithms used in the papers as that will be discussed in the other dimensions of analysis.

Video Sequences

Video cameras are one of the first sensor modalities in which transfer learning has been applied to the area of activity recognition [121]. Video cameras provide a dense feature-space for activity recognition which potentially allows for extremely fine-grained recognition of activities. Spatio-temporal features are extracted from video sequences for

characterizing the activities occurring in them. Activity models are then learned using these feature representations.

One drawback of video processing for activity recognition is that the use of video cameras raises more issues associated with user privacy. In addition, cameras need to be well positioned and track individuals in order to capture salient data for processing. Activity recognition via video cameras has received broad attention in transfer learning research [33, 35, 56, 60, 69, 117, 119, 120, 121, 124].

Wearable sensors

Body Sensor Networks are another commonly used sensing mechanism to capture activity related information from individuals. These sensors are typically worn by the individuals. Strategic placement of the sensors helps in capturing important activity related information such as movements of the upper and lower parts of the body that can then be used to learn activity models. Sensors in this category include, inertial sensors such as accelerometers and gyroscopes, sensors embedded in smart phones, radio frequency identification sensors and tags. Researchers have applied transfer learning techniques to both activity recognition using wearable accelerometers and activity recognition using smartphones but we have not seen any transfer learning approaches applied to activity recognition using RFID tags. This may be due in part to the relatively low use of RFID tags in activity recognition itself.

Within wearable sensors, two types of problems are generally considered. The first is the problem of activity recognition itself [6, 12, 19, 40, 51, 53, 95, 111, 126, 127], and the second is the problem of user localization, which can then be used to increase the accuracy of the activity recognition algorithm [73, 76, 77, 79, 129] . Both problems

present interesting challenges for transfer learning.

Ambient Sensors

Ambient sensors represent the broadest classification of sensor modalities which we define in this chapter. We categorize any sensor that is neither wearable nor video camera into ambient sensors. These sensors are typically embedded in an individual's environment. This category includes a wide variety of sensors such as motion detectors, door sensors, object sensors, pressure sensors, and temperature sensors. As the name indicates, these sensors collect a variety of activity related information such as human movements in the environment induced by activities, interactions with objects during the performance of an activity, and changes to illumination, pressure and temperature in the environment due to activities. Researchers have only recently begun to look at transfer learning applications for ambient sensors with the earliest work appearing around 2008 [107]. Since then the field of transfer learning for activity recognition using ambient sensors has progressed rapidly with many different research groups analyzing the problem from several different angles [20, 44, 88, 89, 90, 91, 95, 109, 128].

Crossing the sensor boundaries

Clearly, transfer learning within individual sensor modalities is progressing. Researchers are actively developing and applying new techniques to solve a variety of problems within any given sensor modality domain. However, there has been little work done that tries to transfer knowledge between any two or more sensor modalities. Kurz et al. [53] and Roggen et al. [95] address this problem using a teacher/learner model which is discussed further in Section 2.4.3. Hu et al. [44] introduce a transfer learning technique for

successfully transferring some knowledge across sensor modalities, but greater transfer of knowledge between modalities has yet to be explored.

2.4.2 Physical Setting Differences

Another useful categorization of transfer learning techniques is the types of physical differences between a source and target dataset across which the transfer learning techniques can achieve a successful transfer of knowledge. In this section, we describe these differences in a formal setting and provide illustrative examples drawn from activity recognition.

We use the terminology for domain, task and transfer learning defined in Section 2.3.1 to describe the differences between source and target datasets. These differences can be in the form of the feature-space representation, the marginal probability distribution of the instances, the label space, and/or the objective predictive function. When describing transfer learning in general, using such broad terms allows one to encompass many different problems. However, when describing transfer learning for a specific application, such as activity recognition, it is convenient to use more application specific terms. For example, differences in the feature-space representation can be thought of in terms of the sensor modalities and sampling rates and differences in the marginal probability distribution can be thought of in terms of different people performing the same activity, or having the activity performed in different physical spaces.

Even when limiting the scope to activity recognition, it is still infeasible to enumerate every possible difference between source and target datasets. In this chapter, we consider some of the most common or important differences between the source and

Table 3: Relationship between formally defined transfer learning differences and the applied meaning for activity recognition.

Formal Definition	**Applied Meaning**
$\chi_t \neq \chi_{s_i}$ for $0 < i < n$	sensor networks, sensor modality, or physical space
$P(X_t) \neq P(X_{s_i})$ for $0 < i < n$	time, people, devices, or sampling rates
$Y_t \neq Y_{s_i}$ for $0 < i < n$	activities or labels
$f_t(x) \neq f_{s_i}(x)$ for $0 < i < n$	time, people, devices, sampling rates, activities, or labels

target datasets including time, people, devices, space, sensor types, and labels. Table 3 summarizes the relationship between each of these applied differences and the formal definitions of transfer learning differences.

Differences across time, people, devices, or sensor sampling rates result in differences in the underlying marginal probability distribution, the objective predictive function, or both. Several papers focus specifically on transferring across time differences [51, 73, 76, 111], differences between people [16, 40, 88, 127], and differences between devices [126, 129].

Differences created when comparing datasets from different spaces or spatial layouts are reflected by differences in the feature-spaces, the marginal probability distributions, the objective predictive functions, or any combination of these. As the number of differences increases, the source and target datasets become less related making transfer learning more difficult. Because of this, current research usually imposes limiting assumptions about what is different between the spaces. Several researchers, for example, assume that some meta-features are added which provide space-independent information [20, 89, 90, 91, 107, 109]. For WiFi localization, Pan et al. [77] assume that the source and target spaces are in the same building. Applying transfer learning to video

clips from different spaces usually results in handling issues of background differences [13, 120, 121] and/or issues of camera view angle [60].

Differences in the labels used in the datasets are obviously reflected by differences in the label space and the objective predictive function. Compared to the other differences discussed previously, transferring between differences in the label space has received much less attention in the current literature [44, 56, 117, 124, 128].

One of the largest differences between datasets occurs when the source and target datasets have a different sensor modality. This makes the transfer learning problem much more difficult and relatively little work has been done in this direction. Hu and Yang have started work in this direction in [44]. Additionally, Calatroni et al. [12], Kurz et al. [53] and Roggen et al. [95] take a different approach to transferring across sensor modality by assuming a classifier for the source modalities can act as an expert for training a classifier in the target sensor modality.

2.4.3 Data Labeling

In this section we consider the problem of transfer learning from the perspective of the availability of labeled data. Traditional machine learning uses the terms *supervised learning* and *unsupervised learning* to distinguish learning techniques based on the availability and use of labeled data. To distinguish between source and target labeled data availability we introduce two new terms, *informed* and *uninformed*, which we apply to the availability of labeled data in the target area. Thus, informed supervised (IS) transfer learning implies that some labeled data is available in both the target and source domains. Uninformed supervised (US) transfer learning implies that labeled

data is available only in the source domain. Informed unsupervised (IU) transfer learning implies that labeled data is only available in the target domain. Finally, uninformed unsupervised (UU) transfer learning implies that no labeled data is available for either the source or target domains. One final case to consider is teacher/learner (TL) transfer learning, where no training data is directly available. Instead a previously-trained classifier (the Teacher) is introduced which operates simultaneously with the new classifier to be trained (the Learner) and provides the labels for observed data instances.

Two other terms that are often used in machine learning literature and may be applicable here are *inductive* and *transductive* learning. Inductive learning refers to learning techniques which try to learn the objective predictive function. Transductive learning techniques, on the other hand, try to learn the relationship between instances. Pan and Yang [78] extend the definitions of inductive and transductive learning to transfer learning, but the definitions do not create a complete taxonomy for transfer learning techniques. For this reason, we do not specifically classify recent works as being inductive or transductive in nature, but we note here how the inductive and transductive definitions fit into a classification based upon the availability of labeled data.

Inductive learning requires that labeled data be available in the target domain regardless of its availability in the source domain. Thus, most informed supervised and informed unsupervised transfer learning techniques are also inductive transfer learning techniques. Transductive learning, however, does not require labeled data in the target domain. Therefore, most uninformed supervised techniques are also transductive transfer learning techniques. Table 4 summarizes this general relationship.

Several researchers have developed and applied informed, supervised transfer learning techniques for activity recognition. These techniques have been applied to activity

Table 4: General relationship between inductive/transductive learning and the availability of labeled data.

Label Availability	**Most Common Approach**
Informed Supervised	Inductive Learning
Informed Unsupervised	Inductive Learning
Uninformed Supervised	Transductive Learning
Uninformed Unsupervised	Unsupervised Learning

recognition using wearables [6, 50, 72, 76, 111, 129] and to activity recognition using cameras [33, 56, 69, 120, 121, 124].

Research into transfer-based activity recognition using ambient sensors has almost exclusively focused on uninformed supervised transfer learning [20, 44, 45, 88, 107, 109, 111, 128], but a few algorithms are able to take advantage of the labeled target data if it is available [89, 90, 91]. This focus on uninformed supervised transfer learning is most likely due to the allurement of building an activity recognition framework that can be trained offline and later installed into any user's space without requiring additional data labeling effort. Wearables have also been used for uninformed supervised transfer learning research [40, 73, 77, 79, 122, 126, 127] as have cameras[13, 35, 60, 117, 119].

Despite the abundance of research using labeled source data, research into transfer learning techniques for activity recognition in which no source labels are available is extremely sparse. Pan et al. [73] have applied an uninformed unsupervised technique, transfer component analysis (TCA) to reduce the distance between domains by learning some transfer components across domains in a reproducing kernel Hilbert space using maximum mean discrepancy. We are unaware of any other work for uninformed unsupervised transfer-based activity recognition. We are also unaware of any work on informed unsupervised transfer-based activity recognition. The lack of research into

informed unsupervised transfer-based activity recognition is not surprising because the idea of having labeled target data available and not having labeled source data is counter-intuitive to the general principle of transfer learning. However, informed unsupervised transfer learning may still provide significant benefits to activity recognition.

The teacher/learner model for activity recognition is considerably less studied than the previously discussed techniques. However, we feel that this area has significant promise for improving transfer learning for activity recognition and making activity recognition systems much more robust and versatile. Roggen et al. [95], Kurz et al. [53], and Calatroni et al. [12] apply the teacher/learner model to develop an opportunistic system which is capable of using whatever sensors are currently contained in the environment to perform activity recognition.

In order for the teacher/learner model to be applicable, two requirements must be met. First, an existing classifier (the teacher) must already be trained in the source domain. Second, the teacher must operate simultaneously with a new classifier in the target domain (the learner) to provide the training for the learner. For example, Roggen et al. [95] equip a cabinet of drawers with an accelerometer for each drawer and then a classifier is trained to recognize which drawer of the cabinet is being opened or closed. This classifier becomes the teacher. Then several wearable accelerometers are attached to the person opening and closing the drawers. Now, a new classifier is trained using the wearable accelerometers. This classifier is the learner. When the individual opens or closes a drawer, the teacher labels the activity according to its classification model. This label is given to the learner which can then be used as labeled training data in real-time without the need to supply any manually labeled data.

The teacher/learner model presents a new perspective on transfer learning and introduces additional challenges. One major challenge of the teacher/learner model is that the accuracy of the learner is limited by the accuracy of the teacher. Additionally, the system's only source of a ground truth comes from the teacher and thus the learner is completely reliant upon the teacher. It remains to be explored whether the learner can ever outperform the teacher and if it does so, whether it can convince itself and others of this superior performance. Finally, while the teacher/learner model provides a convenient way to transfer across different domains, an additional transfer mechanism would need to be employed to transfer across different label spaces.

2.4.4 Type of Knowledge Transferred

Pan and Yang [78] describe four general classifications for transfer learning in relation to what is transferred, instance transfer, feature-representation transfer, parameter transfer, and relational-knowledge transfer.

Instance Transfer

Instance transfer reuses the source data to train the target classifier, usually by re-weighting the source instances based upon a given metric. Instance transfer techniques work well when $\chi_s = \chi_t$ i.e., the feature space describing the source and target domains are the same. They may also be applied after the feature representation has first been transferred to a common representation between the source and target domains.

Several researchers have applied instance transfer techniques to activity recognition. Hachiya et al. [40] develop an importance weighted least-squares probabilistic classification approach to handle transfer learning when $P(X_s) \neq P(X_t)$ (i.e., the co-variate

shift problem) and apply this approach to wearable accelerometers. Venkatesan et al. [50, 110, 111] extend the AdaBoost framework proposed by Freund and Schapire [36] to include cost-sensitive boosting which tries to weight samples from the source domain according to their relevance in the target domain. In their approach, samples from the source domain are first given a relevance cost. As the classifier is trained, those instances from the source domain with a high relevance must also be classified correctly. Xian-ming and Shao-zi apply TrAdaBoost (a different transfer learning extension of AdaBoost) [26] to action recognition in video clips [120] . Lam et al. weight the source and target data differently when training an SVM to recognize target actions from video clips [56]. Training a typical SVM involves solving the following optimization problem:

$$\begin{aligned} \min_{\vec{w},\xi}\{\frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}\xi_i\} \\ \text{s.t. } y_i(\vec{x_i}\cdot\vec{w}+b) - 1 + \xi_i \geq 0, \xi_i \geq 0 \end{aligned} \tag{2.1}$$

where $\vec{x_i}$ is the i^{th} datapoint and y_i, ξ_i are the label and slack variable associated with $\vec{x_i}$. $\vec{w}$ is the normal to the hyperplane. C is the parameter that trades off between training accuracy and margin size. However, to allow for the different source and target weights, they solve the following optimization:

$$\begin{aligned} \min_{\vec{w},\xi}\{\frac{1}{2}||\vec{w}||^2 + C_s\sum_{i=1}^{n}\xi_i + C_t\sum_{i=n+1}^{n+m}\xi_i\} \\ \text{s.t. } y_i(\vec{x_i}\cdot\vec{w}+b) - 1 + \xi_i \geq 0, \xi_i \geq 0 \end{aligned} \tag{2.2}$$

where the parameters are the same as before except the first n datapoints are from the source data and the last m datapoints are from the target data.

Unlike the previous instance-based approaches which weight the source instances based on similarity of features between the source and target data, Zheng et al. [128] use an instance-based approach to weight source instances based upon the similarity between the label information of the source and target data. This allows them to transfer the labels from instances in the source domain to instances in the target domain using web-knowledge to relate the two domains [44, 45]. Taking a different approach, several researchers [12, 53, 95] use the real-time teacher/learner model discussed in the previous section to transfer the label of the current instance in the source domain to the instance in the target domain.

Feature-Representation Transfer

Feature-representation transfer reduces the differences between the source and target feature spaces. This can be accomplished by mapping the source feature space to the target feature space such as $f : \chi_s \rightarrow \chi_t$, by mapping the target feature space to the source feature space such as $g : \chi_t \rightarrow \chi_s$, or by mapping both the source and target feature spaces to a common feature space such as $g : \chi_t \rightarrow \chi$ and $f : \chi_s \rightarrow \chi$. This mapping can be computed manually [107] or learned as part of the transfer learning algorithm[33, 44, 60, 88, 129].

When the mapping is part of the transfer learning algorithm a common approach is to apply a dimensionality reduction technique to map both source and target feature-space into a common latent space [72, 73, 76, 79]. For example, Chattopadhyay et al. [16] use Isomap [102] to map both the source and target data into a common low-dimensional space after which instance-based transfer techniques can be applied.

In some cases, meta-features are first manually introduced into the feature space and

then the feature space is automatically mapped from the source domain to the target domain [6, 20, 109]. An example of this is the work of Rashidi and Cook [91]. They first assign a location label to each sensor indicating in which room or functional area the sensor is located. Then activity templates are constructed from the data for both the source and target data, finally a mapping is learned between the source and target datasets based upon the similarity of activities and sensors [89, 90].

Parameter Transfer

Parameter transfer learns parameters which are shared between the source and target tasks. One common use of parameter transfer is learning a prior distribution shared between the source and target datasets. For example, one technique [13] models the source and target tasks using a Gaussian Mixture Model which share a prior distribution, another algorithm [33] learns a target classifier using a set of pre-trained classifiers as prior for the target classifier, and van Kasteren et al. [107] propose a method to learn the parameters of a Hidden Markov Model using labeled data from the source domain, and unlabeled data from the target domain. Later they extend this work to learn hyperparameter priors for the HMM instead of learning the parameters directly [109].

Another common example of parameter transfer assumes the SVM parameter w can be split into two terms: w_0, which is the same for both the source and target tasks, and v, which is specific to the particular task. Thus $w_s = w_0 + v_s$ and $w_t = w_0 + v_t$. Several works adopt this approach [69, 121].

Using a different approach to parameter transfer, a transfer learning algorithm [77, 79] can extract knowledge from the source domain to impose additional constraints on a quadratically-constrained quadratic program optimization problem for the target

domain. Along a similar line of thought, Zhao et al. [126, 127] use information extracted from the source domain to initialize cluster centers for a k-means algorithm in the target domain.

Relational-Knowledge Transfer

Relational-knowledge transfer applies to problems in which the data is not independent and identically distributed (i.i.d.) as is traditionally assumed but can be represented through multiple relationships [78]. Such problems are usually represented with a network or graph. Relational-knowledge transfer tries to transfer the relationships of in the source domain to the target domain. This type of transfer learning is not heavily explored, and as far as we are able to determine, no research is currently being pursued in transfer learning for activity recognition using relational-knowledge transfer.

2.5 Heterogeneous Transfer Learning

Up until this point we have focused on literature specifically addressing transfer learning for activity recognition applications. However, we can also look at additional transfer learning techniques which have not yet been applied to the activity recognition domain. We limit this to transfer learning approaches which can be applied to solve the new environment problem or the new sensing platform problem.

Domain adaptation is a specific branch of transfer learning that targets the case when the source and target data are not from the same domain. However, most of those works assume the difference is in the marginal probability distribution of the domains.

Daumé and Marcu model the probability distribution using a mixture model [29].

They assume that the source data comes from a mixture of a source probability distribution and a general probability distribution and that the target data similarly comes from a mixture of a target probability distribution and a general probability distribution. They then learn the parameters of these distributions from the data and use the source data to bolster the estimation of the the target data.

Several researchers apply feature-space transformations to overcome differences in the marginal probability distributions. Blitzer et al. [7, 8] propose Structural Correspondence Learning (SCL) to use the correlation between certain pivot features (which have the same semantic meaning in both domains) and other features to create a common feature representation. Pan et al. [75] construct a bipartite graph with connections between pivot features and non-pivot features that contain co-occurring feature values. They then apply spectral clustering to align the features and create a common feature-space representation.

Daumé et al. transform the source and target feature spaces into a higher dimensional representation with source, target and common components [30]. They then extend this to use unlabeled data by introducing co-regularization to force the source and target components to predict the same label on the unlabeled data [28]. Zhong et al. use kernel mapping to map features in the source and target domains to a new feature space where the conditional and marginal probabilities are more closely aligned [131]. They prove that a classifier trained in the new feature space has a bounded error.

Using a different approach, Pan et al. [73] perform domain adaptation via dimensionality reduction. Using Transfer Component Analysis [74], they reduce the distance between domains by projecting the features onto a shared subspace. As in the previous approaches, the technique focuses on the differences in the distribution of the data and

assumes the feature space is the same.

Chattopadhyay et al. use domain adaptation on multiple source domains to detect fatigue using SEMG signal data [16]. Their algorithm combines the output from multiple source classifiers to predict a label for unlabeled data in the target domain. These data instances are then combined with labeled data in the target domain and a final classifier is built. The label predictions from the multiple source domains are combined using a weighted voting schemed where the weights are based upon the similarity between the source and target domain at a per-class level.

We focus on transfer learning problems where the source and target domains are different because they have different feature spaces. This is commonly referred to as heterogeneous transfer learning in the literature and is formally defined below.

Definition 2.4 (Heterogeneous Transfer Learning) *Given a set of source domains* $DS = D_{s_1}, ..., D_{s_n}$ *where* $n > 0$*, a target domain,* D_t*, a set of source tasks* $TS = T_{s_1}, ...T_{s_n}$ *where* $T_{s_i} \in TS$ *corresponds with* $D_{s_i} \in DS$*, and a target task* T_t *which corresponds to* D_t*, transfer learning improves the learning of the target predictive function* $f_t()$ *in* D_t *where* $\chi_t \cap (\chi_{s_1} \cup ...\chi_{s_n}) = \emptyset$*.*

Dai et al. attempt solving the heterogeneous transfer learning problem by extending the risk minimization framework [55] and developing a translator between feature spaces based upon co-occurrence data (feature-feature, feature-instance, instance-feature, or instance-instance) between the source and target datasets [24]. Prettenhofer extends SCL to the heterogeneous transfer learning case by use a translation oracle (i.e. a domain expert or bi-lingual dictionary) to enumerate several pivot features. These pivot features are then correlated to the other features in both domains and a cross-lingual

classifier is trained [85].

Shi and Yu apply dimensionality reduction to heterogeneous feature spaces. In order to project the features from different feature spaces onto a single unified subspace they require that the data instances be linked as in multi-view learning. The ith data instance in the jth feature space is also the ith data instance in the kth feature space. Yang et al. extend the probabilistic latent semantic analysis (PLSA) [43] to improve image clustering results [123]. Images features are clustered to latent variables while annotations from social media are simultaneously clustered to the same latent variables. By clustering both the annotations and the image features the overall clustering results are improved.

Manual mapping strategies have also been used to overcome differences in the feature spaces. For example, Van Kasteren et al. [107, 109] group sensors by their location/function. Sensors in the source domain are then mapped to similar sensors in the target domain. Rashidi and Cook also map sensors based on location/function but apply additional transfer learning techniques to better align the source and target datasets [90, 91]. Our approach eliminates the need to manually map the feature spaces as this is handled by the algorithm. Additional domain adaptation approaches can then be applied to further improve the knowledge transfer. USFSR requires the manual specification of meta-features but this specification only occurs once and can be applied to map multiple source and target domains. The techniques of both Rashidi and Van Kasteren require a mapping to be defined for each source and target pair. Additionally, the manual mapping strategies are domain dependent, while FSR is applicable to a variety of different problems.

Many different techniques for heterogeneous transfer learning can be adapted from co-training or multi-view learning where instance-instance co-occurrence data is explicitly

available. Multi-view learning algorithms have been successfully applied to a variety of domains including natural language processing [83, 82], image recognition [18], wifi-localization [76], facial recognition [130] and robotic object recognition [48]. The co-training algorithm has been around for over a decade but continues to be a popular approach [9, 37, 49, 61, 93, 113]. However, no one has implemented these techniques for activity recognition using different sensor modalities as the multiple views [21].

2.6 Summary

The previous sections analyzed a large body of transfer-based activity recognition research along four different dimensions. Looking at each dimension separately provides an orderly way to analyze so many different papers. However, such separation may also make it difficult to see the bigger picture. Table 5, therefore, summarizes the classification of existing works along these four dimensions.

Table 5: Summarization of existing work based on the four dimension of analysis.

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[6]	wearables	new activities and labels	IS	feature-representation
[12]	wearables	different device, placement	TL	instance-based

Table 5: (continued from the previous page.)

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[13]	video camera	background, lighting, noise, and people	IS, US	parameter-based
[16]	wearables	people	IS	feature-representation and instance-based
[19]	wearables	people	IS	parameter-based
[20]	ambient sensors	location, layout, people	US	feature-representation
[33]	video camera	web-domain vs consumer domain.	IS	feature-representation and parameter-based
[35]	video camera	view angle	US	feature-space
[40]	wearables	people	US	instance-based
[44]	ambient sensors, wearables	label space, location	US	instance-based and feature-representation

Table 5: (continued from the previous page.)

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[45]	ambient sensors, wearables	label space	US	instance-based
[50]	wearables	people and setting	IS	instance-based
[53]	wearables	sensors	TL	instance-based
[56]	video camera	labels	IS	instance-based
[60]	video camera	view angle	US	feature-representation
[69]	video camera	activity sets, labels	IS	parameter-based
[72]	wearables	time	IS	feature-representation
[73]	wearables	time	US, UU	feature-representation
[76]	wearables	time	IS	feature-representation
[77]	wearables	space, location	US	parameter-based

Table 5: (continued from the previous page.)

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[79]	wearables	space, time, device	IS, US	feature-representation and parameter-based
[88]	ambient sensors	people	US	feature-representation
[89]	ambient sensors	layout, sensor network	IS, US	feature-representation
[90]	ambient sensors	layout, sensor network	IS, US	feature-representation
[91]	ambient sensors	layout, sensor network, people	IS, US	feature-representation
[95]	ambient sensors, wearables	devices	TL	instance-based
[107]	ambient sensors	location	US	feature-representation and parameter-based

Table 5: (continued from the previous page.)

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[109]	ambient sensors	location	US	feature-representation and parameter-based
[110]	wearables	people, setting	IS	instance-based
[111]	wearables	people, setting	IS	instance-based
[117]	video camera	labels	US	feature-representation
[119]	video camera	view angle	US	parameter
[120]	video camera	background, people	IS	instance
[121]	video camera	background, video domain	IS	parameter-based
[122]	wearables	space, time, device	IS, US	feature-representation and parameter-based
[124]	video camera	activities performed	IS	distance function

Table 5: (continued from the previous page.)

Paper	**Sensor Modality**	**Difference**	**Labeling**	**Type of Knowledge Transfer**
[126]	wearables	mobile device, sampling rate	US	parameter-based
[127]	wearables	people	US	parameter-based
[128]	ambient sensors, wearables	activity labels	US	instance-based
[129]	wearables	devices	IS	feature-representation

2.7 Grand Challenges

Although transfer-based activity recognition has progressed significantly in the last few years, there are still many open challenges. In this section, we first consider challenges specific to a particular sensor modality and then we look at challenges which are generalizable to all transfer-based activity recognition.

As can be seen in Table 7, performing transfer-based activity recognition when the source data is not labeled has not received much attention in current research. Outside the domain of activity recognition, researchers have leveraged the unlabeled source data to improve transfer in the target domain [27, 87, 116] but such techniques have yet to be applied to activity recognition.

Another area needing more attention is relational-knowledge transfer for activity recognition as indicated in Table 8. Relational-knowledge transfer requires that there

Table 6: Existing work categorized by sensor modality and the differences between the source and target datasets.

Sensor Modality	$\chi_s \neq \chi_t$	$\mathrm{P}(X_s) \neq \mathrm{P}(X_t)$	$Y_s \neq Y_t$	$f_s(x) \neq f_t(x)$
Video	[33, 35, 60, 117, 119]	[13, 33, 120, 121, 119]	[56, 69, 124]	[13, 33, 56, 60, 69, 120, 121, 124]
Wearable	[12, 53, 79, 95, 122, 126, 129]	[19, 40, 50, 72, 73, 76, 77, 79, 110, 111, 122]	[6, 44, 45, 128]	[6, 40, 44, 45, 50, 72, 73, 76, 77, 79, 111, 122, 128]
Ambient	[20, 89, 90, 91, 95, 107, 109]	[20, 88, 89, 90, 91, 107, 109]	[44, 45, 128]	[20, 44, 45, 88, 89, 90, 91, 107, 109, 128]

Table 7: Existing work categorized by sensor modality and data labeling.

Sensor Modality	**Informed Supervised**	**Uninformed Supervised**	**Informed Unsupervised**	**Uninformed Unsupervised**
Video	[13, 33, 56, 69, 120, 121, 124]	[13, 35, 60, 117, 119]	-	-
Wearable	[6, 19, 72, 76, 79, 110, 111, 122, 129]	[40, 44, 45, 50, 73, 77, 79, 122, 126, 127, 128]	-	[73]
Ambient	[89, 90, 91]	[20, 44, 45, 88, 89, 90, 91, 107, 109, 128]	-	-

Table 8: Existing work categorized by sensor modality and the type of knowledge transferred.

Sensor Modality	**Instance Based**	**Feature Representation**	**Parameter Based**	**Relational Knowledge**
Video	[56, 120]	[33, 35, 60, 117]	[13, 33, 69, 119, 121, 124]	-
Wearable	[12, 40, 44, 45, 50, 53, 95, 110, 111, 128]	[6, 72, 73, 76, 79, 122, 129]	[19, 77, 79, 122, 126, 127]	-
Ambient	[44, 45, 95, 128]	[20, 44, 88, 89, 90, 91, 107, 109]	[107, 109]	-

exist certain relationships in the data which can be learned and transferred across populations. Data for activity recognition has the potential to contain such transferable relationships indicating that this may be an important technique to pursue. See [31, 64, 65, 66] for examples of relational-knowledge transfer.

Tables 6-8 also indicate several more niche areas which could be further investigated. For example, in the video camera domain, most of the work has focused on informed supervised parameter-based transfer learning, while the other techniques have not been heavily applied. Similarly, transferring across different labels spaces is a much less studied problem in transfer-based activity recognition. Finally, we note that parameter-based transfer learning is also less studied for the ambient sensor modality.

The current direction of most transfer-based activity recognition research is to push the limits on how different the source and target domains and tasks can be. The scenarios discussed in Section 2.3.2 illustrate the importance of continuing in this direction. More work is needed to improve transfer across sensor modalities and to transfer knowledge across multiple differences. To fill this need, we develop techniques for transferring knowledge between heterogeneous feature-spaces. This, in turn, provides solutions to the new environment and new sensing platform problems, enabling the creation of personalized activity recognition ecosystems.

Chapter 3

New Environment Problem

3.1 Introduction

Traditional supervised machine learning techniques rely on the assumptions that the training data and test data have similar probability distributions and that the classification task is the same for both datasets. Ideally we would like to be able to use labeled data from a different domain to improve learning in the target domain. One example would be to use labeled data from one or more smart apartments to recognize activities in a new smart apartment which may have a different layout, different residents, or different lifestyles or behavioral patterns. Another example would be using the labeled data from a smart apartment to perform activity recognition in a smart office. The previous examples would not only exhibit different probability distributions between the source and target domains, but also likely have entirely different feature spaces. In these cases, traditional machine learning techniques often fail to correctly classify the test data.

With heterogeneous learning, transfer between vastly different domains becomes feasible. The majority of heterogeneous transfer learning techniques map the source feature space to the target feature space or to map both the source and target feature space to a shared feature space. However, we show that by reversing this model and mapping the target feature space to the source feature space one can leverage an existing hypothesis in the source feature space to find a better mapping between feature spaces. Additionally,

by mapping the target feature space to the source feature space one can easily create ensemble learners which further improve the accuracy of the proposed techniques.

We propose a class of novel heterogeneous transfer learning techniques, Feature-Space Remapping (FSR), which is capable of handling different feature spaces without the use of a translation oracle or instance-instance co-occurrence data. We term the technique a "remapping" because the original raw data is already mapped onto a feature space and FSR remaps the data to a different feature space. The technique can be used in either the informed or uninformed transfer learning setting and we provide details for both cases. FSR uses only a small amount of labeled data in the target domain to infer relations to the source domain and can optionally operate without any labeled data in the target domain or other linkage data. For simplicity, we present FSR here assuming the feature-space is a vector of real-valued numbers. However, it is straightforward to extend the FSR approach to handle categorical or discrete values as well.

In addition to presenting FSR for transferring knowledge from a single source domain to a target domain, we also show how FSR can effectively combine the information from multiple source domains by using an ensemble learner to increase the classification accuracy in the target domain.

3.2 Background

Although FSR focuses on different feature spaces, it does not rely on the other dimensions of the transfer learning problem remaining constant. Indeed the datasets we use in the experimental section have differences in the marginal probability distributions as well as in the label space. As with all transfer learning problems we do rely on the basic

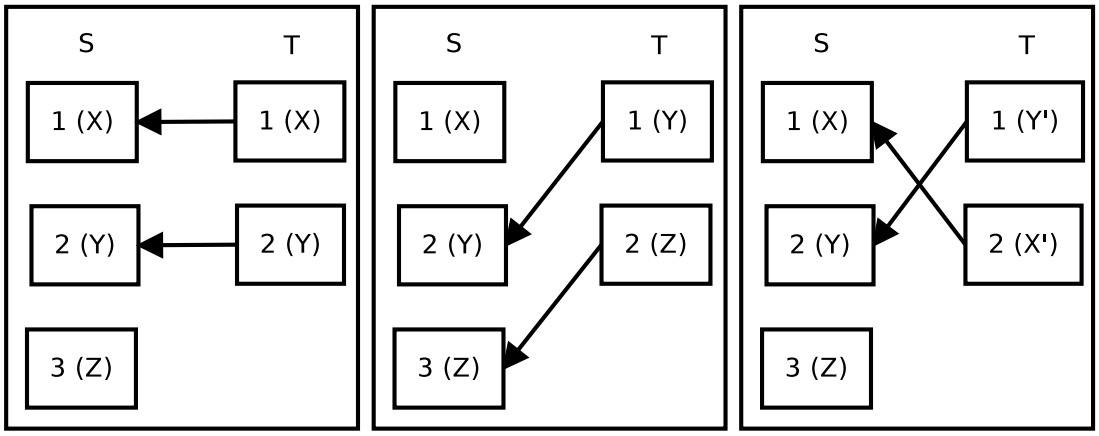
assumption that there exists some relationship between the source and target areas which allows for the successful transfer of knowledge from the source to the target.

When the feature spaces of the domains are different, we assume that they can be different both in terms of the number of dimensions and in the organization of the dimensions. To illustrate this point, consider two different domains, one consisting of two dimensional data and the other consisting of three dimensional data. It could be the case that the first two dimensions are the same in both domains (see Figure 3a); however, it could also be the case that the first two dimensions of the target domain correspond with the last two dimensions of the source domain (see Figure 3b), or perhaps only the first dimension of the target domain corresponds with the last dimension of the source domain. It may even be the case that the dimensions are entirely different, but a mapping between dimensions could still allow the knowledge gained in one domain to be used effectively in the other domain (see Figure 3c). FSR learns a mapping from the target feature space to the source feature space regardless of the exact differences between dimensions.

3.2.1 Illustrative Example

Before describing our new FSR algorithms named GAFSR, GrFSR, and SFSR, we put forward an example transfer learning scenario to illustrate the concepts introduced throughout the discussion. To that end, let us consider the transfer learning problem for activity recognition in a smart environment using ambient sensors.

Ambient sensors are typically embedded in an individual's environment. Examples of ambient sensors may include motion detectors, door sensors, object vibration sensors,



(a) Map 1 (b) Map 2 (c) Map 3

Figure 3: Example mappings from target T (two-dimensional data) to source S (three-dimensional data). Some features may be equivalent in both feature-spaces as in Map 1 and Map 2 or the features may just be similar as in Map 3.

pressure sensors, and temperature sensors. As the name indicates, these sensors are designed to disappear into the environment while collecting a variety of activity related information such as human movements in the environment induced by activities, interactions with objects during the performance of an activity, and changes to illumination, pressure and temperature in the environment due to activities. Table 1 shows some example data from a smart home with ambient motion sensors.

Suppose there are two homes (a source home and a target home) equipped with these ambient sensors. The source home already has an activity recognition model trained for that home. The target home does not yet have an activity recognition model trained. In order to use the model from the source home to recognize activities in the target home, they must use a common feature-space. A common approach to activity recognition using ambient sensors is to formulate the problem as a bag of sensors approach over some sliding window of time or sensor events. This means that the sensors from one home must be mapped onto the sensors from the other home. Specifically, the features

of one domain must map onto the features (or dimensions) of the other domain. This could be accomplished by mapping the sensors in the target home to the sensors in the source home, mapping the sensors in the source home to sensors in the target home, or mapping both the source and target sensors to a common set of generic labels (for example, location-based mapping such as kitchen, bedroom, etc).

This mapping is just the initial step in the transfer learning. Once a shared feature-space is achieved, additional transfer learning may be necessary to resolve differences in the marginal probabilities (the residents in one home may spend half the day sleeping, while the residents in the other home only sleep 6 hours a day) or differences in the classification task (the set of activities recognized may be different). The techniques we present here focus on achieving this initial transformation of the feature-space.

3.3 Methods

Traditionally, domain adaptation problems have focused on the case when $D_s \neq D_t$, usually because $P(X_s) \neq P(X_t)$. For example, in activity recognition, the behavior of an individual may change over time, multiple individual may utilize the same space differently. This creates situations where the feature-space has not changed by the probability distribution of the features over that feature-space has changed. When domain adaptation has been applied to problems where $\chi_s \neq \chi_t$ there is usually a trivial transformation between feature spaces. An example of this is found in document classification, where the domain dimensions are typically word counts in each document. To compare documents with different words, a user can set the word counts for the unseen words to zero. This allows the user to easily define a common feature space between documents.

Additional transfer learning techniques may still be necessary because $P(X_s) \neq P(X_t)$ but the initial feature-space transformation is trivial. This trivial transformation works because the semantic meaning of the dimensions is assumed to be known.

In this work, we present a heterogeneous transfer learning algorithm where the feature space must be transformed in a non-trivial manner, as is the case in the new environment problem. The semantic meaning of the dimensions is assumed to be either unknown or incompatible between the source and target domains. In the activity recognition domain, this is equivalent to having sensor values but not knowing from which sensor (type or location) it originated. Unlike many other heterogeneous transfer learning techniques, we do not rely on co-occurrence data such as dictionaries, social annotations of images, or multi-view data. Additionally, we do not assume that $P(Y_s|X_s) = P(Y_t|X_t)$ or even that $Y_s = Y_t$ but we do assume that they must still be related.

We specifically consider the case when both the source and target domains can be represented by a bag-of-features and related features have similar value distributions. This does not account for features which may be related through a linear or non-linear transformation such as $x = 10y+3$. Differences in linear scaling can be removed through the application of normalization techniques but this may cause the FSR technique to incorrectly map features which would otherwise be clearly unrelated.

To achieve the desired feature-space transformation, we view the problem as a new machine learning task to learn a mapping from each dimension in the target feature space to a corresponding dimension in the source feature space. More formally this can be written as follows: Given source data X_s, target data X_t and a hypothesis $H_s : \chi_s \rightarrow Y_s$ find a mapping $\theta(\chi_t, \chi_s)$ such that $error_\theta(H_s)$ is minimized where $error_\theta(H_s)$ represents the empirical error on the target domain by using H_s on the mapped target data. Notice

the distinction between this problem definition and other approaches typically applied to heterogeneous transfer learning. Traditional heterogeneous transfer learning approaches usually map source features to target features or source and target features to a common feature space and then learn a hypothesis on this common feature space. In our approach however, we map the target features to source features and we use an already learned hypothesis to guide the mapping process and avoid the duplication of work. If the mapping process proceeded in the other direction we would need to relearn a new hypothesis for each step of the search which would greatly increase the computational complexity of these techniques. By mapping from target to source we also gain the ability to combine multiple data sources through ensemble learning which will be discussed in Section 3.4. It is possible to relearn a new hypothesis after performing the mapping. It is also possible to apply additional transfer learning approaches after first obtaining a unified feature-space.

The number of possible mappings between source and target feature spaces grows exponentially as the number of features increases. Even for lower dimensional data, searching through all possible mappings quickly becomes computationally infeasible. Using feature-feature, feature-instance or instance-instance co-occurrence data could be used to guide the search but FSR operates under the assumption that this type of data is not available. When labeled data is available in the target domain, the empirical error of a classifier tested on the mapped data can provide a quantitative method for evaluating candidate mappings. Using the labeled target data we present three FSR algorithms. First we present Genetic Algorithms for Feature-Space Remapping (GAFSR) which explores the search space using random permutations of possible mappings. This method is the most computationally expensive but also explores the largest amount of mapping

space. Next we present Greedy Search for Feature-Space Remapping (GrFSR) which applies the fitness function of the genetic algorithm to greedily select an approximation to the optimal mapping without searching through all possible mappings. Finally, we present Similarity Feature-Space Remapping (SFSR) which uses less computationally expensive heuristics to select an approximation to the optimal mapping by comparing the similarity between features in the source and target space. The SFSR technique has two different variations Informed SFSR (ISFSR) and Uninformed SFSR (USFSR).

The techniques we propose generate a many-to-one mapping. This is because multiple dimensions (features) in the target space can be mapped to a single feature in the source space but one feature in the target domain will never map to multiple features in the source domain. We could make the mapping stricter by enforcing a one-to-one mapping (with null mappings allowed) or we could relax the mapping by allowing weighted many-to-many mappings. However, if we allowed a many-to-many mapping the search space (which is already too large for brute-force searching) would grow even larger. For many situations, a many-to-one mapping makes the most sense intuitively. For example, consider a hallway which is lined with several narrow-view motion sensors in one apartment and a hallway which has a single wide-view motion sensor in another apartment. Each narrow-view motion sensor could map to the single wide-view motion sensor in the other apartment but the wide-view motion sensor should just map to the single narrow-view motion sensor which best encapsulates its behavior.

After the mapping has been obtained, the mapping must be applied to the target data to be classified using the hypothesis learned on the source data. Because these techniques produces a many-to-one mapping, the procedure for combining the multiple dimensions must also be defined. For dimensions with numerical values, one could use an

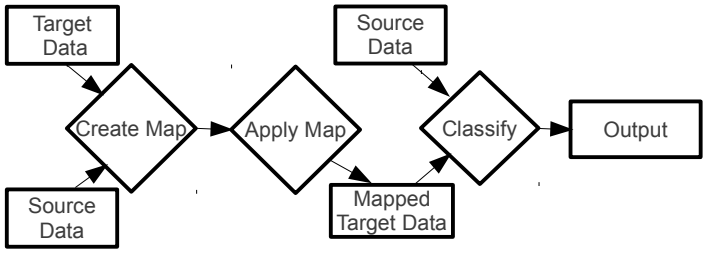


Figure 4: Flowchart of the mapping process. The source and target data are first analyzed to find a mapping from the target features to the source features. Next, the target data is mapped onto the source feature space. Finally, the mapped target data can be classified using a classifier which has been trained on the source data.

aggregate value such as minimum, maximum, total, or average. For categorical values, one could use a voting protocol. For each instance in the target data the features are mapped to the source features. When multiple features in the target data are mapped to single feature in the source data, the feature values are combined using the specified aggregation protocol. In this work we use the summed value to aggregate target features mapped to the same source feature. The entire process is summarized in Figure 4. The techniques differ in how they create the map but the rest of the steps are all identical.

3.3.1 Genetic Algorithm Feature-Space Remapping

The goal of the GAFSR technique is to find a near optimal mapping $\theta(\chi_t, \chi_s)$ such that the error of the hypothesis on the target data is minimized. If n is the number of features in χ_s and m is the number of feature in χ_t then there are n^m possible mappings, making it impractical to try all possible mappings. Instead, GAFSR uses a standard genetic algorithm approach to explore the search space looking for reasonable solutions.

Genetic algorithms are a class of local search techniques which has been studied for the past several decades [67, 96]. The motivation for genetic algorithms is rooted in the

biological process of reproduction and evolution [38]. The basic idea is to start with a random population of strings (chromosomes) and evaluate their fitness according to a specified function (the fitness function). Chromosomes pairs are then selected (usually probabilistically according to the normalized fitness score) and the selected chromosomes are mated via crossover to produce new offspring. This process is then repeated a number of times until a stopping criterion is met. Pseudocode for GAFSR is found in Algorithm 1. The key components of a genetic algorithm are: the chromosome definition, the fitness function, and the mutation parameters. Each are described below.

Algorithm 1: GAFSR Algorithm

```
Data: X_t Target Features
Data: X_y Source Features
Data: s population size, g number of generations
Data: c cross-over rate, r mutation rate
Generate s initial random mappings from X_t to X_y // i.e Chromosomes
for i ← 0 to g do
    Evaluate fitness of each mapping;
    Select pairs of mappings probabilistically weighted according to fitness;
    With probability c, Swap portions of mappings between the selected pair;
    With probability r, Mutate the mappings;
Evaluate fitness of each mapping;
Return mapping with best fitness;
```

The chromosome is defined such that each sensor in the target dataset is a gene with $n+1$ possible values (1 for each source feature plus a null feature), thus the chromosome is composed of m genes with $n+1$ possible values for each gene. From a practical standpoint, n can be seen as the number of sensor in the source domain and m can be seen as the number of sensor in the target domain when using a bag of sensors approach to the activity recognition problem.

We compare two different fitness functions based upon the unweighted average recall

(UAR) and the accuracy (ACC) of the target dataset obtained using a naïve Bayes classifier which has been trained on the source dataset. The unweighted average recall is given by Equation 3.1 and the accuracy is given by Equation 3.2. In both of these equations N is the total number of instances, K is the number of labels, and A is the confusion matrix where A_{ij} is the number of instances of class i classified as class j.

$$UAR = \frac{1}{K} \sum_{i=1}^{K} \frac{A_{ii}}{\sum_{j=1}^{K} A_{ij}} \tag{3.1}$$

$$ACC = \frac{1}{N} \sum_{i=1}^{K} A_{ii} \tag{3.2}$$

The first fitness function, given in Equation 3.3, is defined as just the UAR of the target dataset obtained using a naïve Bayes classifier which has been trained on the source dataset. We use the average recall instead of the overall accuracy because the datasets are imbalanced. A large percentage of the instances are represented by only a few class labels. Using the unweighted average class recall is one technique for accounting for this imbalance [107]. The second fitness function, given in Equation 3.4, is defined as the twice the UAR plus the overall accuracy (ACC) on the target dataset obtained using a naïve Bayes classifier which has been trained on the source dataset. This function is chosen to improve the overall accuracy obtained by the mapping technique while still preserving the high unweighted average recall.

$$F1 = UAR \tag{3.3}$$

$$F2 = ACC + 2 * UAR \tag{3.4}$$

The parameters of the genetic algorithm are chosen using limited validation testing to find parameters which yield decent results. They are set to the following values:

- Population Size: 118
- Mutation rate: .06
- Crossover rate: .80
- Crossover type: 2-point cross-over
- Number of Generations: 100

In addition, we use the technique referred to as elitism where the best solution so far is preserved across generations. This prevents the algorithm from losing the best solution due to the random mutations and crossovers.

The asymptotic run-time of the proposed genetic algorithm is $O(S*G*N*d_s)$ where S is the size of the population, G is the number of generations N is the number of labeled target instances and d_s is the number of dimensions in the source feature-space. We have purposely excluded the cost of creating the population for each generation because the cost of the fitness function shown here is the dominating factor. For the size of the activity recognition datasets we test here, $S * G \approx d_s^2$ giving us an asymptotic run-time of $O(N * d_s^3)$.

3.3.2 Greedy Search for Feature-Space Remapping

An alternative to genetic methods for searching a space is applying a greedy search, which does not rely on the partially-random biologically-inspired search mechanism found in

genetic algorithms. To compare our genetic solution to a greedy approach, we introduce GrFSR which applies the same fitness function employed by the genetic algorithm to greedily search through the mapping space and find an approximation to the best mapping function.

The greedy algorithm selects a single feature in the target domain to consider. It then maps this feature to all possible features in the source domain one at a time (including the null feature, which in effect ignores the corresponding feature in the target domain) while all other features in the target domain are mapped to null. The resulting mapping is applied to the labeled target data and tested using the hypothesis obtained from the source data. The mapping that produces the best result according to Equation 3.4 is selected as the best mapping for that target feature. This is repeated for all the target features. A final mapping is produced by combining the best mapping produced for each target feature. Pseudocode for the algorithm is given in Algorithm 2.

Algorithm 2: GrFSR Algorithm

```
Data: X_t Target Features
Data: X_s Source Features
for x ∈ X_t do
    for y ∈ X_s do
        fit[y] ← Fitness(x → y);
    mapping[x] ← max(fit);
Return mapping;
```

The asymptotic run-time of GrFSR is $O(d_s * d_t * N * d_s)$ where d_s is the number of dimension in the source feature-space, d_t is the number of dimension in the target feature-space and N is the number of labeled data instance in the target domain. This run-time is equivalent to $O(Nd^3)$ if $d_s \approx d_t$.

3.3.3 Similarity Feature Space Remapping

In Similarity Feature Space Remapping, rather than exploring the entire search space of possible mappings we instead use heuristics to select a mapping that approximate the optimal mapping. SFSR computes meta-features as a means to relate source and target features. These meta-features can be defined and computed multiple ways which will be discussed in Sections 3.3.3 and 3.3.3. Algorithm 3 shows the pseudocode for the SFSR technique and each step is discussed in detail below. To simplify the presentation of the SFSR algorithm, for now let us assume that meta-features have already been calculated for the source and target features. One can think of the meta-features as a vector of numbers assigned to each feature in the source and target space. These vectors can then be compared to each other to find features with similar meta-features.

Algorithm 3: SFSR Algorithm

```
Data: X_t Target Features
Data: X_y Source Features
for x ∈ X_t do
    metas[x] ← ComputeMetaFeatures(x);
for x ∈ X_s do
    metas[x] ← ComputeMetaFeatures(x);
for x ∈ X_t do
    for x ∈ X_s do
        similarity[x][y] ← ComputeSimilarity(metas[x], metas[y])
for x ∈ X_t do
    mapping[x] ← max(similarity[x]);
Return mapping;
```

SFSR computes a similarity matrix S between source features and target features. This is done by computing a similarity score for each feature-feature pair based upon the meta-features computed for the given features. The similarity score is computed as

the average similarity between the source and target meta-feature values. Formally, this score is given by Equations 3.5 and 3.6.

$$S_{xy} = \frac{1}{N} \sum_{i=1}^{N} \Omega(m_x^i, m_y^i) \tag{3.5}$$

where x is the xth source feature, y is the yth target feature, N is the number of meta-features and Ω is the normalized similarity between two meta-features m_x^i and m_y^i, the ith meta-feature of feature x and y respectively. We calculate the normalized similarity between two meta-features as the absolute value of the difference between meta-feature values divided by the maximum possible difference between the meta-features to obtain a normalized value between 0 and 1. This is shown in Equation 3.6.

$$\Omega(m_x^i, m_y^i) = 1 - \frac{|m_x^i - m_y^i|}{\max(m_x^i, m_y^i \forall x \in D_s \forall y \in D_t) - \min(m_x^i, m_y^i \forall x \in D_s \forall y \in D_t)} \tag{3.6}$$

If the meta-feature values are all positive, which is the case for the experiments we show here, the normalized similarity equation can be simplified to:

$$\Omega(m_x^i, m_y^i) = 1 - \frac{|m_x^i - m_y^i|}{\max(m_x^i, m_y^i \forall x \in D_s \forall y \in D_t)} \tag{3.7}$$

SFSR computes a mapping $L : y \rightarrow x$ by selecting source feature x with maximal similarity to target feature y as given by the similarity matrix S.

$$L(y) = \arg\max_{x \in D_s} S_{xy} \tag{3.8}$$

If we assume the meta-feature computation is linear, SFSR has a running time of $O(d_s * d_t + n + m)$ where d_s and d_t is the dimensionality of the source and target data, respectively, and n and m are the number of source and target instances, respectively.

This run-time is explained by the following observations. First, each dimension in the target domain is compared to each dimension in the source domain, resulting in the $d_s * d_t$ term. Second, assuming the meta-feature computation is linear in the number of data instances, then computing the meta-features requires $O(n + m)$ time. Finally, applying the mapping requires a single pass through the target data or $O(m)$ time.

As mentioned earlier, the defining and calculating of meta-features can be done in multiple ways. If some labeled target data is available, it can be used to calculate domain-independent meta-features (i.e. meta-features that can be applied to any heterogeneous transfer learning problem). We refer to this as Informed Similarity Feature-Space Remapping (ISFSR) because it requires the labeled target data. If no labeled target data is available then domain-dependent meta-features must be defined. We refer to this as Uninformed Similarity Feature Space Remapping (USFSR) because it does not require the label target data.

Informed Similarity Feature-Space Remapping

Searching through all possible mappings to find the mapping which minimizes the error of the hypothesis on the target data is computationally expensive. However, since the hypothesis has been learned using the source training data one would expect the error to be minimized by selecting mappings for which the feature-label co-occurrence data is similar in the source and target datasets. This leads to our first heuristic for mapping source and target features. ISFSR computes the feature-label co-occurrence data for each feature in the source and target space by calculating the expected value of the feature given the label using the labeled training data. More formally, if $Y = Y_s \cup Y_t$

then the feature-label co-occurrence data for each feature and label is computed as:

$$E(x|c) = \frac{1}{n_c} \sum_{i=1}^{n} x_i \tag{3.9}$$

where x is the feature, c is the label such that $c \in Y$, n_c is the number of data instances with label c, x_i is the value of feature x on the ith data instance with a label of c. This assumes a real-valued number space. One could easily extend this to categorical values by using the count of occurrences of each category as an estimation of the probability that the given feature will have the given categorical value.

Each feature-label co-occurrence value now becomes a meta-feature for the given feature. Thus $E(x|c)$ is a meta-feature for feature x and x will have $z = |Y|$ such meta-features, one for each label c. Using feature-label co-occurrence data as a meta-feature keeps the ISFSR asymptotic run time within the previously stated bound of $\mathrm{O}(d_s * d_t + n + m)$. This is because the meta-feature calculation is linear in the number of instances. We compute $E(x|c)$ for each label $c \in Y$. This can be done in a single pass through the datasets and thus requires $O(n + m + y)$ time. Typically $n >> y$ and $m >> y$ so this term can be simplified to just $O(n + m)$.

Additionally, using feature-label co-occurrence data for the meta-features provides domain independent meta-features so that meta-features for the specific problem do not need to be specified by a domain expert. Thus any domain for which labeled data exists can apply this feature mapping technique without setting any parameters, defining any relations, or defining any additional meta-features.

To understand why using the the feature-label co-occurrence data as a heuristic to find an approximation to the optimal mapping works we go back to the original problem definition. Given source data X_s, target data X_t and a hypothesis $H_s : \chi_s \rightarrow Y_s$

find a mapping $\theta(\chi_t, \chi_s)$ such that $error_\theta(H_s)$ is minimized. This error is minimized by maximizing the number of agreements between $H_S(\theta(q))$ and $f_t(q)$ as shown in Equation 3.10 where q is a data instance in X_t and $\theta(q)$ is the mapped data in the source domain space.

$$\max_{\theta} \sum_{q \in X_t} \begin{cases} 1, & \text{if } H_s(\theta(q)) = f_t(q). \\ 0, & \text{if } H_s(\theta(q)) \neq f_t(q). \end{cases} \tag{3.10}$$

A naïve Bayes classifier can learn a hypothesis by estimating $P(c)$ and $P(q_i|c)$ based upon their observed frequencies and applying Bayes rule to estimate the posterior probability $P(c|q)$. The class c with the highest posterior probability is selected as the class label for q [68]. Thus, if the hypothesis is expressed as a naïve Bayes classifier and if we approximate the true predictive function $f_t()$ also using a naïve Bayes formulation then Equation 3.10 can be expressed as shown in Equation 3.11.

$$\max_{\theta} \sum_{q \in X_t} \begin{cases} 1, & \text{if } \arg\max_{c \in Y} P(c) \prod_{i=1}^{d_t} P(\theta(q_i)|c) = \arg\max_{c \in Y} P(c) \prod_{i=1}^{d_t} P(q_i|c). \\ 0, & \text{if } \arg\max_{c \in Y} P(c) \prod_{i=1}^{d_t} P(\theta(q_i)|c) \neq \arg\max_{c \in Y} P(c) \prod_{i=1}^{d_t} P(q_i|c). \end{cases} \tag{3.11}$$

Under this representation, selecting the mapping for each feature that has the most similar feature-label co-occurrence value can be seen as a greedy approximation to minimize the empirical error on the mapped target data. Indeed, when the feature values are restricted to either 0 or 1, the feature-label co-occurrence value $E(x|c)$ is equivalent to the estimation of the probability that the feature has a value of 1 given the class label, $P(x = 1|c)$.

Uninformed Similarity Feature-Space Remapping

When labeled data is unavailable in the target domain we still need some way to link correlated source and target features. In this case we define meta-features which can be used as a heuristic to guide the mapping process. Meta-features should have the following attributes: 1) The meta-features should not depend on any relationship between different features. 2) Features with similar meta-feature values should also have similar conditional probability distributions. The first stipulation allows meta-features to be applied, calculated and compared between different feature spaces.

To clarify this concept consider the following examples. In activity recognition using motion sensors, the time of day when motion sensor A fires would be an acceptable meta-feature. On the other hand, the amount of time between sensor A firing and sensor B firing would not be an acceptable meta-feature because it depends on the relationship between sensor A and sensor B. However, the amount of time between sensor A firing and any unspecified sensor firing is acceptable because it again depends only upon sensor A.

The second stipulation is important because it provides a basis for using the meta-features as a heuristic to select a mapping between features. The meta-features provide some indication that the features have similar conditional probability distributions and if the conditional probability distributions of two features are similar then the mapping process will be more likely to select that pair for mapping.

Defining meta-features and creating the feature dataset is a domain specific task. The meta-features used for activity recognition may not be applicable to the document classification domain. In Table 9 we describe the meta-features we use for the activity

recognition problem and using the example data shown in Table 1 we show the meta-feature values for some of the sensors. These meta-features have been chosen to be consistent with the previously discussed meta-feature stipulations.

Table 9: Meta-features defined for activity recognition.

Meta-feature Description	**Meta-Feature**	**M021**	**MA020**	**M018**
average sensor event frequency over 1 hour time periods (x24)	03:00	3	1	0
	04:00	2	0	0
	05:00	2	0	0
	06:00	4	0	0
	07:00	0	0	0
	08:00	14	14	2
average sensor event frequency over 3 hour time periods (x8)	03:00	7	1	0
	06:00	18	14	2
average sensor event frequency over 8 hour time periods (x3)	00:00	11	1	0
	08:00	14	14	2
average sensor event frequency over 24 hour time periods (x1)	00:00	25	15	2
average and standard deviation of the time of day of this sensor event (seconds)	avg.	26015.36	30356.40	31594.5
	std. dev.	6831.72	4555.85	2.50
average and standard deviation of the time between this sensor event and the previous sensor event (seconds)	avg.	760.79	1.34	1.31
	std. dev.	1862.31	1.02	0.52
average and standard deviation of the time between this sensor event and the next sensor event (seconds)	avg.	722.94	13.66	5.85
	std. dev.	1833.35	44.59	2.11
average and standard deviation of the time between this event and the next event from this sensor (seconds)	avg.	761.41	1305.67	4.53
	std. dev.	1862.06	4699.58	0.0
probability the next sensor event is from the same sensor	prob.	0.76	0.72	0.0

All of these meta-features can be computed in linear time therefore the asymptotic run time of $O(d_s * d_t + n + m)$ is still achieved.

As an extension, if labeled target data is available, one could easily combine the domain-dependent meta-features with the feature-label co-occurrence meta-features to provide additional information when selecting a feature-space mapping. One could also compute the features on a per-class basis. For example, the frequency of a sensor event

could instead be computed as the frequency of a sensor event given the activity label. However, in order to avoid over-fitting the data, this may require more labeled data than is typically available in transfer learning scenarios.

3.4 Combining Multiple Data-sources

One of the major benefits of the above mapping approaches is that they can be used to combine data from multiple source domains in a straightforward manner. One example where multiple source domains might arise is a single individual with labeled data in multiple smart environments (home, office, car, etc). Another example would be multiple smart apartments with labeled data which can be used to recognize activities in new smart apartment. An ensemble classifier can be built by mapping the target domain to each source domain and training a separate base classifier for each source domain. The output from these source classifiers can then be combined by the ensemble meta-classifier to make the final prediction. We refer to this as Ensemble Learning via Feature-Space Remapping (ELFSR).

Ensemble methods have been used in a variety of situations with great success. According to Hansen and Salamon, a necessary and sufficient condition for ensemble classifiers to be more accurate than any of the individual classifiers are for the classifiers to be accurate and diverse [42]. An *accurate* classifier is one which has a classification accuracy better than random guessing [32]. Two classifiers are diverse if the errors they make are different (and preferably uncorrelated) [32]. Most ensemble techniques defined to date generate a set of diverse classifiers. Bagging, for example, generates classifiers by repeatedly sub-sampling the original data with replacement [10]. Boosting iteratively

reweights samples based on the accuracy of the previous iteration [36]. In ELFSR, each classifier is drawn from a different domain, leading to a naturally diverse set of classifiers.

Once the classifiers are generated, the output must be combined to obtain the final result. Several approaches have been used including majority voting, weighted voting, summing the probabilities, and training a new learner on the output of the classifiers or *stacking* [118]. Stacking is a supervised technique and thus requires additional labeled data to train the ensemble classifier. This means that stacking can be readily combined with ISFSR, which already uses labeled data.

Work on ensemble classifiers for transfer learning has mainly focused on boosting techniques [80, 120, 125]. As there has been very little work on transfer learning using voting or stacking ensemble classifiers, we compare the results of several different ensemble configurations using activity recognition from multiple smart apartments as the source domains and activity recognition for a different smart apartment as the target domain. Specifically, we consider two voting ensembles (a majority voting ensemble and a summation voting ensemble), and two stacking ensembles (via naïve Bayes and via a decision tree). The voting ensembles have the advantage of not requiring any labeled data in the target domain, while the stacking techniques require a small amount of labeled data.

3.4.1 Voting Ensemble

One of the simplest methods for combining multiple classifiers is through majority voting. Each classifier votes for the class label it predicts for the given instance and the label receiving the most votes wins.

The drawback to the majority voting ensemble classifier is that the ensemble throws away important information by only considering the most likely label as predicted by each classifier. The summation voting ensemble classifier rectifies this weakness by summing up the predicted probability of each label for each classifier and then assigning the label with the highest summed probability.

3.4.2 Stacking

In stacking, the output of each source classifier is fed into the ensemble classifier which then produces the final classification. Here we consider two different classification algorithms for the ensemble classifier, naïve Bayes and decision trees. One of the drawbacks to using stacking is the requirement of labeled data to train the ensemble classifier. Rather than test both USFSR and ISFSR with the stacking technique we only consider the result of using ISFSR since ISFSR already uses a small amount of labeled data in the target domain. We use stacking with ISFSR without requiring any additional labeled data in the target domain.

3.5 Conclusions

The new environment problem is encountered every time a sensing platform is deployed to a new environment. This often leads to a new feature-space since the environment is likely to have require different sensors in different quantities in different locations. The previous approaches to solving this problem requiring learning a new model specific to the new environment or attempting to learn a generalized model using a domain expert to provide a mapping between feature-spaces. In this chapter we have proposed

Feature-Space Remapping as a novel technique to handle the new environment problem. The FSR algorithms we propose each have different trade-off in regards to the run-time complexity, the amount of labeled data required, and the availability of source feature-spaces. GAFSR and GrFSR are both more expensive in terms of run-time but will likely yield a better mapping than ISFSR. ISFSR is more efficient but will likely yield poorer mappings. USFSR is the only technique which can be used when no labeled data is available in the target domain and also runs efficiently. USFSR also requires domain-specific knowledge. ELFSR is the best technique when multiple source datasets are available since it is able to combine information from each dataset.

Chapter 4

New Sensing Problem

4.1 Introduction

Every day brings new advances in ubiquitous computing. Sensing and data processing capabilities are being introduced and embedded into our homes, our phones, our cars, our clothing and our world. Despite the ever-increasing prevalence of heterogeneous sensing platforms, most activity recognition research remains segmented and focused on individual sensor types. Researchers are developing activity recognition algorithms using cameras, wearable accelerometers, or ambient motion sensors with little overlap between different sensor modalities. The resulting techniques fine-tune performance for an isolated class of devices but do not make effective use of other sensor devices as they become available.

Neglecting the presence of additional sensor devices ignores the wealth of information that may otherwise be readily available. In this chapter, we introduce a method for different smart devices and sensors to share information in order to achieve more accurate activity recognition. We postulate that heterogeneous devices can collaborate, utilizing data from smart phones, smart homes, smart vehicles, and other data sources to create a personal activity recognition ecosystem. We focus specifically on the ability to transfer knowledge between heterogeneous activity recognition systems with the goal of increasing the accuracy of the collaborative system while decreasing the amount of

labeled data that is necessary to train the system.

As an example, consider the problem of activity recognition in a smart home. A model can be trained to recognize an activity that occurs in a particular home based on motion sensor data. If the user also wants to start using a phone-based recognizer, the labeling-and-training process must be repeated. To overcome this problem, we propose designing inter-device multi-view learning techniques to allow the existing smart home to act as a teacher for the new smart phone. We compare alternative multi-view approaches and empirically compare the approaches using heterogeneous activity data. We also consider extensions to the multi-view approaches which can handle three or more views. Each view may have different amounts of labeled training data. Each view may also have differ in their ability to differentiate between classes. This results in the various views having different levels of achievable accuracies. We empirically evaluate the impact of different accuracies under several multi-view approaches.

4.2 Background

Multi-view machine learning algorithms represent instances using multiple distinct features sets or views [100]. The relationship between views can be used to align the feature spaces using methods such as Canonical Correlation Analysis [46] or Manifold Alignment [114]. Alternatively, multiple classifiers can be trained for each view and the labels can be propagated between views as in Co-Training [9] or Co-EM [70].

Multi-view learning has also been used as a heterogeneous transfer learning technique [78] that applies knowledge learned from a previous domain and task to a new, related domain and task.

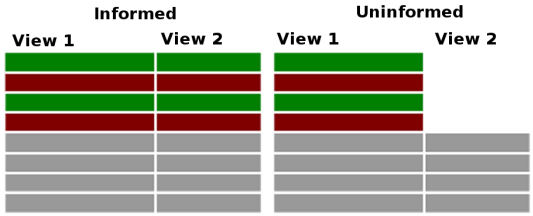


Figure 5: Multi-view transfer learning. The dark colored bands represent labeled data while the light gray bands represent unlabeled data. In informed multi-view transfer learning both views have some labeled data and a large amount of unlabeled data. In uninformed multi-view transfer learning only the source view (View 1) has labeled data but both views have unlabeled data.

One way to categorize transfer learning approaches is based upon the availability of labeled data. The terms *supervised* and *unsupervised* learning refer to the availability of labeled data in the source space (view 1), while the terms *informed* and *uninformed* learning refer to the availability of labeled data in the target space (view 2). Figure 5 illustrates the difference between informed supervised transfer learning and uninformed supervised transfer learning for multi-view learning [21]. In this figure, colored bands represent the labeled activities while the light gray bands represent unlabeled activities.

We investigate several multi-view learning techniques to transfer knowledge between different views. We consider both informed and uninformed techniques and we analyze the trade-off of each technique. We also propose the use of a teacher-learner technique for this problem [53]. We discuss the relationship between this and other multi-view techniques. In addition, we derive tighter estimates and bounds of the learner accuracy and highlight the applicability of Probably Approximately Correct (PAC) bounds to the teacher-learner technique.

4.3 Methods

To illustrate our approach, consider a scenario in which a home has been equipped with multiple sensors to monitor motion, temperature, and door open/closures. Sensor data is collected, annotated with ground truth activity labels, and used to train an activity classifier. The resident recently purchased a smart phone and wants to train the phone sensors to recognize the same activities. This way the phone can continue to monitor activities that are performed out in the community and can update the original model when the resident returns home. Whenever the smart phone is located inside the smart home, both sensing platforms will collect data while activities are performed, resulting in a multi-view problem where smart home sensor data represents one view and smart phone sensor data represents a second view. Working from this example, we now described our proposed approaches.

4.3.1 Informed Multi-view Learning

Co-Training and Co-EM represent informed supervised learning techniques. Co-Training is one of the first developed multi-view learning techniques [9]. In Co-Training, a small amount of labeled data in each view is used to train two classifiers, one for each view. These classifiers then assign labels to a subset of the unlabeled data. The newly-labeled instances are added to the set of labeled data and the process is repeated. A simple method to extend this approach to multiple views is to add additional classifiers, one for each view. Our Co-Training approach, adapted from Blum and Mitchell [9], is summarized in Algorithm 4. This algorithm is described for the binary classification task but easily extends to k-ary classification problems by allowing each classifier to

Algorithm 4: Co-Training Algorithm

Data: a set L of labeled training examples
Data: a set U of unlabeled training examples
Create a pool U' of u examples, $U' \subseteq U$;
while $U' \neq \emptyset$ **do**
    Use L to train classifier h_1 for view 1;
    Use L to train classifier h_2 for view 2;
    ...;
    Use L to train classifier h_k for view k;
    Label most confident p positive examples and n negative examples from U' using h_1;
    Label most confident p positive examples and n negative examples from U' using h_2;
    ...;
    Label most confident p positive examples and n negative examples from U' using h_k;
    Add the self-labeled examples to L;
    Replenish U' using $k * p + k * n$ examples from U;

label n positive examples for each class instead of labeling p positive examples and n negative examples.

Co-EM is a variant of Co-Training that has been shown to perform better in certain situations [70]. Unlike Co-Training, Co-EM labels the entire set of unlabeled data every iteration. Again, a simple method to extend this approach to multiple views is to add additional classifiers, one for each view. The Co-EM algorithm is summarized in Algorithm 5. Convergence can be measured here as the number of labels that change each iteration. Alternatively, a fixed number of iterations can be specified.

4.3.2 Uninformed Multi-view Learning

Manifold Alignment has been proposed as a technique for transferring knowledge between two different views without requiring any labeled data [114]. It assumes that

Algorithm 5: Co-EM Algorithm

Data: a set L of labeled training examples
Data: a set U of unlabeled training examples
Use L to train classifier h_1 on view 1;
Create a set U_1 by using h_1 to label U;
for $i \leftarrow 0$ **to** n **do**
    Use $L \cup U_1$ to train a classifier h_2 on view 2;
    Create a set U_2 by using h_2 to label U;
    Use $L \cup U_2$ to train a classifier h_3 on view 1;
    ...;
    Use $L \cup U_{k-1}$ to train a classifier h_k on view k;
    Create a set U_1 by using h_k to label U;

the data from both views share a common latent manifold which exists in a lower-dimensional subspace. The basic idea is that the two feature spaces can be projected onto a lower-dimensional subspace and the pairing between views can then be used to optimally align the subspace projections onto the latent manifold. A classifier can then be trained using projected data from the source view and tested on projected data from the target view. The details are shown in Algorithm 6.

Algorithm 6: Manifold Alignment Algorithm

Data: a set L of labeled training examples in view 1
Data: a set U_1 and U_2 of paired unlabeled training examples, one for each view
$X, EV \leftarrow$ `PCA(`U_1`)` ;
$Y \leftarrow$ `PCA(`U_2`)` ;
`// Apply Procrustese Analysis` $U\Sigma V^T \leftarrow$`SVD(`$Y^T X$`)`;
$Q \leftarrow UV^T$;
$k \leftarrow$ `Trace(`Σ`)/Trace(`$Y^T Y$`)`;
$Y' \leftarrow kYQ$;
Project L onto low-dimensional embedding using EV;
Train classifier on projected L;
Test classifier on Y';

The final existing method we consider is a teacher-learner model that was introduced by Kurz et al. to train new sensor systems using the output of existing sensor systems

[53]. The details of the method are shown in Algorithm 7. The approach is ideal when a new sensor (or set of sensors) is added to an existing system. This creates a natural setting where labeled data is available in the source view (i.e. the existing sensors) but not in the target view (i.e. the new sensors). When activities of interest occur, the existing system can generate the activity label and share that label with the new system.

We note that the teacher-learner algorithm is equivalent to a single-iteration version of Co-EM when no labeled data is available in the target view. Recognizing the teacher-learner model as a variation of multi-view learning allows us to provide a stronger theoretical foundation for the technique. Valiant introduces the framework of Probably Approximately Correct (PAC) learning which provide bounds on the probability that the selected function will how a low generalization error [106]. Blum and Mitchell show that multi-view learning has PAC bounds if the target concept is learnable from random classification noise in the standard PAC model [9]. Specifically, they prove that given three assumptions the PAC bounds hold for learning in the second view. The assumptions are: 1) the two views are conditionally independent given the class label, 2) either view is sufficient to correctly classify the examples, and 3) the accuracy of the first view is at least weakly useful. Thus under these assumptions we are guaranteed that the resulting classifier for the target view will have PAC bounds.

In addition to the simple extensions to these algorithms, which allow for more than two views to be used, we also consider another technique for incorporating multiple views. We refer to this technique as Personalize ECOsystems with Ensembles (PECO-E). In this approach, multiple views are first combined into a single view using ensemble methods so that only two views are present. The Co-Training, Co-EM and Teacher-Learner algorithms can then be applied on these two views. In this work, we use a

Algorithm 7: Teacher-Learner Algorithm

Data: a set L of labeled training examples in view 1
Data: a set U of unlabeled training examples
Use L to train a classifier h_1 on view 1;
Create a set U_1 by using h_1 to label U;
Use U_1 to train a classifier h_2 on view 2;
Use U_1 to train a classifier h_3 on view 3;
...;
Use U_1 to train a classifier h_k on view k;

Algorithm 8: Personalized Ecosystem Algorithm

Data: a set L of labeled training examples in view 1
Data: a set U of unlabeled training examples
Use L to train a classifier h_1 on view 1;
Create a set U_1 by using h_1 to label $U' \subset U$;
$L \rightarrow L \cup U_1$;
$U \rightarrow U - U_1$;
Apply Algorithm 4 or 5

weighted voting ensemble where a classifier from each view votes for multiple class labels. Each vote is weighted by the classifier's confidence in the classification label.

We also propose a new algorithm Personalized ECOsystem (PECO) which is a combination of the teacher-learner algorithm and an informed transfer learning technique such as Co-Training or Co-EM. The pseudo-code is shown in Algorithm 8. We hypothesize that such a combination will increase the accuracy of the learner without requiring that any labeled data be available to the learner. Initially, the teacher provides a few labels to the learner using Algorithm 7. Then we transition to an iterative model by subsequently applying either Algorithm 4 or Algorithm 5. In this way, the learner can continue to benefit from the expertise of the teacher while at the same time contributing back its own expertise.

In our home-phone scenario, the smart home may initially act as a teacher because it

has labeled activity data. When the home and phone occupy the same space the home can opportunistically "call out" activity labels in situations when the home and phone both observe the resident performing an activity. The resident may leave the home, taking his phone with him. While out, the phone will observe new activity situations and possibly receive activity labels for those situations. When the individual returns home with the phone, the home and phone can now act as colleagues, providing expertise from each classifier to improve the robustness of the individual activity models on each sensor platform.

4.4 Accuracy Bounds

Without labeled data in the target view, we cannot directly compute empirical performance measures such as model accuracy. We can, however, still compute bounds for the expected-case, worst-case, and best-case performance of the learner. To do so we make the assumption that the previously observed accuracy of the teacher on the labeled data is a good predictor of the accuracy of the teacher on the unlabeled data and that we know the level of agreement between the teacher and the learner. We define level of agreement between the teacher and the learner in Equation 4.1 where N is the number of unlabeled instances, $h_1(x)$ is the teacher's prediction for x, and $h_2(x)$ is the learner's prediction for x.

$$q = \frac{1}{N} \sum_{x \in U} \begin{cases} 1 & \text{if } h_1(x) = h_2(x) \\ 0 & \text{if } h_1(x) \neq h_2(x) \end{cases} \tag{4.1}$$

For binary classification tasks, if p represents the accuracy of the teacher then Equation 4.2 is the expected accuracy of the learner under the assumption that the agreement between the teacher and the learner is independent of whether or not the teacher correctly predicts the class label.

$$\begin{aligned} r &= pq + (1-p)(1-q) \\ &= 2pq + (1-q-p) \end{aligned} \tag{4.2}$$

The first term, pq, represents the expected accuracy of the learner given that the teacher correctly classified the instance. The second term represents the expected accuracy of the learner given that the teacher incorrectly classified the instance. While the first term is straightforward, the second term requires explanation. First, every disagreement between the teacher and the learner on the instances that the teacher misclassifies results in a correct classification for the learner since this is a binary classification task. $(1-p)$ represents the inaccuracy of the teacher and $(1-q)$ represents the level of disagreement between the teacher and the learner. Thus, $(1-p)(1-q)$ represents the expected accuracy of the learner given that the teacher misclassified the instance.

To prove best-case and worst-case learner accuracy bounds, we split the level of teacher-learner agreement into two parts. The term q_1 represents teacher-learner agreement on instances that the teacher correctly classifies and q_2 represents agreement on instances that the teacher incorrectly classifies. Substituting these values into Equation 4.2 results in Equation 4.3. The resulting accuracy is optimized by maximizing q_1 and minimizing q_2.

$$r = q_1 * p + (1 - p)(1 - q_2) \tag{4.3}$$

In this discussion, q_1 and q_2 are subject to the following constraints: 1) $q_1 * p + q_2 * (1 - p) = q$, 2) $0 \leq q_1 \leq 1$, and 3) $0 \leq q_2 \leq 1$. These constraints ensure that the original level of agreement q is preserved and that q_1 and q_2 are valid levels of agreement. Note that in the first constraint, minimizing q_2, maximizes q_1 and vice versa. If $p \geq q$ then q_2 has a minimum value of 0 and all the constraints are satisfied. This implies $q_1 = q/p$ is the maximum value of q_1. Substituting into Equation 4.3 leads to Equation 4.4.

$$r = q + 1 - p \tag{4.4}$$

If $p < q$ then q_1 has a maximal value of 1 and all of the constraints are satisfied. This implies that $q_2 = (q - p)/(1 - p)$ is the minimum value of q_2. Substituting into Equation 4.3 leads to Equation 4.5

$$r = p + 1 - q \tag{4.5}$$

Finally, Equations 4.4 and 4.5 are unified into a single equation which represents the upper bound of the learner accuracy, as shown in Equation 4.6.

$$r = 1 - |p - q| \tag{4.6}$$

The lower bound on the learner accuracy is found by minimizing q_1 and maximizing q_2 in Equation 4.3 subject to the same constraints on q_1 and q_2. If $(1 - p) \geq q$ then q_1 has a minimum value of 0 and all the constraints are satisfied. This implies that

$q_2 = q/(1-p)$ is the maximum value of q_2. Substituting into Equation 4.3 leads to Equation 4.7

$$r = 1 - p - q \tag{4.7}$$

If $(1-p) < q$ then q_2 has a maximal value of 1 and all of the constraints are satisfied. This implies that $q_1 = (q - 1 + p)/p$ is the minimum value of q_1. Substituting into Equation 4.3 leads to Equation 4.8.

$$r = q - 1 + p \tag{4.8}$$

Finally, Equations 4.7 and 4.8 are unified into a single equation which calculates the lower-bound of the accuracy of the learner in Equation 4.9.

$$r = |1 - p - q| \tag{4.9}$$

Note that these bounds can be extended to the k-ary classification problem. The upper bound remains the same. The lower bound becomes 0 if $(1-p) \geq q$ and stays the same if $(1-p) < q$. To compute the expected bounds we first note that Equation 4.2 could have an additional term z which is the probability that the learner correctly classifies an instance given that the teacher misclassified the instance and that the learner disagrees with the teacher on that instance. This leads to Equation 4.10.

$$r = pq + (1-p)(1-q)z \tag{4.10}$$

For binary classification, $z = 1$ and can therefore be ignored. For k-ary classification however, $z <= 1$. We propose two different estimates for z. The first estimate uses the

number of classes without considering the distribution of the class labels $z = 1/(k-1)$. The second estimate makes direct use of the distribution of class labels and is shown in Equation 4.11 where $P(y)$ is the probability that an instance has a class label of y and Y represents the set of class labels. $P(y)$ can be estimated using the observed frequency of each class label.

$$\begin{aligned} z &= \sum_{x \in Y} P(x) \sum_{y \neq x \in Y} \frac{P(y)P(y)}{(1-P(x))(1-P(x))} \\ &= \sum_{x \in Y} \frac{P(x)}{(1-P(x))^2} \sum_{y \neq x \in Y} P(y)^2 \end{aligned} \tag{4.11}$$

The intuitive explanation of z here is that z is the probability the teacher assigns a class label of x times the probability that y is the true class label times the probability the learner selects the correct class label of y all of which is summed over each possible class label and normalized by the the remaining probabilities given that x is not the class label.

We also consider another estimation for the expected accuracy of the learner. Rather than assuming that the agreement between the teacher and the learner is equivalent for all class labels we instead estimate the following conditional distributions: $\alpha = P(h_1() = x | f_1() = y)$, the probability that the teacher classifies an instance as x given that the instance has a true class label (activity label) of y and $\beta = P(h_2() = y | h_1() = x)$, the probability that the learner classifies an instance as y given that the teacher classified the instance as x. Both of these probabilities can be estimated without using any labeled data in the second view. The expected bound is then given by Equation 4.12. In this case we know longer explicitly distinguish between the teacher being right and the teacher being wrong. Instead, it is handled implicitly by $y = x$ and $y \neq x$.

$$r = \sum_{y \in Y} P(y) \sum_{x \in Y} \alpha\beta \tag{4.12}$$

In addition to the previously proposed estimates of the accuracy of the learner, we also consider the average of the upper and lower bounds. This method avoids calculating class distributions and conditional probabilities. It also avoids explicit assumptions about the probability of the teacher and learner agreeing. Instead, the assumptions is that the learner is unlikely to happen to maximize or minimize Equation 4.3 but will instead fall in the middle of these two extremes. Interestingly, when $p \geq q$ and $(1-p) < q$ then the average of the upper and lower bounds is just q.

Finally, the expected accuracy of the learner can be underestimated but simplified to $r = pq$ which is the value used by Kurz et al. [53]. These expected, best and worst-case bounds provide insight on expected performance of the learner by determining the maximum, minimum and likely actual accuracy of the newly trained system. The accuracy bounds are computed without needing to obtain labeled data in the target view. Using labeled data in the source view, the teacher accuracy can be estimated. Then, using unlabeled data in both the source and target view, the agreement between the teacher and the learner can be computed. Finally, the accuracy bounds can be computed using these two values.

4.5 Conclusions

The new sensing platform problem is encountered every time a new sensing platform is installed into an environment with a pre-existing sensing platform. When no pre-existing sensing platform is present, we instead treat it as a new environment problem. The presence of multiple sensing platforms allows us to frame the new sensing platform problem as a multi-view transfer learning problem. We have adapted several multi-view algorithms to this situation. When labeled data is available for each sensing platform or view, we can use the informed multi-view approaches such as co-training or co-expectation maximization. When no labeled data is available for the new sensing platform, the uninformed multi-view algorithms such as teacher-learner and manifold alignment can be applied. We have also developed two novel algorithms PECO and PECO-E. PECO-E can be applied to either informed or uninformed transfer learning while PECO is specifically for the uninformed transfer learning scenario.

When the uninformed transfer learning techniques are applied to the new sensing platform problem, we have no way of directly computing the accuracy of the newly trained system since we do not have any manually annotated training data. Instead, we have developed upper and lower bounds on the accuracy of the newly trained system given the accuracy of the pre-existing system and the amount of agreement between the systems. This allows us to train a new activity recognition system and estimate the accuracy of that system without any human intervention to provide labeled training examples.

Chapter 5

Results

5.1 FSR Experimental Results

FSR and its proposed extensions can be applied to a variety of different transfer learning problems. We primarily evaluate the performance of these techniques in the activity recognition domain and, for generalizability, we also show some results in the document classification domain.

5.1.1 Activity Recognition

We use a dataset consisting of data from 18 different smart apartments. The apartments are single residence assisted-living care facilities. Specific statistics for each apartment are found in Table 10. Each apartment is equipped with motion sensors and door sensors. The number of sensors range from 17 to 39 with an average of 28.7 sensors and a standard deviation of 6.21. The layout for the apartments is shown in Figure 6. Each dataset has been annotated with 37 different activities, shown in Table 11, with the total amount of labeled data spanning one month of time per dataset. Not all apartments have all 37 activity labels as indicated in the table. We consider all possible combinations of source and target datasets, yielding a total of 306 possible pairings. We use a single day of labeled data for the target domain and all 30 days of labeled data for the source domain. This data is event-based so we use the event-based feature representation (see Section

2.2.1) and set the window size k to 10.

Table 10: Summary statistics of the activity recognition dataset

Id	# Features	# Labels	# Instances	# USFSR Meta-Features	# ISFSR Meta-Features
1	35	29	133157	1575	1295
2	17	26	53669	765	629
3	37	31	178137	1665	1369
4	29	29	57918	1305	1073
5	39	32	141181	1755	1443
6	26	32	149391	1170	962
7	26	30	183945	1170	962
8	26	28	98768	1170	962
9	34	30	102466	1530	1258
10	24	30	143145	1080	888
11	38	30	157736	1710	1406
12	24	29	135451	1080	888
13	32	32	116641	1440	1184
14	26	31	195611	1170	962
15	23	29	100255	1035	851
16	33	32	179693	1485	1221
17	23	29	92740	1035	851
18	24	30	117067	1080	888

Fitness Function

Before comparing the FSR techniques against each other and other baseline techniques, we first consider the effect of the choice of fitness function on overall performance of GAFSR. Performance is measured using both the accuracy (given by Equation 3.2) and the unweighted average recall (given by Equation 3.1). We report both the accuracy and the recall because accuracy scores are biased towards the majority class. For balanced class distributions this has little effect on the metric, but it may not be suitable for unbalanced class distributions. Using the unweighted average recall eliminates this bias and treats all classes equally [107]. Note that accuracy can also be considered as the

(a) Apt. 1 (b) Apt. 2 (c) Apt. 3 (d) Apt. 4

(e) Apt. 5 (f) Apt. 6 (g) Apt. 7 (h) Apt. 8

(i) Apt. 9 (j) Apt. 10 (k) Apt. 11 (l) Apt. 12

(m) Apt. 13 (n) Apt. 14 (o) Apt. 15 (p) Apt. 16

(q) Apt. 17 (r) Apt. 18

Figure 6: Apartment layouts from the activity recognition dataset

Table 11: List of activities and the relative frequency of occurrence of each activity

Activity	**Frequency**	**Activity**	**Frequency**
Enter Home	0.0031	Personal Hygiene	0.0545
Eat Lunch	0.0070	Leave Home	0.0026
Cook Dinner	0.0534	Eat Dinner	0.0100
Exercise	0.0002	Cook Lunch	0.0274
Wash Dinner Dishes	0.0127	Relax	0.0191
Read	0.0103	Wash Lunch Dishes	0.0077
Phone	0.0029	Evening Meds	0.0037
Eat Breakfast	0.0101	Watch TV	0.0405
Cook	0.0348	Wash Breakfast Dishes	0.0126
Eat	0.0066	Groom	0.0087
Housekeeping	0.0113	Toilet	0.0434
Wash Dishes	0.0088	Work At Desk	0.0004
Sleep Out Of Bed	0.0034	Work At Table	0.0253
Morning Meds	0.0053	Cook Breakfast	0.0320
Take Medicine	0.0036	Bed Toilet Transition	0.0156
Bathe	0.0175	Work	0.0329
Other Activity	0.2789	Entertain Guests	0.0837
Sleep	0.0407	Work On Computer	0.0498
Dress	0.0194		

average recall weighted by the number of instances in the class. Throughout the rest of this discussion, recall will refer to the unweighted average recall.

Figure 7 shows the accuracy and recall results of the two different fitness functions for GAFSR averaged over all 306 pairings. In this case we use the full 30 days of labeled data in both the source and target domain as we are interested only in the relative performance difference between the two fitness functions. As can be seen in the figure, including the overall accuracy in the fitness function improves the accuracy with only a slight drop in the recall. We conducted a student's t-test and found that the difference in accuracy is significant ($p < 0.05$) while the difference in recall scores is not ($p = 0.13$).

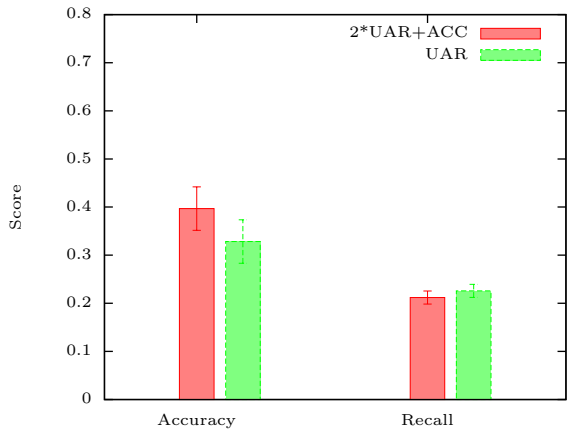


Figure 7: Comparison of fitness functions for GAFSR. The average accuracy and recall scores over all 306 source-target pairings are shown. Including the accuracy score in the fitness function improves accuracy without degrading the average recall.

Baseline Comparisons

We are now ready to compare the three proposed techniques, GAFSR, GrFSR, and ISFSR, against several other baselines. This comparison is used to meet Objective 1.1. GAFSR and GrFSR use the fitness function specified in Equation 3.4. ISFSR uses the feature-label co-occurrence meta-features as described in Equation 3.9. The first baseline, *Manual*, uses the generalized sensor locations (kitchen, bedroom, etc) to map sensors from one apartment to another. The second baseline *None* classifier treats all sensor events as coming from a single source. Essentially this eliminates the sensor dimension and only considers the time of day and day of week of the activity. The *Manual* technique is the mapping technique currently used by most researchers in activity recognition [21, 91, 107]. It does not require any labeled data in the target domain, but it does require the manual definition of sensor locations. On the other hand, *None* provides a lower bound on the expected performance. The last baseline we consider, *Self* is a classifier trained and tested in the target domain. All of the techniques

use a naïve Bayes classifier trained on the source domain and tested on the target domain. We considered other base classification algorithms such as SVMs, Decision Trees and Nearest Neighbors. However, since the meta-features used in ISFSR are specifically related to naïve Bayes classification we have found that it gives good results without the computational overhead of some of the other methods. For comparison purposes, we also include results for ISFSR when a decision tree has been used as the base classification method.

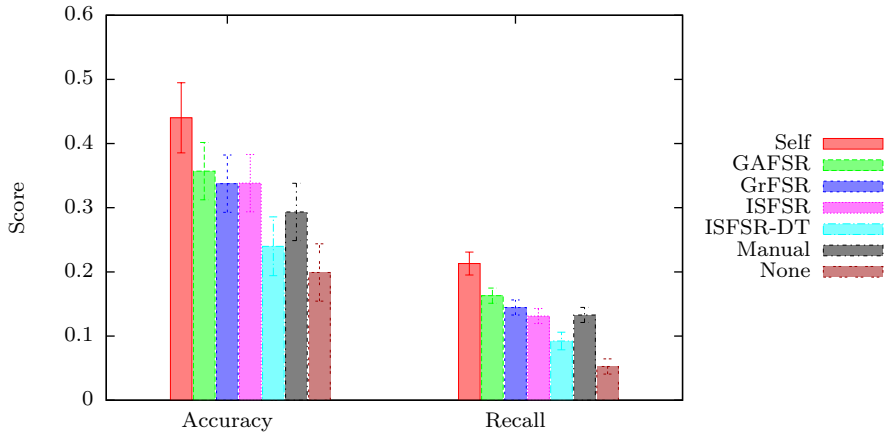


Figure 8: Classification accuracy and recall on the target domain using a single source domain. *Manual* and *None* provide baseline comparisons. *Manual* is the mapping specified by a domain expert. *None* does not apply any mapping at all. GAFSR, GrFSR and ISFSR are all able to perform as good or better than the *Manual* technique. The performance of GAFSR, GrFSR, and ISFSR is ordered by the computational complexity of each technique, highlighting the benefit of exploring the mapping space at the cost of increased running times.

The results are shown in Figure 8. A one-way ANOVA is performed and the resulting p-value is less than .0001. The 95% confidence interval is depicted with the error bars. All three FSR techniques match or beat the two baselines of *Manual* and *None*. As the amount of time spent exploring or computing a good mapping between the target and source domains increases the resulting accuracy and recall scores also increases. GAFSR

achieves the best performance scores but it also requires the most time to run, while ISFSR uses the fewest number of computations but also has the lowest performance scores of the three techniques. Note that the performance gap between ISFSR and GrFSR is much smaller than the gap between GAFSR and GrFSR.

Matching the performance of the *Manual* mapped technique is a positive result as it implies that transfer learning can be used to reduce or eliminate the need for a domain expert to supply a mapping between domains. The FSR mapping is able to outperform the *Manual* mapping technique because the manual mapping technique is based solely upon the location of the sensors. This is effective when the resident in the source dataset performs activities in the same locations as the resident in the target dataset. For example, both residents are likely to cook in the kitchen. On the other hand, the manual mapping technique is likely to fail when the residents perform the same activity in different locations. For example, the resident in the source dataset might eat in the living room while the resident in the target dataset might eat in the kitchen. FSR overcomes this problem by mapping features based on correlation with the activity label. The meta-features used by ISFSR are specifically derived to optimize the mapping when a naïve Bayes classifier is used. However, from the performance of ISFSR-DT we see that the mapping works with other classifiers as well. Exploring other mapping strategies and heuristics may lead to further improvements for specific types of classifiers.

All of the techniques exhibit relatively low accuracy and recall scores. This is due to several factors. First, the activity recognition is done in a streaming (un-segmented) fashion. Second, there is a large number of activity classes many of which are similar or partially overlapping in nature (ex. Cooking and Cooking Breakfast). Improving the baseline activity recognition rates is not the focus of this dissertation, but others have

made significant contributions in this direction [22].

In addition to considering the accuracy and recall scores, we can also look at the ROC curve which plots the true positive rate vs the false positive rate. We generate the ROC curve by looking at each class label in a one-vs-all scenario. The instances are sorted by the probability estimate of the classifier and the true positive and false positive rates for that class are then calculated. Appendix A shows the ROC curves for each individual class. Finally, we average the ROC curve over all the classes to obtain the results shown in Figure 9. The Self classifier has the best ROC curve. The ROC curve for Manual and ISFSR are both similar with ISFSR not quite matching that of the Manual technique. GAFSR and GrFSR both have the similar ROC curves but neither one performs as well as ISFSR, Manual or Self. The swapping of performance results compared to the previous metrics can be explained from the one-vs-all nature of the ROC curve. This essentially allows the classifier to assign multiple class labels to a single instance instead of forcing the classifier to pick a single class label. However, GAFSR and GrFSR do not take advantage of this since they have been optimized with the accuracy and recall metrics in mind. Adding a metric such as the Area Under the ROC curve (AUC Score) to the fitness function may improve the ROC curve performance of GAFSR and GrFSR.

The previously-discussed results are the average of 306 different mappings. Individual results show both higher and lower performance. One direction of transfer learning research focuses on how to select the best source dataset. Assuming this problem is solved then we could select the "best" source dataset for each target dataset. We do not claim that this contributes to avoid negative transfer, only that if negative transfer can be predicted and avoided we can improve the results. Figure 10 shows the results of using the best source dataset with the same mapping techniques discussed earlier. Under this

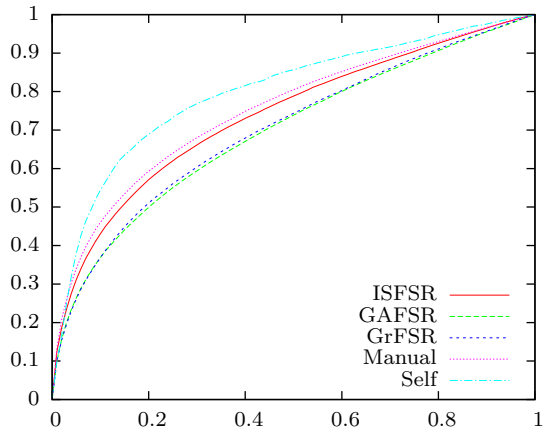


Figure 9: FSR ROC curve averaged over all classes. The ISFSR and Manual show similar performance to one another and do not match the performance of Self. GAFSR and GrFSR also show similar performance to one another and have the lowest ROC curves.

scenario, the accuracy scores of the three techniques are nearly equivalent with ISFSR actually performing the best. The recall scores of the three techniques continue to be ordered by the computational complexity of the technique. Again all three techniques are able to outperform the baseline techniques of *Manual* and *None* but this time they even match or beat the performance of *Self*. A one-way ANOVA is performed and the resulting p-value is less than .0005. The 95% confidence interval is depicted with the error bars.

We next compare USFSR against the same baselines. USFSR uses the meta-features described in Table 9. As can be seen in Figure 11, the USFSR algorithm performs reasonably well if only the accuracy score is considered. Its performance nearly matches that of the manual technique. However, when the recall score is considered, USFSR performance drops significantly. USFSR operates with significantly less information than the informed transfer learning techniques because it does not have any labeled

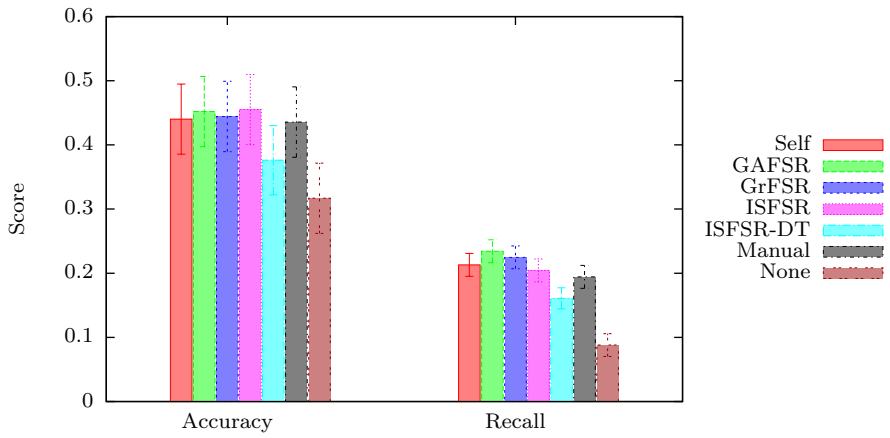


Figure 10: Classification accuracy and recall on the target domain using the best single source domain. This assumes that the best dataset to transfer from could be identified a priori. *Manual* and *None* provide baseline comparisons. *Manual* is the mapping specified by a domain expert. *None* does not apply any mapping at all. A one-way ANOVA is performed and the resulting p-value is less than .0005. The 95% confidence interval is depicted with the error bars.

data in the target domain. In this case, making meaningful mappings between domains becomes extremely challenging.

Figure 12 shows the results of using the best source dataset with the same mapping techniques discussed earlier. In this case, USFSR is no better than the *None* baseline and neither technique performs as well as the manual mapping.

Ensemble Learning

Next, we consider different techniques which utilize data from multiple source datasets. These experiments meet Objective 1.2 in showing the effectiveness of FSR to combine multiple source datasets via ensemble learning. We compare against the following techniques. *Self* uses a naïve Bayes classifier which has been trained on the full amount of labeled target data using 3-fold cross-validation. *Combined* combines all of the source

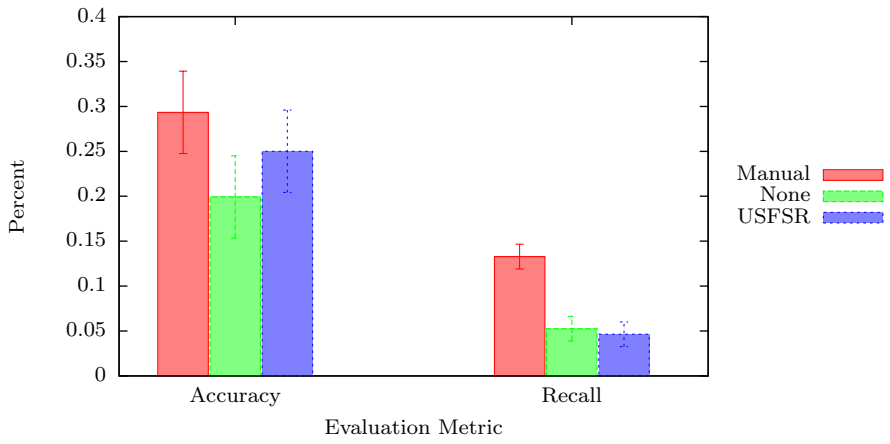


Figure 11: Classification accuracy on the target domain using a single source domain. *Manual*, and *None* both provide baseline comparisons. *Manual* is the mapping specified by a domain expert. *None* does not apply any mapping at all. USFSR does not have enough information to make an effective mapping.

domain data into one big dataset with sensor mappings being manually defined by location. A naïve Bayes classifier is trained on all of the source data and then tested on the target data. The ensemble techniques each train one naïve Bayes classifier per source dataset and the ensemble is then tested on the target domain. As in the previous experiments only one day of labeled target data is used by ISFSR to make the mapping.

Figure 13 shows the results using the voting ensemble techniques while Figure 14 shows the results using the stacking ensemble techniques. In neither case do we attempt to select the best source datasets we simply use all available source dataset.

Again we use the accuracy and unweighted average recall for performance metrics. The performance of the voting ensembles is mixed. USFSR is still unable to compete with the techniques which use more information (labeled data or manual mappings). The ISFSR voting ensembles perform comparably to the combined dataset. The trade-off is that the combined dataset requires a manually-mapped specification while the

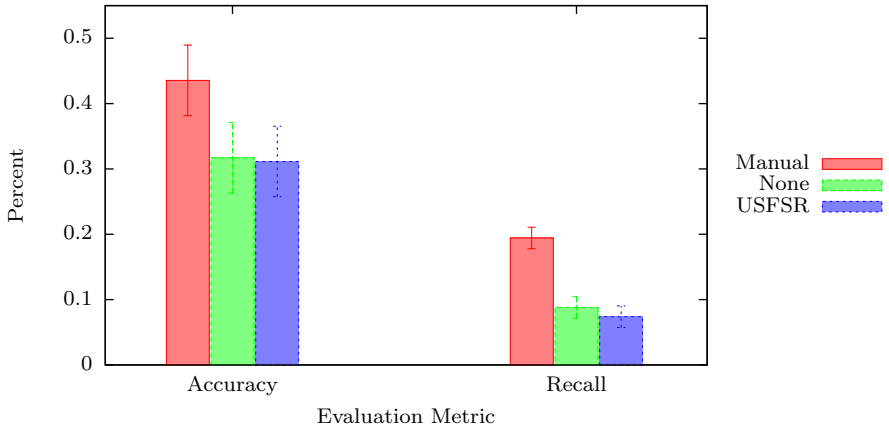


Figure 12: Classification accuracy on the target domain using the best single source domain. This assumes that the best dataset to transfer from could be identified a priori. *Manual*, and *None* both provide baseline comparisons. *Manual* is the mapping specified by a domain expert. *None* does not apply any mapping at all.

ISFSR voting ensembles require a small amount of labeled data in the target domain. Neither ISFSR nor *Combined* performs as well as a classifier trained and tested only on labeled target data again indicating that additional domain adaption technique may be beneficial.

The performance of the stacking ensembles stand out above the rest. Both stacking ensembles achieve higher performance in terms of the accuracy and recall scores than the combined dataset or the *Self* classifier. It does this using only a single day's worth of labeled data and no manual mapping is required. The *Self* approach uses nearly 30 days of labeled data and is trained and tested on the same dataset (with cross-validation), while the Combined approach uses no labeled data in the target domain but requires a manual mapping to be specified.

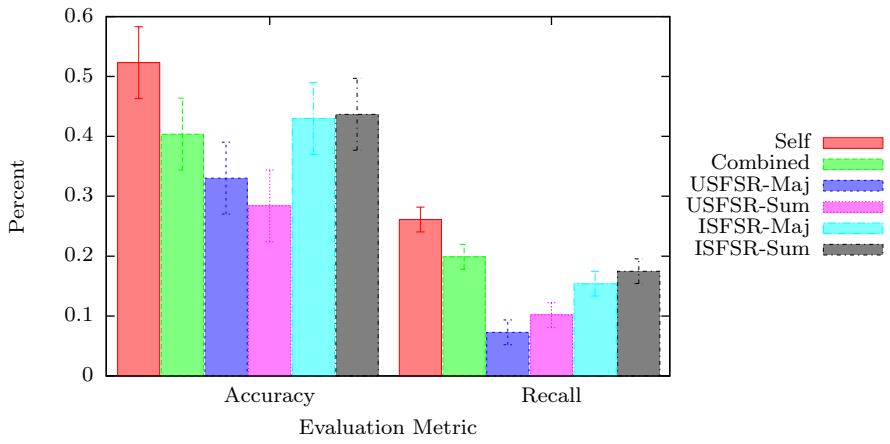


Figure 13: Classification accuracy on the target domain using multiple source domains with a voting ensemble. *Self* and *Combined* provide baseline comparisons. *Self* is the result when the source and target dataset are the same and uses the all the labeled target data, while *Combined* uses the mappings provided by a domain expert to build a generic classifier. Matching the performance of *Combined* is a positive result.

Learning Curve

Now, we look at two different types of learning curves. The first one considers how the accuracy and recall scores change as the amount of labeled data increases. The second one considers how the accuracy and recall scores change as the number of source datasets increases.

This first experiment shows the effect of the amount of labeled target data on the accuracy and recall score of the ISFSR algorithm. As in the previous experiments, we use the 306 possible pairings of the activity recognition datasets. However, this time we vary the number of days of labeled target data from 0.25 to 30. We also include a comparison to a baseline classifier, *Self*, which uses a naïve Bayes classifier which has been trained only on the labeled target data and is tested on the remaining target data. Figure 15 shows the results. Clearly, adding more labeled target data is initially beneficial. However, for ISFSR, the increase in accuracy begins to level off

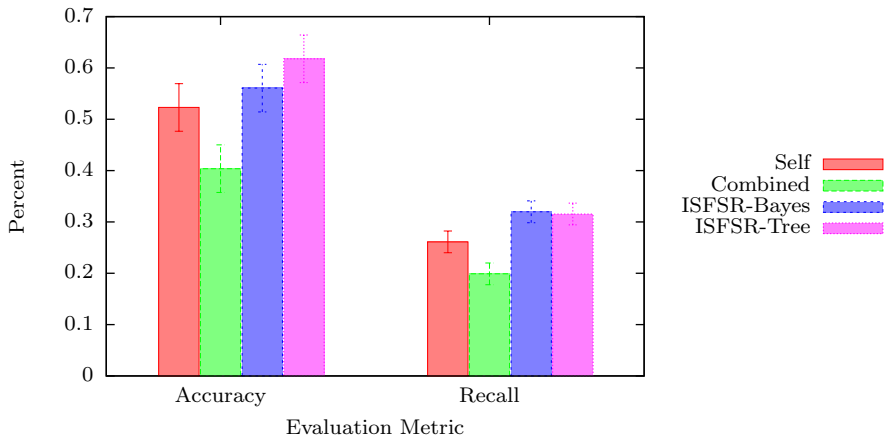


Figure 14: Classification accuracy on the target domain using multiple source domains with stacking ensembles. *Self* and *Combined* provide baseline comparisons. *Self* is the result when the source and target dataset are the same and uses the all the labeled target data, while *Combined* uses the mappings provided by a domain expert to build a generic classifier. The performance of ISFSR-Bayes and ISFSR-Tree both manage to beat these baselines representing a considerable gain for the transfer learning techniques.

after approximately ten days of labeled target data. The increase in recall appears to peek between five and ten days of labeled target data after which point the recall score declines slightly. This may indicate that having too much labeled data causes ISFSR to over-fit the data. Comparing ISFSR against the baseline *Self* we see that initially ISFSR is able to outperform the baseline. As the amount of labeled data exceeds one day though, *Self* begins to outperform ISFSR.

The second experiment shows the effect of changing the number of datasets used in the ensemble learning. Figure 16 shows the learning curve for each ensemble technique as the number of source datasets increases. For USFSR-Summation, ISFSR-Summation, ISFSR-Bayes, and ISFSR-Tree, the performance increases with an increasing number of datasets. Most of the improvement is achieved within the first seven datasets, after which performance improvement tapers off. For USFSR-Majority and ISFSR-Majority,

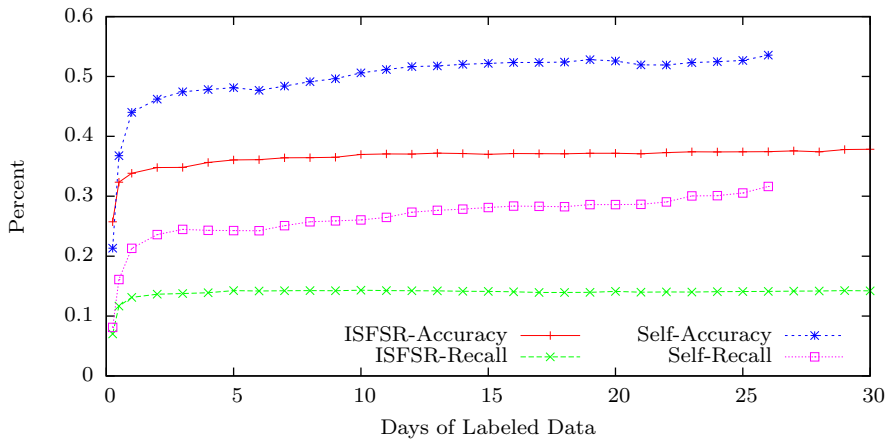


Figure 15: ISFSR and *Self* accuracy and recall scores as the amount of labeled target data increases. Accuracy continues to show improvement with the increase of labeled target data while the recall score peeks at five and ten days of labeled data in the target domain.

the accuracy performance improves with an increasing number of datasets, but the recall performance remains almost constant regardless of the number of datasets. This illustrates the fact that important distinguishing information is being discarded by the majority voting scheme.

5.1.2 Document Classification

Objective 1.3 seeks to show the generalizability of FSR to other domains. In this section we apply FSR to several scenarios involving document classification. We test ISFSR on the newsgroups dataset [57]. The newsgroups dataset is a collection of approximately 20,000 documents across 20 different topics. The topics are organized in a hierarchical manner. Following the processing steps used by Dai et al. [25] and Pan et al. [73], the source and target datasets are created by first selecting two top-level categories as the class labels. The documents are then split by sub-categories to form a source dataset

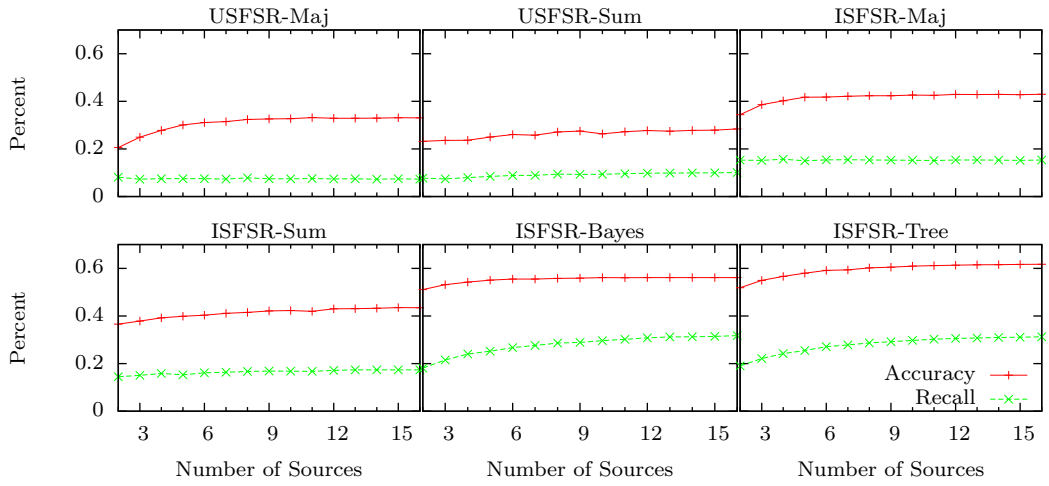


Figure 16: Learning curve for the ensemble classifiers where the number of source classifiers ranges from 2 to 16. Each ensemble technique quickly improves with more source classifiers but the performance improvements then begin to level off.

and a target dataset. The resulting datasets are shown in Table 12. We also show basic statistics about the datasets in Table 13.

Each pair of datasets is processed separately so that the alignment and number of attributes is the same for datasets in the same row but different for datasets in different rows (i.e. the feature-space of D_s sci. vs. talk is the same as the feature space of D_t sci. vs. talk but the feature-space of D_s rec. vs. sci. is not the same as the feature space of D_t sci. vs. talk. Additionally, the source distribution is different from the target distribution for all datasets because the documents come from different sub-categories; however, they are still related because they come from the same top-level categories.

As in the work of Dai et al. [25] and Pan et al. [73] we train ISFSR on D_s and then test ISFSR on D_t for each row in the table. This is not a heterogeneous transfer learning problem, but rather a domain adaptation problem. However, we also take the transfer learning problem one step further and test each D_t on classifiers trained on the D_s of the other rows. This creates a heterogeneous transfer learning problem for document

Table 12: Breakdown of the 20 newsgroups dataset for transfer learning

Dataset	$\mathbf{D_s}$	$\mathbf{D_t}$
comp vs. sci	comp.graphics comp.os.ms.windows.misc sci.crypt sci.electronics	comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x sci.med sce.space
comp vs. talk	comp.graphics comp.sys.mac.hardware comp.windows.x talk.politics.mideast talk.religion.misc	comp.os.ms.windows.misc comp.sys.ibm.pc.hardware talk.politics.guns talk.politics.misc
rec vs. sci	rec.autos rec.sport.baseball sci.med sci.space	rec.motorcycles rec.sport.hockey sci.crypt sci.electronics
rec vs. talk	rec.autos rec.motorcycles talk.politics.guns talk.politics.misc	rec.sport.baseball rec.sport.hockey talk.politics.mideast talk.religion.misc
sci vs. talk	sci.electronics sci.med talk.politics.misc talk.religion.misc	sci.crypt sci.space talk.politics.guns talk.politics.mideast

classification. In addition to $P(X_s) \neq P(X_t)$ because the source and target data comes from different sub-domains, now $\chi_s \neq \chi_t$ because the source and target data come from different top-level domains. In this new problem we no longer know which words are the same in the different domains (i.e. "bit" may be the *ith* word in the source domain but we have no idea which index corresponds to "bit" in the target domain or even if the word "bit" is found in the target domain, let alone if it has the same semantic meaning in both domains. This also means that although technically $Y_s = Y_t$ because we use (0,1) for the class labels, semantically $Y_s \neq Y_t$ because the source task may be to classify documents as either belonging to recreation or science while the target task may be to

Table 13: Summary statistics of the newsgroups datasets

Id	# Features	# + Instances	# - Instances	# Meta-Features
$D_s(cs)$	9892	1958	1972	19784
$D_t(cs)$	9892	2923	1977	19784
$D_s(ct)$	10624	2914	1568	21248
$D_t(ct)$	10624	1967	1685	21248
$D_s(rs)$	14974	1984	1977	29948
$D_t(rs)$	14974	1993	1972	29948
$D_s(rt)$	15253	1984	1685	30506
$D_t(rt)$	15253	1993	1568	30506
$D_s(st)$	15327	1971	1403	30654
$D_t(st)$	15327	1978	1850	30654

classify documents as belonging either to talk or computers.

Domain Adaptation

For the Newsgroups dataset, we compare the ISFSR technique using 10% of the labeled data in D_t to perform the mapping against several baselines. *Self* uses a naïve Bayes classifier which has been trained and tested on the target dataset using 10-fold cross-validation. *None* uses a naïve Bayes classifier which has been trained on the source dataset and tested on the target dataset. The source and target feature spaces are adjusted to have the same number of features by adding zero-valued features as necessary. No attempt is made to adjust for the domain differences. *TCA* is a domain adaptation technique which projects both the source and the target domain onto a shared subspace of reduced dimensionality [73]. We compare our results against the unsupervised TCA using a linear kernel with 30 dimensions. We also compare against the semi-supervised TCA (SSTCA) using a linear kernel with 30 dimensions. Since the class distribution is balanced in these datasets we report only the accuracy scores. Figure 17 shows the results. Error bars are shown at the 95% confidence level.

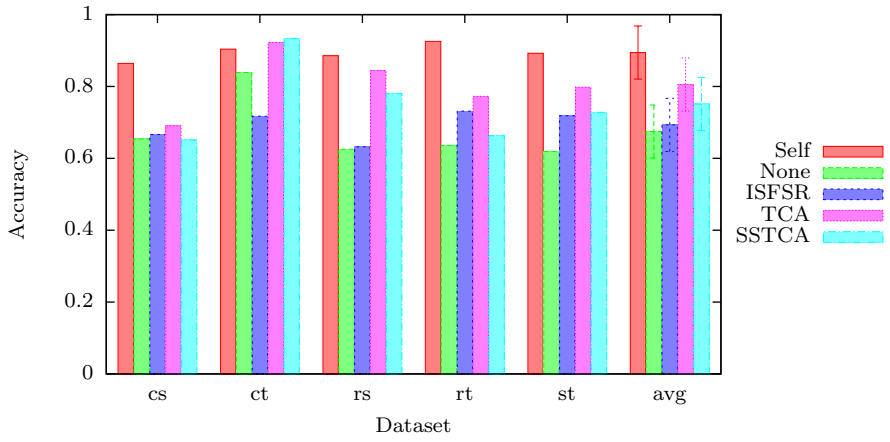


Figure 17: Newsgroups dataset results with $P(X_s) \neq P(X_t)$. ISFSR is not the best choice in this situation but is usually better than not performing any type of transfer. *Self* is the result when the source and target dataset are the same (and uses all the labeled target data).

From these results it is clear the ISFSR is not the best technique for transfer learning when $\chi_s = \chi_t$ and $P(X_s) \neq P(X_t)$. This is not surprising since ISFSR is designed mainly to handle different feature spaces. The performance results of ISFSR are low on the first three datasets (cs, ct, and rs), with ISFSR only slightly outperforming *None* on cs and rs, and actually performing worse than *None* on ct. *TCA* and *SSTCA* also struggles to improve performance on the cs dataset but do well on the ct and rs datasets. These results show the importance of further research into detecting and avoiding negative transfer. ISFSR performs much better on the last two datasets (rt and st), improving the classification accuracy by approximately 10% as compared to *None*. The accuracy of ISFSR is still lower than *TCA*, but the gap is much narrower. On two of the datasets (cs and rt) ISFSR even performs better than *SSTCA*. Note that ISFSR is a technique which has been designed to specifically handle the case when $\chi_s \neq \chi_t$, while *TCA* is designed to handle the case when $P(X_s) \neq P(X_t)$. In this experiment, $\chi_s = \chi_t$ but

$P(X_s) \neq P(X_t)$. When viewed in this light, the results of the two algorithms are not surprising. Of interest is that ISFSR is able to show some improvement in many cases even when $\chi_s = \chi_t$ and $P(X_s) \neq P(X_t)$.

The second interesting thing to note is that when ISFSR performs well, *TCA* tends to do worse (compare ct and rs to rt and st). One possible explanation for this might be that the differences between $P(X_s)$ and $P(X_t)$ are greater in rt and st. As the differences increase, the problem begins to more closely resemble the case when $\chi_s \neq \chi_t$. The negative correlation between ISFSR and *TCA* is not manifested on the results for the cs dataset. The reason this occurs is unclear but it may be related to the fact that the performance of *Self* is lowest for the cs dataset, possibly indicating that the dataset is harder to learn than the other datasets. Further research is needed to investigate these ideas.

Heterogeneous Feature-Spaces

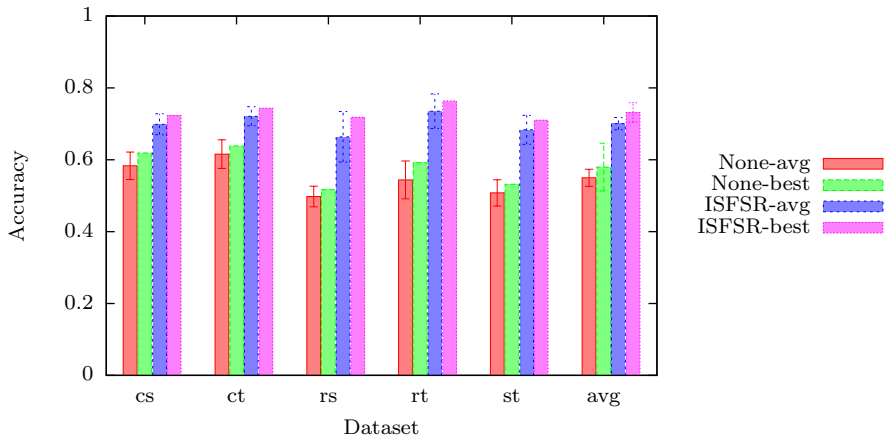


Figure 18: Newsgroups dataset results with heterogeneous transfer and no translation oracle. ISFSR clearly outperforms the baseline technique where features-spaces are not aligned.

The second experiment we conduct using the newsgroups dataset involves transferring knowledge between datasets where $\chi_s \neq \chi_t$. This is a significant step away from the previous experiment where, $\chi_s = \chi_t$ and $P(X_s) \neq P(X_t)$. It is also the first time this type of problem has been considered for document classification when no translation oracle is available. Since this is the first time such a problem has been tried, we cannot directly compare against any previous results. We report the results for each target dataset, averaged over all five source datasets, and the best results for each target dataset. In this experiment, the only baseline we have to compare against is the performance when no transfer is performed (*None*). This is similar to applying a random mapping between the feature spaces. The results are shown in Figure 18. Not surprisingly, the accuracy of None is close to random guessing, ranging from 50-60%. The exciting result is that the accuracy of ISFSR is much better, achieving as high as 73% accuracy when averaged over the source datasets and 76% accuracy for the best dataset. A two-tailed paired t-test gives a p-value of .00005 over all the datasets, and p-values between .01 and .002 for the individual datasets.

We emphasize that this transfer problem reflects differences along three of the four possible transfer variables. Specifically, $\chi_s \neq \chi_t$, $P(X_s) \neq P(X_t)$, and $f_t() \neq f_s()$. Additionally, although $Y_s = Y_t$, as we are using 0 and 1 for class labels, semantically the 0 and 1 represent different labels in the different datasets. We have successfully trained a classifier to recognize documents as belonging to the categories of "recreation" or "talk" and used the learned model to classify documents as belonging to either "computers" or "science".

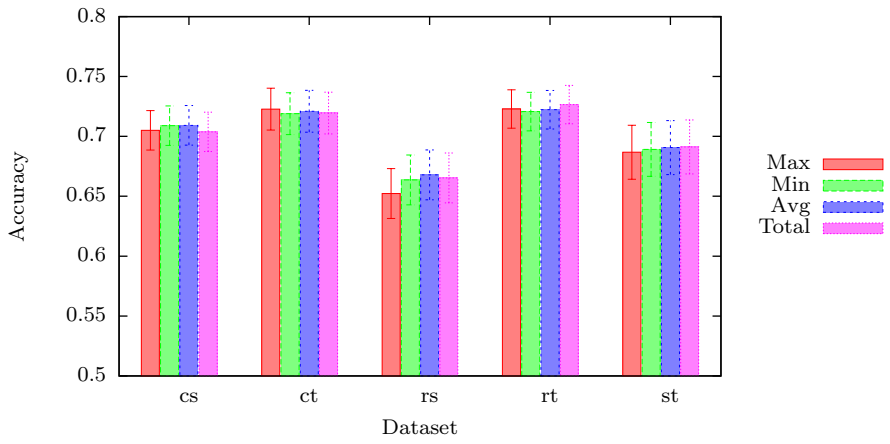


Figure 19: Newsgroups dataset results comparing different aggregation techniques. The process of aggregating target features which mapped to the same source feature has little effect on the overall performance of ISFSR.

Aggregation Results

The third experiment we conduct, using the newsgroups dataset, shows the effect of choosing a particular aggregation method for mapping multiple dimensions in the target domain to a single dimension in the source domain. Specifically, we compare ISFSR using the following aggregation techniques: Maximum, Minimum (greater than 0), Average and Total. The results are shown in Figure 19. Surprisingly, the aggregation method has little effect on the overall accuracy of the technique as applied to the newsgroups datasets. Running an ANOVA on the results yields a p-value of .95 indicating that the results are not statistically significant.

The reason the aggregation technique has little effect on the accuracy results is not clear. However, we can rule out the explanation that there just is not much aggregation to be done. A quick look at the generated mappings shows that hundreds of attributes in the target domain map to tens of attributes in the source domain and many more attributes in the source domain have two or more attributes mapping to them from the

target domain. Thus there is indeed a large amount of aggregation occurring. We can think of a few other possible explanations, the features being aggregating may be of little importance in defining class boundaries or the features being aggregated may have similar enough values that any of the aggregation techniques work equally well. We plan to investigate these ideas in future work.

Ensemble Learning

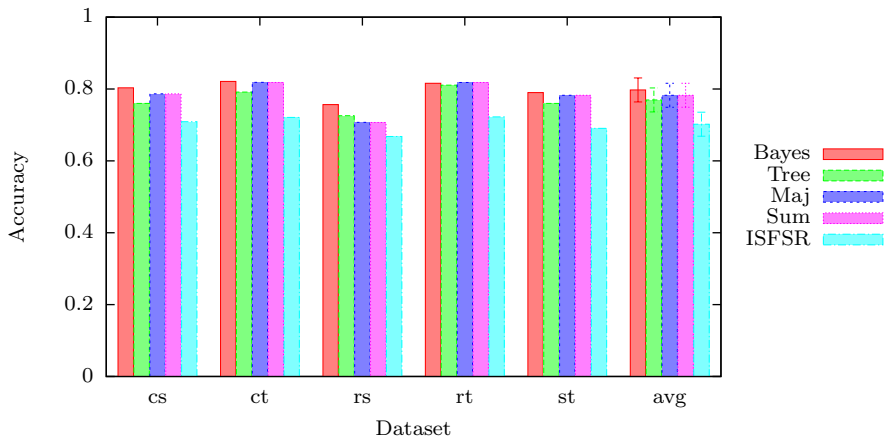


Figure 20: Newsgroups dataset results comparing ensemble techniques. No single ensemble technique is clearly better than any other ensemble technique. However, all of the ensemble techniques perform better than when only a singe source domain is employed.

We also evaluate the performance of the ELFSR techniques on the newsgroups datasets. For each newsgroup target dataset D_t we use all the other newsgroup datasets as source datasets. This gives us a total of nine source datasets for each target dataset. We consider both voting ensembles and stacking ensembles. The results are shown in Figure 20. *Bayes* is a stacking ensemble using Naive Bayes as the ensemble classifier, *Tree* is a stacking ensemble using a Decision Tree as the ensemble classifier, *Maj* is a majority voting ensemble, *Sum* is a sum of probabilities voting ensemble, and ISFSR is

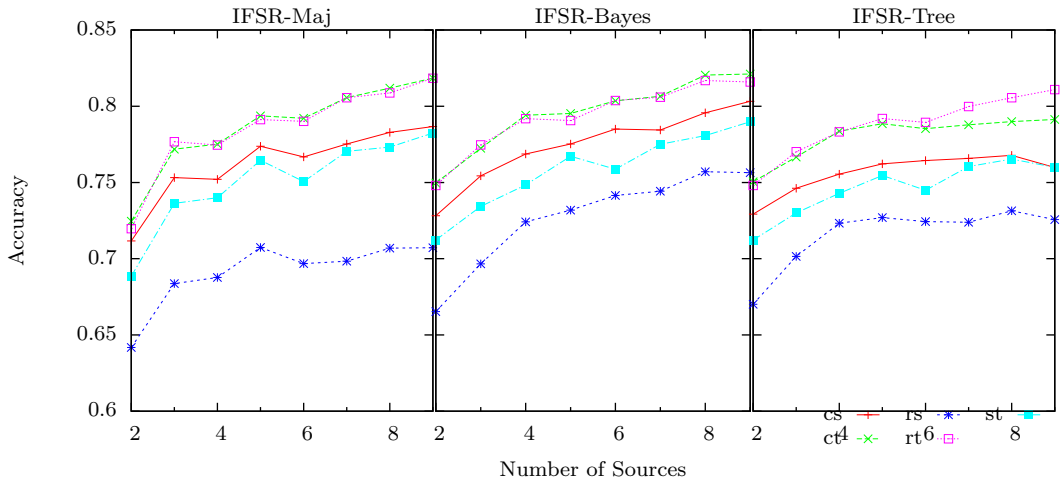


Figure 21: Learning curve for the ensemble classifiers. As more source classifiers are included the accuracy continues to improve.

the average result without using an ensemble learner. All of the ensemble techniques evaluated show better results than the basic ISFSR technique. Applying a one-way ANOVA to the results yields a p-value of .003 indicating that the difference in means are statistically significant. Unlike in the activity recognition domain, here we do not see as much difference between the ensemble techniques themselves as each technique performs similarly to the others. The Naive Bayes stacking ensemble has the highest accuracy scores but the other techniques are within a few percentage points. As can be seen by the confidence intervals, the means for each technique show significant overlap with each other except for the baseline technique.

In addition to comparing the performance of ISFSR and ELFSR against other techniques we also consider how the number of source datasets affects the performance achieved by the techniques.

We generate learning curves for the newsgroups dataset as shown in Figure 21. As the number of source classifiers increase, so does the overall accuracy. This performance increase occurs most rapidly with the inclusion of the first few classifiers and then slowly

tapers off as more source classifiers are added.

One interesting pattern that emerges in majority voting learning curve is the effect of odd and even number of source datasets. Each time the number of source datasets increases from odd to even there is essentially no improvement. However, each time the number of source datasets increases from even to odd there is a corresponding jump in the resulting accuracy. This makes sense intuitively because with an even number of sources, ties are broken arbitrarily (leading to an average accuracy of 50% for the tied cases). When a new classifier is added it acts as the tie-breaking vote. Since the accuracy of the classifier is greater than 50% we would expect the performance to increase, which it does.

We have looked at experimental results addressing Objectives 1.1 - 1.3. The FSR techniques are shown to perform well against the manual mapping technique. In many cases, they also perform better than a classifier trained and tested solely on the target dataset. Furthermore, FSR is shown to generalize to other domains including document classification.

5.2 Multi-view Experimental Results

The next set of experimental results focus on the new sensing platform problem and support Objectives 1.4 - 1.7. We evaluate our proposed multi-view approaches to learning from heterogeneous devices using two activity recognition datasets by varying amounts of labeled training data (graphs are on a log scale). Both datasets collect data from multiple, heterogeneous sensor classes and both include data collected from multiple participants. To avoid issues related to changing data distributions we evaluate the data

from each participant separately. In every experiment we report activity accuracy results using 10-fold cross validation and averaging over the entire collection of participants. We also report the unweighted average recall.

In the Opportunity dataset [94], 4 participants performed 5 rounds of scripted activities to check object placements, make/drink coffee, make a sandwich, and relax in a chair. They also performed 1 round of drill activities such as open/close the refrigerator, clean the table, and open/close a drawer. The twelve 3-axis wearable accelerometers represent one view and the seven wearable inertial measurement units represent the second view. We use the same features as Sagha et al. [97] which consist of the raw sensor values sampled every 500ms averaged over a 5-second window. This is the continuous feature representation described in Section 2.2.2. The dataset is labeled for locomotion activities: Stand, Walk, Sit and Lie.

In the CASAS PUCK dataset, 25 participants performed three trials of six activities in a smart space that is equipped with ambient motion and door sensors, object vibration sensors and two on-body 6-axis accelerometers [98]. The six activities to be detected are sweeping, medication, cooking, watering plants, hand washing, and washing countertops. The location of the motion sensors are shown in Figure 22 and 23. Object vibration sensors have been placed on objects including the broom, dustpan, duster, water pitcher, bowl of noodles, measuring cup, glass, fork, watering can, hand soap dispenser, dish soap dispenser, pill dispenser, medicine bottles and other items. The object vibration sensors prior to being attached to objects are shown in Figure 23. The on-body accelerometers are attached to the participants dominant arm and waist as shown in Figure 24.

In order to provide consistency we employ the same features as in the Opportunity dataset, namely the raw sensor values sampled every 500ms and averaged over a 5-second

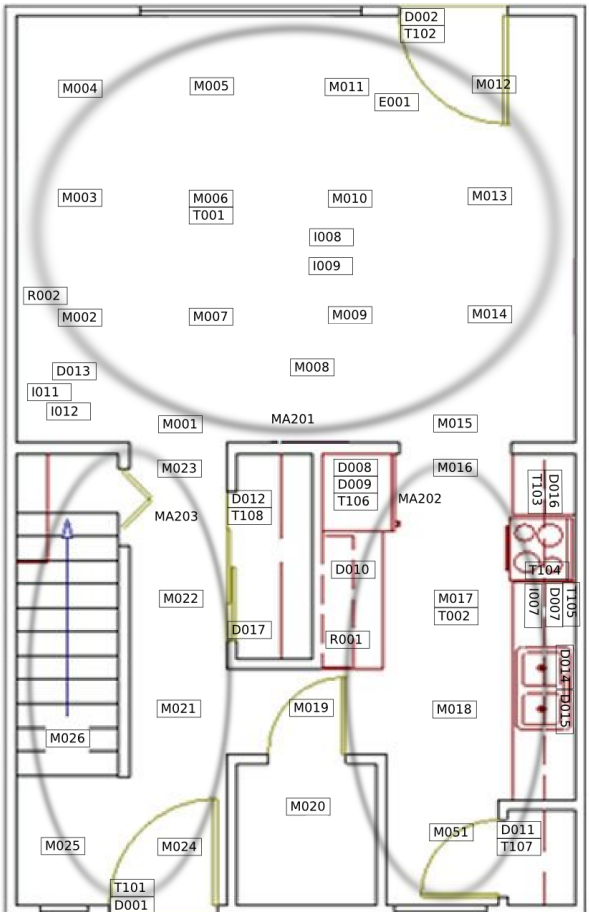
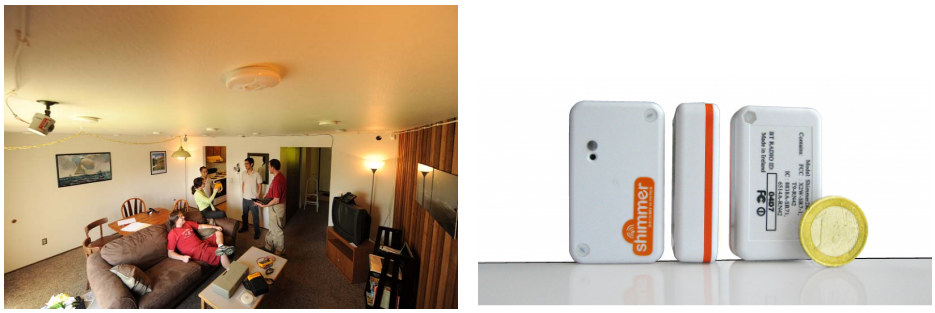


Figure 22: Ambient motion sensor placement. The small square 'MXXX' labels are narrow-view motion sensors. The larger circles represent the areas seen by the wide-view motion sensors.

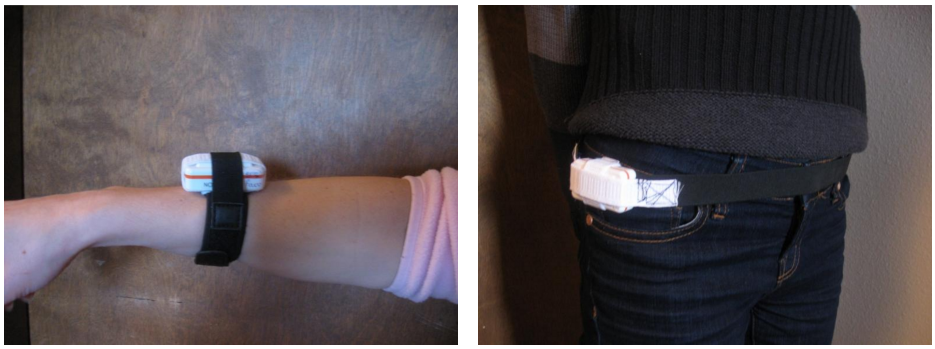
window. The labels are the six activities: sweeping, medication, cooking, watering plants, hand washing, and washing countertops.

All of the proposed algorithms are compatible with virtually any base classification technique. In these experiments a decision tree classifier is used as overall it performed the best for the two datasets. Other algorithms such as Logistic Regression, k Nearest Neighbors, and Support Vector Machines were tried. However, no single algorithm consistently outperformed the others and in some cases the increased run time made the



(a) Ambient motion sensors (b) Object vibration sensors

Figure 23: Sensors in the apartment



(a) Arm accelerometer (b) Waist accelerometer

Figure 24: Placement of on-body accelerometers

approach impractical for the evaluation.

5.2.1 Two Views

For these experiments, the ambient sensors represent the first view and the accelerometers represent the second view. The results focus on the accuracy and recall scores of the target view but we include the accuracy and recall scores of the source view in Appendix B. In the first experiment, we consider the accuracy of the classifier for the target view (On-body accelerometers view) as the amount of labeled data varies. This shows us how the multi-view learning algorithms perform when both the source and target views have

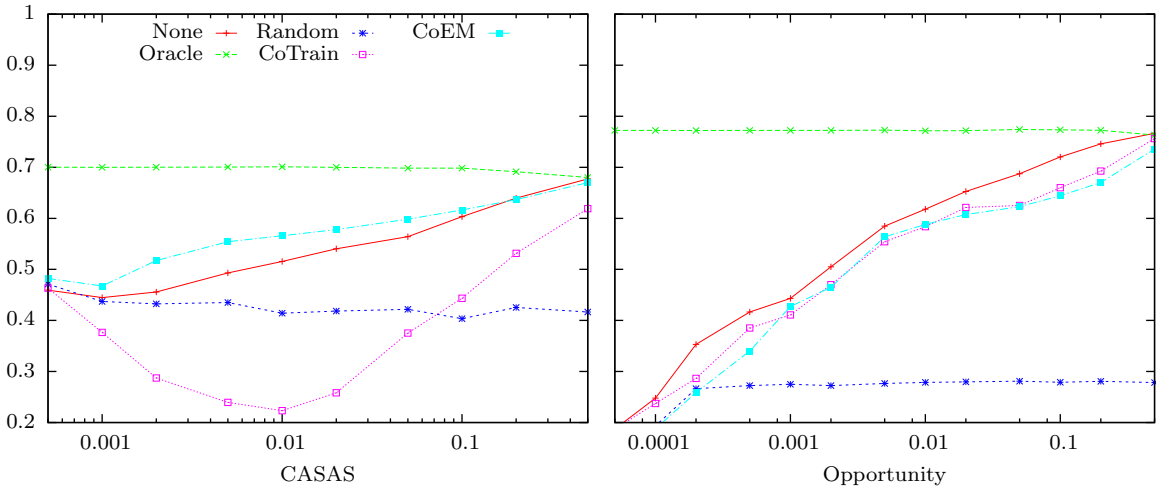


Figure 25: Two view informed MVTL classification accuracy vs. labeled data. Co-EM shows a small improvement over the baseline None on the CASAS dataset. Neither Co-Training nor Co-EM shows improvement over the baseline None on the Opportunity dataset.

a limited amount of labeled data. This supports Objective 1.4 In our personal activity recognition ecosystem example, this tests the scenario of bringing two different activity recognition systems online at the same time. By allowing the systems to collaborate we hope to improve the overall accuracy. We consider three different baselines against which we can compare our results. The first baseline, Oracle, trains a classifier on the unlabeled data using a perfect oracle to label the unlabeled data. The second baseline, None, trains a classifier using only the labeled data available in the target view. The third baseline, Random, randomly assigns a class label weighted by the observed class distribution. For the CASAS dataset, we choose $n = 10$ for the Co-Training algorithm and fix the number of iterations of Co-EM to 10. For the Opportunity dataset, we choose $n = 1000$ for the Co-Training algorithm and fix the number of iterations of Co-EM to 3. Other values of n were considered but the resulting accuracy showed little variation.

The results on the CASAS dataset are shown in Figures 25 and 26. A One-way

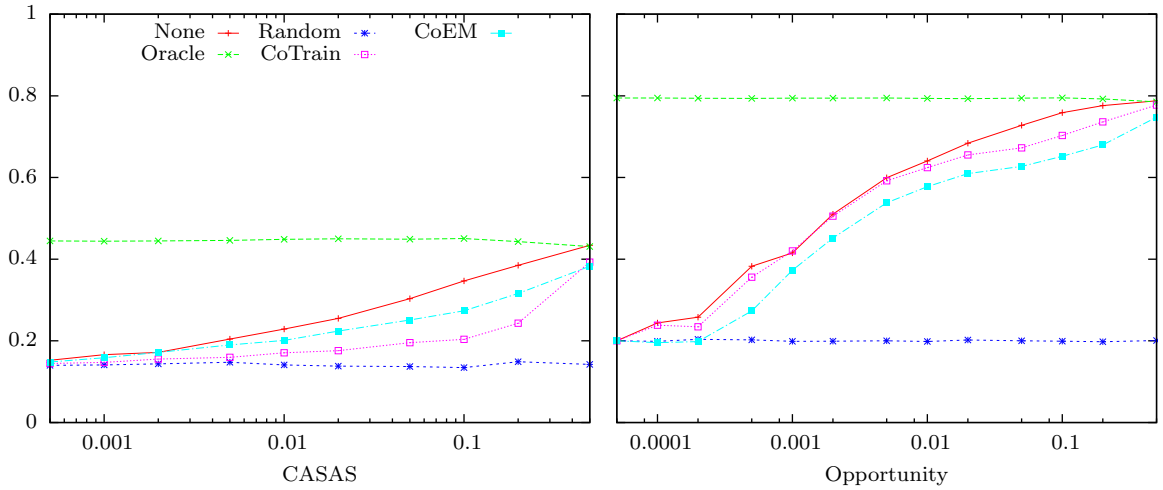


Figure 26: Two view informed MVTL average recall vs labeled data. Co-Training matches or is slightly below the None baseline on the Opportunity dataset. Neither technique outperforms None on the CASAS dataset.

ANOVA test indicates that the differences between mean accuracy values for the Co-EM, Co-Training, and None techniques are significant ($p < 0.05$) on all amounts of labeled training data except when the amount of labeled training data is 0.0005 of the dataset. The Co-EM technique consistently improves upon the baseline None with the margin of improvement decreasing as the amount of labeled data increases. This provides evidence that applying Co-EM can improve the accuracy of the system when bringing two new activity recognition systems online simultaneously. The Co-Training technique does not follow the expected upward learning curve trajectory. It also fails to improve upon the baseline techniques. One possible explanation for this is that the two views are not conditionally independent given the label, which causes the Co-Training algorithm to fail in some situations.

Figure 25 and Figure 26 also show the results for the Opportunity dataset. A One-way ANOVA test indicates that the differences between mean accuracy values for the Co-EM, Co-Training, and None techniques are significant ($p < 0.05$) on amounts of labeled

training data between 0.002 and 0.2 of the dataset. In this situation, the collaboration between new systems does not boost performance and in fact it has a negative effect. The CoEM algorithm fails to match the performance of the None baseline as does the Co-Training algorithm. We suspect this is due to the fact that the two views not only violate the conditional independence assumption but are likely to be highly correlated since both views use similarly-placed wearable sensors.

Next, we repeat the first experiment using the uninformed Multi-view techniques. The labeled data is only used to train the classifier in the source view. This tests the situation where one activity recognition system has been brought online and provided a limited amount of labeled training data. Shortly thereafter, a second activity recognition system is also brought online but no labeled training data is provided. For the Manifold Alignment algorithm we choose d to be the minimum number of dimensions in the source and target views, maximizing the information retained by the dimensionality reduction step. The results on the CASAS dataset are shown in Figure 27 and 28. A One-way ANOVA test indicates that the differences between the means of the three techniques are significant ($p < 0.05$) for all amounts of labeled training data. The Manifold Alignment technique has no clear trend with the accuracy remaining between 40%-50%. The Teacher-Learner technique, on the other hand, shows clear improvement as the amount of labeled data used by the teacher increases. The technique even approaches the accuracy achieved by the ideal Oracle technique.

Figure 27 and Figure 28 also show the results of the Uninformed Multi-view techniques on the Opportunity dataset. The results are similar to the results seen on the CASAS dataset and a One-way ANOVA test confirms the difference between techniques

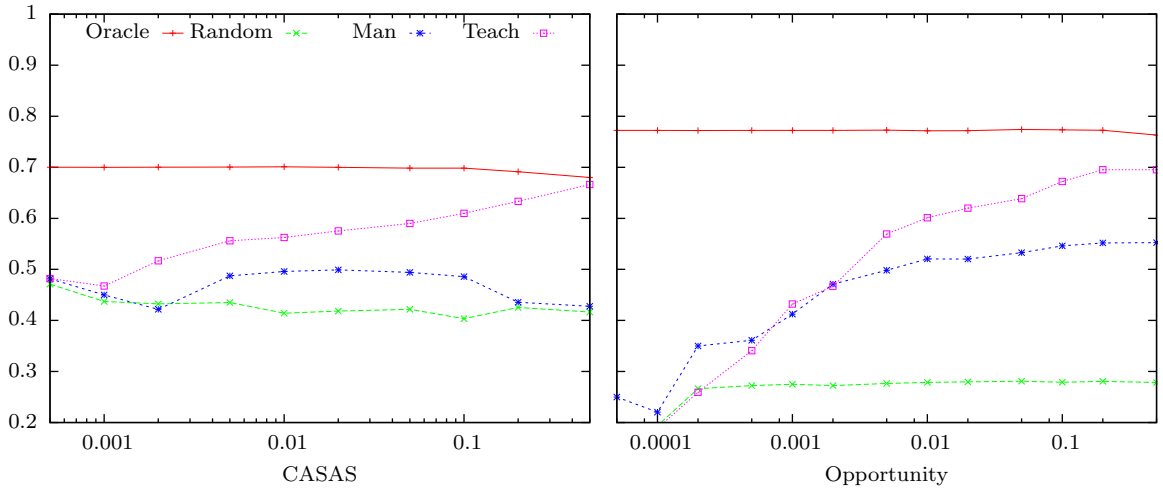


Figure 27: Two view uninformed MVTL classification accuracy vs. labeled data. The teacher-learner algorithm shows steady improvement as the amount of labeled data in the source view increases. It also outperforms Manifold Alignment.

are significant ($p < 0.05$). The Manifold Alignment technique does not perform particularly well although in this case it does show improvement as the amount of labeled data increases. The poor performance of the technique on both datasets is likely due to the invalid assumption that the data from both views can be projected onto a shared manifold in a lower-dimensional subspace. The Teacher-Learner technique again shows clear improvement as the amount of labeled data available to the teacher increases and approaches the ideal accuracy achieved by the Oracle technique.

In contrast to the previous experiments, we now want to look at the performance of these algorithms when a well-trained activity recognition system is in place and a second system is brought online with only a limited amount of available training data. This supports Objective 1.5. We assume that the source view has a significant amount of labeled training data (i.e. 50% of the dataset) and we vary the amount of labels in the target view. Figure 29 and Figure 30 show the results. In this case, we get a much different picture than in the previous experiments. Co-EM and Teacher-Learner, now

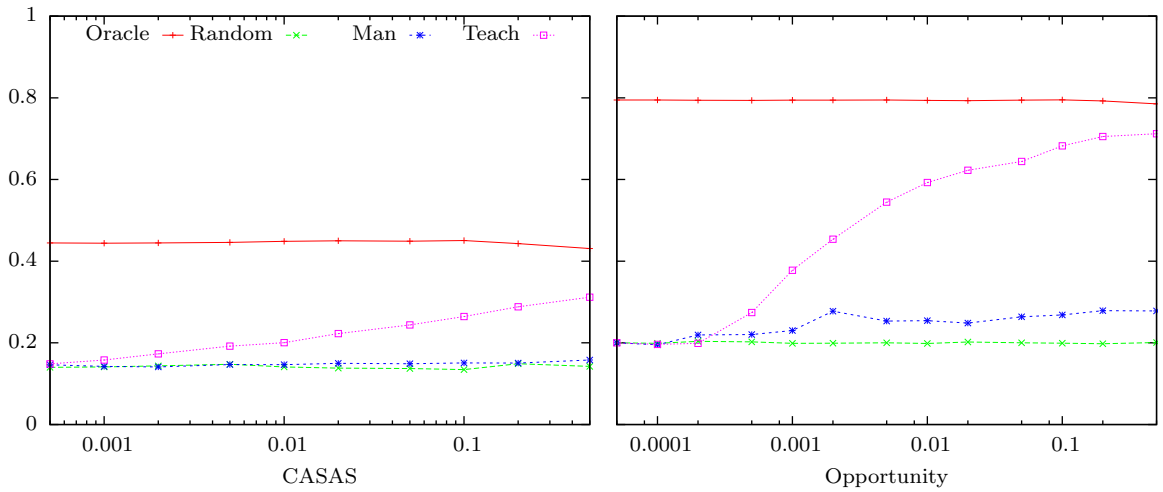


Figure 28: Two view uninformed MVTL average recall vs. labeled data. The teacher-learner algorithm shows steady improvement as the amount of labeled data in the source view increases. Teacher-learner also outperforms Manifold Alignment, which does little better than Random.

come much closer to matching the ideal Oracle technique. They both are significantly better ($p < 0.05$) than the None baseline technique for both the accuracy and the recall metrics. Additionally, Co-Training also performs significantly better ($p < 0.05$) than the baseline None technique as indicated by a paired student's t-test.

We also evaluate our collaboration method, Personalized ECOsystem (PECO), of combining the Teacher-Learner technique with an informed multi-view technique. To do this, we let the teacher bootstrap the initial labeled data for use by Co-Training or Co-EM algorithms. The teacher is initially trained on 50% of the data. The teacher then provides labels for a varying amount of data that the learner can use for training. In this situation, a well-trained activity recognition system is already in place and we now bring online a second activity recognitions system without providing any labeled training data. The PECO accuracy results are shown in Figure 29 and the recall results are

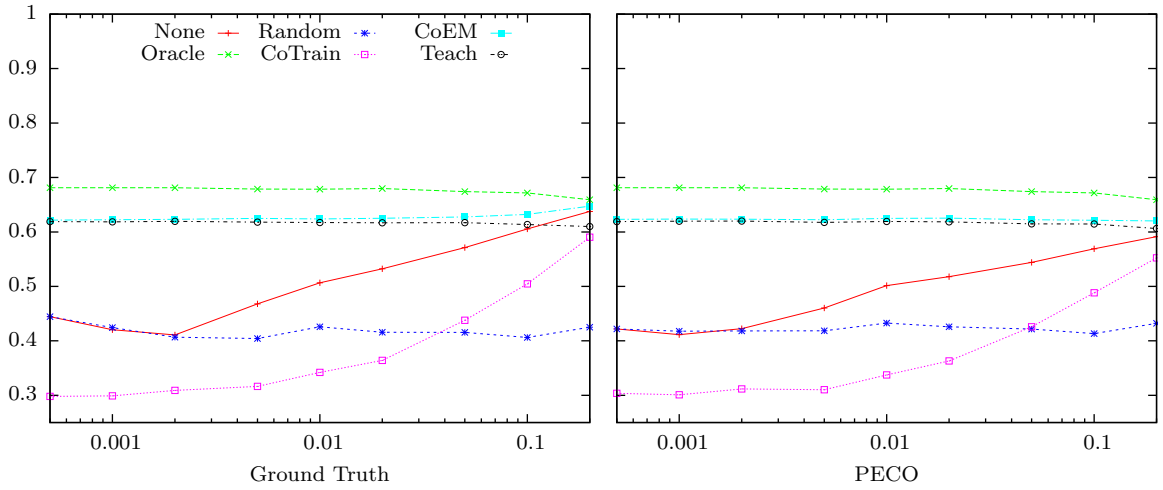


Figure 29: Two view well-trained/PECO accuracy results. CASAS accuracy with 50% labeled source data. Both the co-EM and the teacher-learner algorithms show substantial improvement over the None baseline using ground truth labels or the bootstrapped labels (PECO).

shown in Figure 30. Co-EM with bootstrapping performs better than using the Teacher-Learner technique alone, while Co-Training with bootstrapping does not improve upon the Teacher-Learner method. Interestingly, the results are almost identical to the results of the previous experiment. This indicates that our proposed label bootstrapping method yields performance that is on par with the performance when ground truth labels are available.

These results indicate that we can create a personalized activity-aware ecosystem capable of training and adapting to new sensor classes without requiring human intervention. A single trained activity recognition system is sufficient to train subsequent activity recognition systems without any additional labeled data. This is valuable for applications in which activity recognition needs to smoothly transition between data sources such as environment sensors, wearable or phone sensors, video data, or objects sensors, without the need for expert guidance and without the requirement that labeled

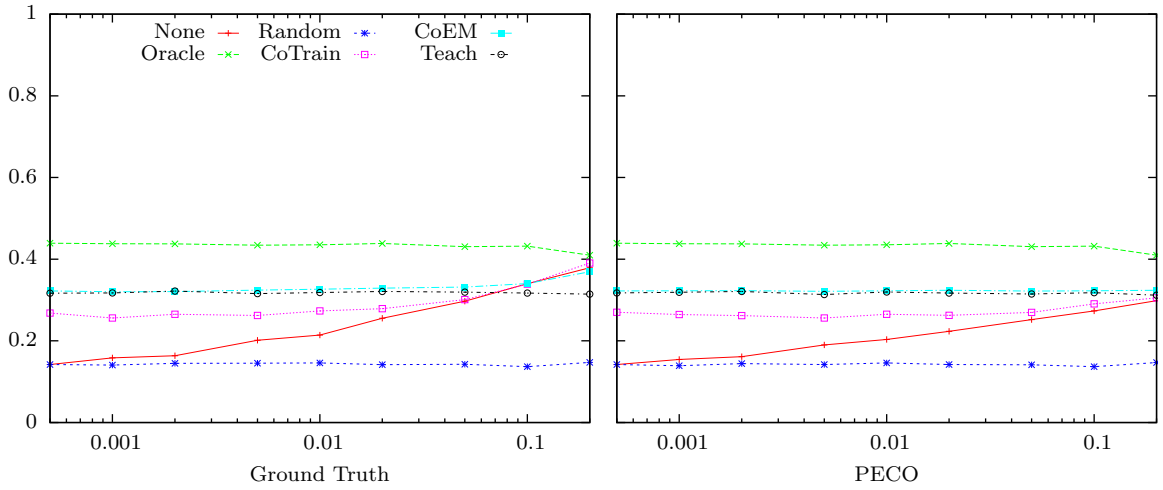


Figure 30: Two view well-trained/PECO recall results. CASAS recall with 50% labeled source data. Both the co-EM and the teacher-learner algorithms show substantial improvement over the None baseline using ground truth labels or the bootstrapped labels (PECO).

data be provided for the new view.

5.2.2 Three Views

Having shown the effectiveness of MVTL when two views (or sensing platforms) are present, we now turn our attention to the effect of adding an additional view into the mix. For these experiments we only consider the CASAS dataset since the Opportunity dataset did not have additional views of sufficient quality. We use the ambient motion sensors as one view, the object sensors as a second view and the on-body accelerometers as the third view. We still use a decision tree classifier and the same features as before. The accuracy and recall scores for each view have been empirically measured using all of the labeled data and applying 10-fold cross-validation. The results are shown in Table 14. Each view differs in its ability to correctly classify the activities being performed.

Table 14: Accuracy and recall scores for each view with all of the labeled data

ID	View	Accuracy	Recall
1	Object Shake Sensors	0.9095	0.8362
2	Ambient Motion Sensors	0.7851	0.6408
3	On-body Accelerometers	0.7250	0.5239

The individual performance of the views affects the overall performance of the multi-view learning algorithms but we cannot explore all possible combinations of views and orderings in this chapter. Instead, we focus on particular examples and discuss how the choices we make affect the results of the algorithms. Each experiment will look at the accuracy and recall score of view 2, the Ambient Motion Sensor view. By focusing on the results of a single view we can make comparisons across different experiments. The results for the source views are included in Appendix B.

In the first experiment, we only use two views. This provides additional support for Objectives 1.4 and 1.5 but will primarily be used for future comparisons when a third view is also considered (Objective 1.6). We report the accuracy and recall of the ambient motion sensor view (View 2) under three scenarios which we will refer to as Equal, Trained and PECO.

The Equal scenario uses an equal amount of labeled training data in both views which we vary from 0.05% to 50% of the total training data. The Trained scenario also uses 0.05% to 50% of the total training data as labeled data, but it also uses an additional 40% of the training data in the object vibration sensor view (View 1) as labeled data. The PECO scenario uses 40% of the training data in the object sensor view as labeled data. Labels for 0.05% to 50% of the total training data for the other view are then bootstrapped using a classifier trained on the object sensor view.

The Equal scenario corresponds to a situation where multiple untrained activity

recognition systems are being brought online simultaneously with a limited amount of labeled training data. The Trained scenario corresponds to a situation in which an existing trained activity recognition systems is already in place and now an additional activity recognition system is being brought online with a limited amount of labeled training data. The PECO scenario corresponds to a situation in which an existing trained activity recognition systems is already in place and now an additional activity recognition system is being brought online without any labeled training data. All of the training labels for the new activity recognition systems are provided by the existing activity recognition system. This uses our PECO algorithm. In the several of the figures we denote the well-trained views with an * and we denote the bootstrapped views with a +.

Note that these experimental results are different from the previous experiments. In the previous experiments the ambient motion sensors view functioned as the well-trained source (teacher) view, while in these experiments it will function as the untrained target (learner) view.

We use the same three baselines as before against which we can compare our results. In the PECO scenario, None uses the bootstrapped class labels. We keep $n = 10$ for the Co-Training algorithm and fix the number of iterations of Co-EM to 10.

The accuracy results are shown in Figure 31 and the average recall results are shown in Figure 32. The results for all three scenarios are similar to the results we saw in the previous section. When both views have equal amounts of labeled training data, the Co-EM and Teacher-Learner algorithms are similar to the baseline comparison of None. Both the accuracy and the unweighted average recall score is slightly higher than the None baseline for most levels of labeled training data. Co-Training, on the other

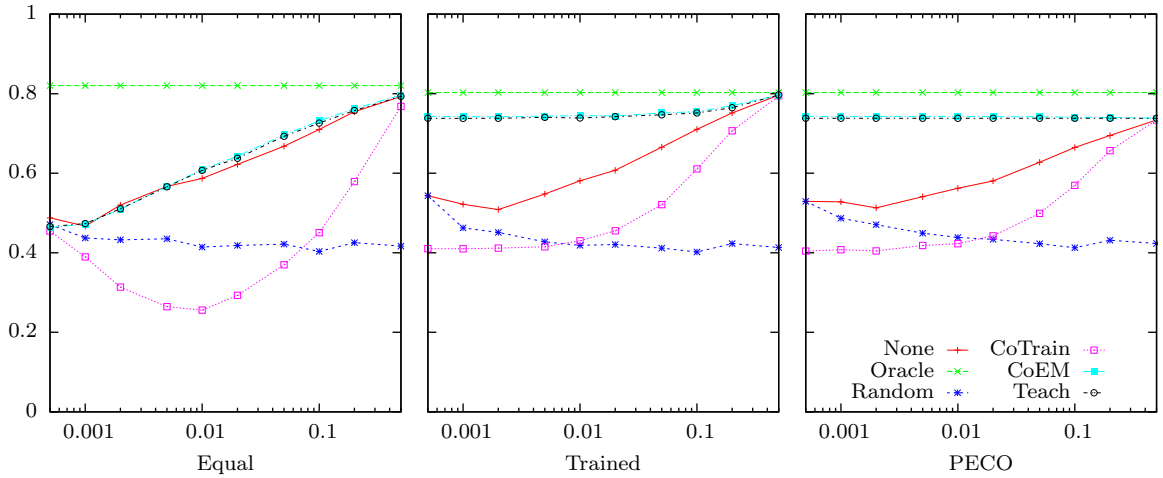


Figure 31: Classification accuracy vs. labeled data using View 1 and View 2. In the Equal scenario both Co-EM and the teacher-learner algorithm show a slight improvement over the None baseline. In the Trained and PECO scenarios Co-EM and teacher-learner both show substantial improvement over the None baseline.

hand, has a much lower accuracy compared to the None baseline while the recall scores are similar to the other techniques. As the amount of labeled data increases all three algorithms come closer to matching the performance of the ideal Oracle results. These results show limited promise for using multi-view learning when both views have similar amounts of labeled training data.

When the object vibration sensor view (View 1) has been trained using an additional 40% of the training data as labeled data, the Co-EM and Teacher-Learner algorithms both outperform the None baseline in terms of accuracy and recall scores. The Co-Training algorithm also outperforms the None baseline in terms of recall scores but does not perform as well as the None baseline in terms of accuracy scores. This indicates that the Co-Training algorithm is able to correctly identify larger proportions of each class than the None technique but does not have as high of an accuracy score due to the imbalanced nature of the classes.

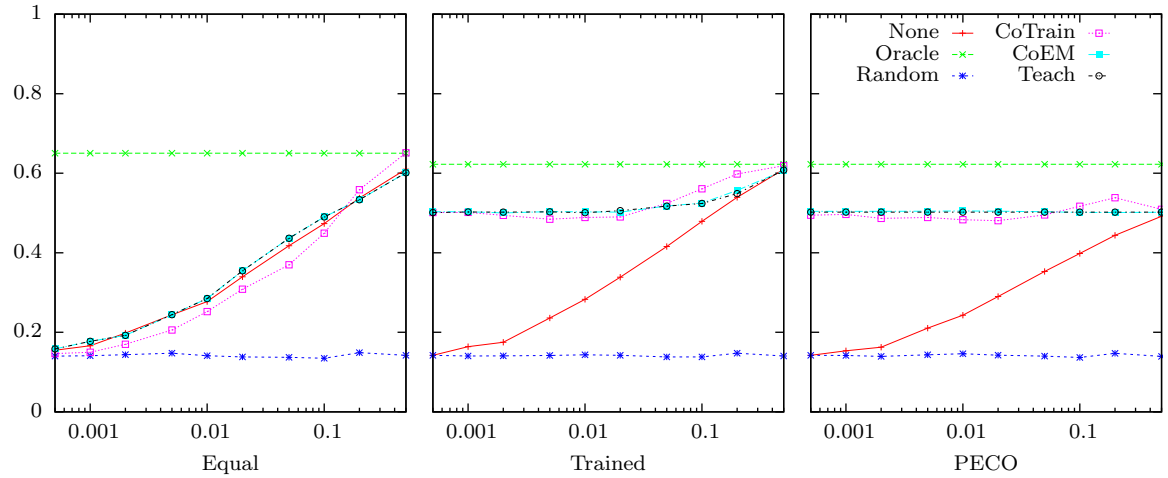


Figure 32: Average recall vs labeled data using View 1 and View 2. In the Equal scenario both Co-EM and the teacher-learner algorithm show a slight improvement over the None baseline. In the Trained and PECO scenarios all three techniques show substantial improvement over the None baseline.

When labels have been bootstrapped using a classifier trained on the object vibration sensor view (View 1) with 40% of the training data, the results are very similar to the previous results using ground truth labels. Only as the amount of bootstrapped data reaches 20% or more of the data do the differences even become noticeable. This is promising for our PECO algorithm because it indicates that bootstrapping a small number of labeled data points is as good as using the ground truth labels.

We now repeat the previous experiments but use the on-body accelerometer view (View 3) in place of the object vibration sensor view (View 1). In both of the previous experiments, the teacher view had a higher potential accuracy than the learner view but in this case the learner view has a higher potential accuracy than the teacher view. Figure 33 shows the accuracy results and Figure 34 shows the recall results.

Now, when both views have equal amounts of labeled training data, all three techniques (Co-Training, Co-EM and Teacher-Learner) perform worse than the baseline

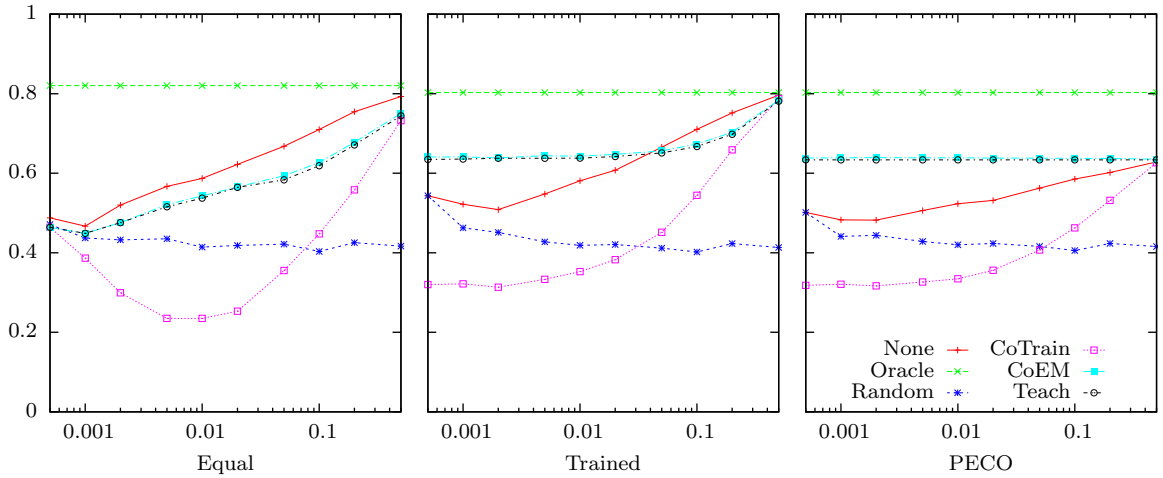


Figure 33: Classification accuracy vs. labeled data using View 3 and View 2. In the Equal scenario no technique improves upon the None baseline. In the Trained and PECO scenarios Co-EM and teacher-learner both show improvement over the None baseline until about 5% of the data is labeled.

None technique. When the on-body accelerometer view (View 3) is trained with an 40% of the training data, the Co-EM and Teacher-Learner technique both out perform the None baseline until about 2% of the training data is labeled. The Co-Training technique exhibits similar performance with the unweighted average recall metric but the accuracy metric is again well below the None baseline. Finally, using bootstrapped labels appears to be an effective strategy and for small amount of labeled data is nearly as good as using the ground truth labels.

From the previous experiments it is obvious that the baseline accuracy of the views affects the performance of the Co-Training, Co-EM and Teacher-Learner algorithm. Figure 35 and Figure 36 demonstrate this explicitly by comparing the average deference in accuracy and recall scores, respectively, between the previous two experiments for each technique.

We now want to look at the affect on the performance of these algorithms as an

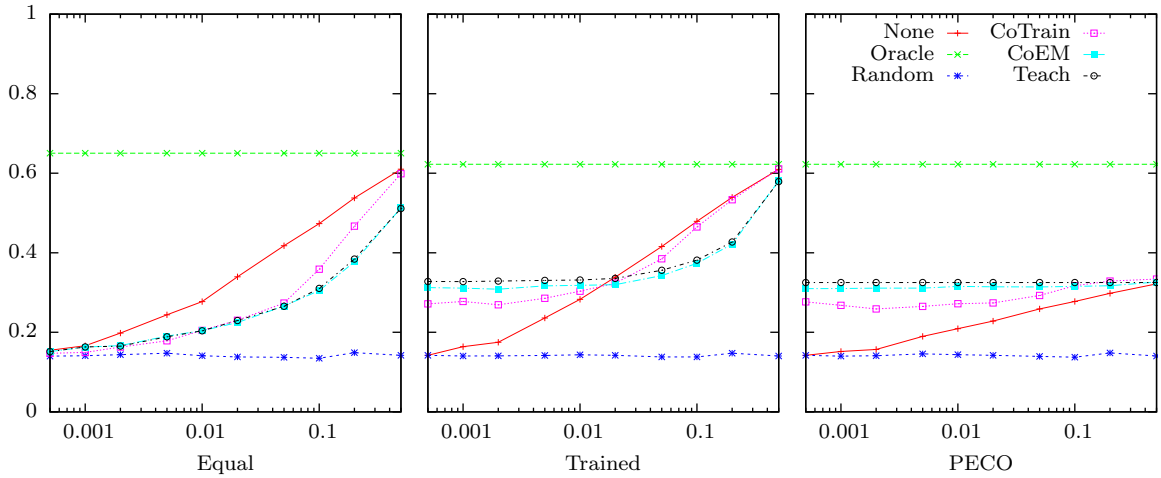


Figure 34: Average recall vs labeled data using View 3 and View 2. In the Equal scenario no technique improves upon the None baseline. In the Trained and PECO scenarios Co-EM and teacher-learner both show improvement over the None baseline until about 2% of the data is labeled.

additional view is introduced. Introducing a third view has both potential benefits as well as drawbacks. For example, a third view may help increase the accuracy by introducing additional diversity. On the other hand, introducing a third view may also be detrimental if the diversity leads to less accurate classification. One can imagine an effect similar to the "telephone" game played by children in which a message is slowly changed and degraded each time it is passed to the next person. To explore these effects we consider the following scenarios in relation to our original scenario using View 1 and View 2. A * represents views which have been trained with the additional 40% of labeled data. A + represents views whose labels have been bootstrapped. (x,y) represents the ensemble of view x and view y.

1. Equal-1,2,3. This is the same as the Equal scenario but View 3 is also included.

2. Trained-1*,2,3. This is the same as the Trained scenario but View 3 is also included.

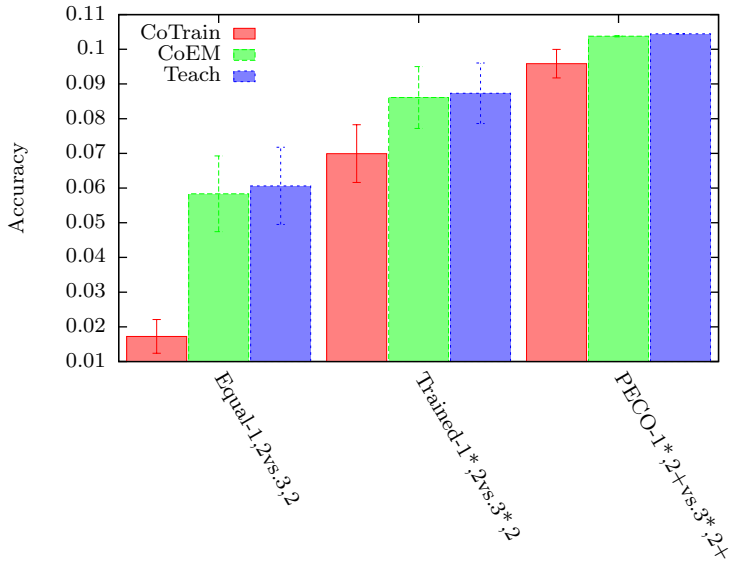


Figure 35: Effects on accuracy of changing the teacher. When a more accurate teacher (i.e. View 1) is used, the accuracy on view 2 increases.

3. PECO-1*,2+,3+. This is the same as the PECO scenario but View 3 is also included.

4. Trained-1*,2,3*. This is the same as Trained but View 3 is also included and uses the additional 40% of labeled data. Only the object vibration sensor view (View 1) acts as the teacher.

5. PECO-1*,2+,3*. This is the same as PECO but View 3 is also included and uses the additional 40% of labeled data. Only the object vibration sensor view (View 1) bootstraps the labels for View 2.

6. Trained-(1*,3*),2. This is the same as Trained but View 1 and View 3 are combined into an ensemble classifier. Both View 1 and View 3 use the additional 40% of labeled data.

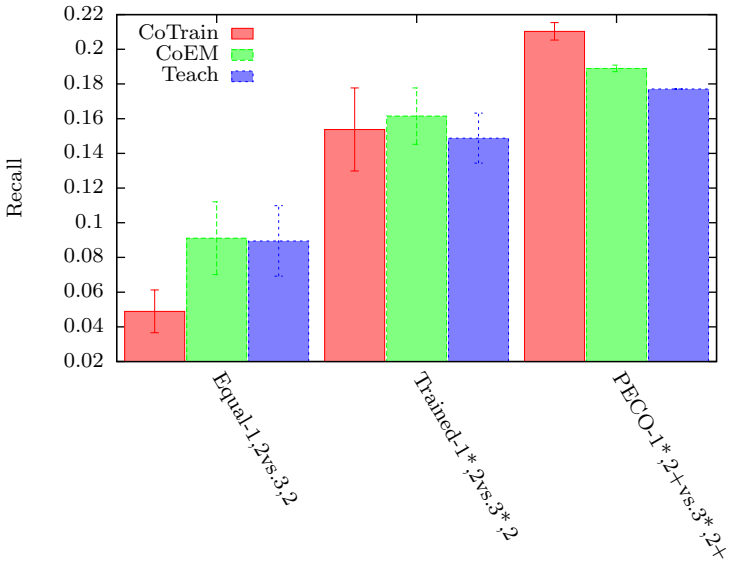


Figure 36: Effects on recall of changing the teacher. When a more accurate teacher (i.e. View 1) is used, the recall on view 2 increases.

7. PECO-(1*,3*),2+. This is the same as PECO but but View 1 and View 3 are combined into an ensemble classifier. Both View 1 and View 3 use the additional 40% of labeled data and the ensemble is used to bootstrap labels for View 2.

By comparing the performance results of each scenario against the performance when only two views are used we can better see the effect that introducing the third view has. This supports Objective 1.6. Figure 37 shows the difference between the accuracy of the original scenario using only View 1 and View 2 and the accuracy of the new scenarios using all three views for each algorithm averaged over the different amounts of labeled data available for training. Positive values indicate that the accuracy of the new scenario is higher than the accuracy of the original scenario. Similarly, Figure 38 shows the difference between the unweighted average recall of the original scenario using only View 1 and View 2 and the unweighted average recall of the new scenarios using

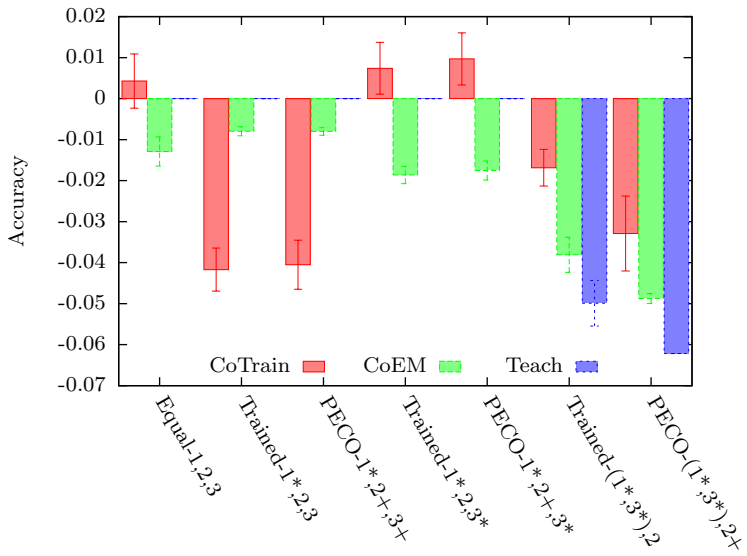


Figure 37: Effects on accuracy of adding an additional view (view 3). Overall, view 3 is less accurate so this has the effect of lowering the accuracy on view 2.

all three views for each algorithm. The Teacher-Learner algorithm is unaffected by the introduction of a third view except in the scenarios in which the third view is combined with the first view using an ensemble method. The Co-Training and Co-EM algorithms, on the other hand, both have lower accuracy and recall scores than the original scenario. The exceptions are the accuracy scores for a few of the Co-Training algorithms but even these show an improvement of less than 1%. The Co-Training algorithm is most affected by the introduction of a third untrained view, while the Co-EM and Teacher-Learner algorithms are most affected by the introduction of a third well-trained view used in an ensemble.

We repeat the same experiments but with the role of the first and third views reversed. Figure 37 shows the difference between the accuracy of the original scenario using only View 3 and View 2 and the accuracy of the new scenarios using all three

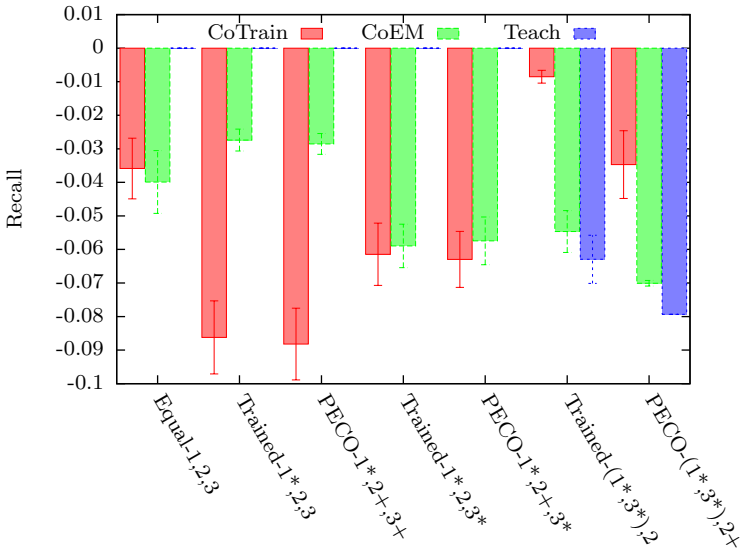


Figure 38: Effects on recall of adding an additional view (view 3). Overall, view 3 is less accurate so this has the effect of lowering the recall on view 2.

views for each algorithm averaged over the different amounts of labeled data available for training. Similarly, Figure 38 shows the difference between the unweighted average recall of the original scenario using only View 3 and View 2 and the unweighted average recall of the new scenarios using all three views for each algorithm. In contrast to the previous experiment, this experiment shows the introduction of a third view as having a beneficial affect on the accuracy and recall scores. As before, the Teacher-Learner algorithm is unaffected by the introduction of a third view except in the scenarios in which the third view is combined with the first view using an ensemble method. The Co-Training and Co-EM algorithms both show considerable improvement when the third view is well trained. When the third view is not previously trained, it has little effect on the resulting accuracy and recall scores.

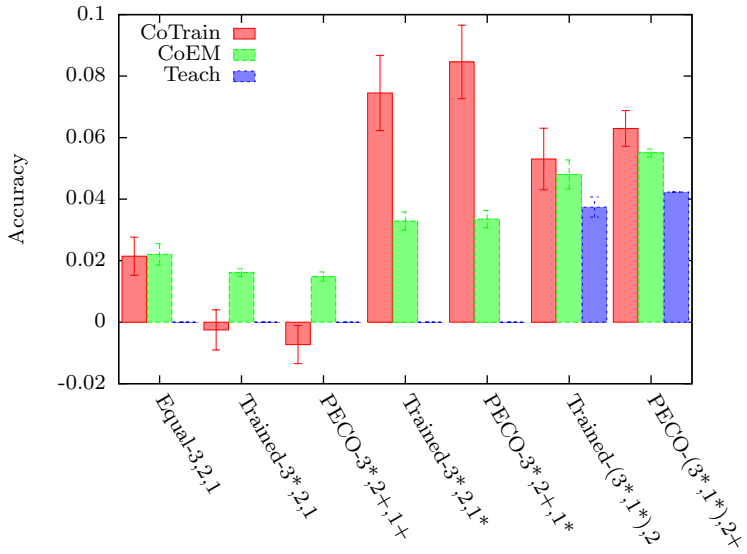


Figure 39: Effects on accuracy of adding an additional view (view 1). Overall, view 1 is more accurate so this has the effect of increasing the accuracy on view 2.

When considered as a whole these results highlight some important differences between the proposed algorithms. Co-Training treats all views equally. This makes the algorithm more stable when the order of the views is changed but also limits the accuracy of the approach by failing to give preference to views which are more accurate. Co-Training is also able to make good use of having multiple well-trained views. The Teacher-Learner algorithm is highly dependent on the selection of a good teacher and does not make any use of having additional well-trained views unless those views are used in an ensemble. The Co-EM algorithm is somewhere in between. The order of the views still affects the accuracy of the algorithm (think "telephone" effect) but not as strongly as the Teacher-Learner algorithm is affected. Ordering views by their achievable accuracy scores seems to yield the best results.

From these results we can draw a few general guidelines. First, when considering

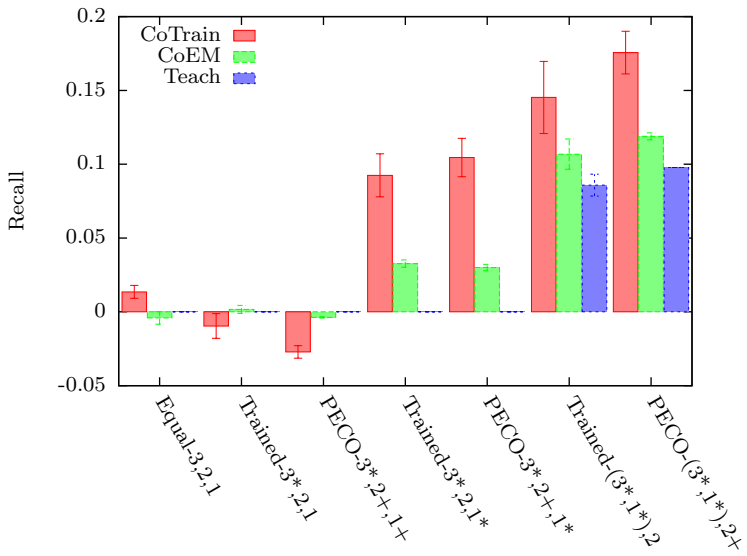


Figure 40: Effects on recall of adding an additional view (view 1). Overall, view 1 is more accurate so this has the effect of increasing the recall on view 2.

which system or systems to use as the teacher, if the accuracy of the systems are known, the most accurate systems should play the role of teacher. However, suppose these collaborative learning techniques have been applied across several generations of smart devices. At this point we may no longer know the exact accuracy of any given system. If the accuracy of the systems are unknown, combining the systems using an ensemble method may help mitigate the risk of selecting an inaccurate system to play the role of teacher but may also result in lower accuracy scores than might otherwise be achieved. Second, bootstrapping a small number of labels appears to be as effective as using ground truth labels. A single trained activity recognition system is sufficient to train subsequent activity recognition systems without any additional labeled data. This is valuable for applications in which activity recognition needs to smoothly transition between data sources such as environment sensors, wearable or phone sensors, video data, or objects

sensors, without the need for expert guidance and without the requirement that labeled data be provided for the new view. However, as the amount of bootstrapped data increases the effectiveness of the technique decreases.

5.2.3 Accuracy Bounds

Finally, we consider the achieved accuracy of the learner in relation to the expected, upper and lower bounds in support of Objective 1.7. For the expected bounds, we consider the $z = 1/(k-1)$ bound proposed in Equation 4.10, the z-priors bound in Equation 4.11, the conditional probability bound in Equation 4.12, the average of the upper and lower bound and the underestimated expected bound of $p * q$. We evaluate these bounds using the ten-fold cross-validation technique described earlier. The values used for computing the bounds, such as the teacher accuracy and the level of agreement between the teacher and the learner, are taken from the observed performance on the validation set.

Figure 41 shows the results for each teacher-learner view combination. The upper and lower bounds do in fact bound the accuracy from above and below. They are not particularly tight bounds, deviating by as much as 20% from the observed accuracy in these experiments. The loose bounds are a result of not knowing the true values for q_1 and q_2 in Equation 4.3 but instead knowing only the value of q.

The simplest estimation $p * q$ of the expected accuracy is also the least accurate. Including the $(1-p)(1-q)z$ term improves the estimate of the accuracy. The conditional expected bounds provides a closer estimate to the observed accuracy but still consistently underestimates the actual accuracy of the learner. Taking the average of the upper and

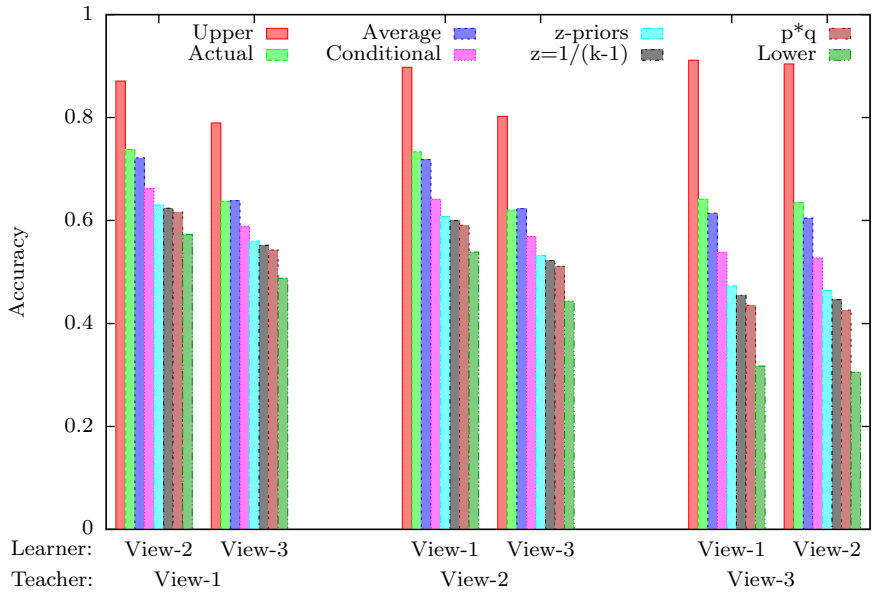


Figure 41: Accuracy bounds for the learner. The upper and lower bounds do indeed bound the actual accuracy. The Conditional estimate, z-priors estimate, $z = 1/(k-1)$ estimate and $p * q$ estimate all consistently underestimate the actual accuracy. The Average of the upper and lower bounds provides the best estimate of the actual accuracy.

lower bound provides the most accurate estimate of the actual learner accuracy.

5.3 Conclusions

In this chapter, we have presented empirical results highlighting the effectiveness of the proposed algorithms and validating each of the proposed objectives. We have shown the effectiveness of FSR techniques in solving the new environment problem (Objective 1.1) with average performance improvements of 4% to 6% over the generalized model. The average performance of FSR techniques does not match the performance of a model trained for the specific environment but the best individual target-source mappings have an average performance improvement of about 1%. The FSR techniques are also shown to be effective at utilizing multiple source domains through ensemble learning (Objective

1.2. When a stacked ensemble is used the performance improvement over the generalized model is between 15% and 20% and the improvement over the specialized model is 5% to 15%. Finally we have shown that FSR techniques are applicable to other domains by evaluating the performance on a document classification problem (Objective 1.3).

The MVTL techniques have been shown to be effective in solving the new sensing platform problem. When two sensing platforms are introduced simultaneously into the new environment and there is no other pre-existing sensing platform in the environment (Objective 1.4), only Co-EM is able to outperform training each model separately. When a pre-existing sensing platform is already trained in the environment and a new sensing platform is introduced (Objective 1.5), the MVTL techniques improve upon training the new model separately by as much as 20%. When considering the effect of integrating three or more different sensing platforms (Objective 1.6), we see that the PECO-E algorithms tends to be the most stable across different situations while the PECO and Co-EM have the potential to yield the best performance but all also highly dependent upon the accuracy of the sensing platforms. Finally, we have evaluated the derived accuracy bounds (Objective 1.7) and found that the upper and lower bounds do indeed bound the observed accuracy of the learner. Further, the average of the upper and lower bound is found to be a good approximation to the observed accuracy, usually approximating the accuracy within a few percentage points.

We can draw some general guidelines outlining when each technique is likely to be most effective. First, determine if you face a new environment problem (i.e. you have two or more dataset with different feature-spaces) or a new sensing platform problem (i.e. you have to or more dataset that share instances). For the new environment problem, if you have multiple source datasets then applying the ELFSR techniques will likely yield

the best performance results. If only a single source dataset is available, then GAFSR or GrFSR are likely to be the most effective but, they also are more computationally expensive. ISFSR may give slightly lower accuracies but will also find a mapping much faster. Finally, if you do not have any labeled data in the target dataset then a manual mapping is your best bet. Currently, USFSR is not up to the task of producing a good mapping automatically. For the new sensing platform problem we can encounter several different scenarios. In the traditional multi-view scenario, where each sensing platform has similar amounts of labeled data, co-EM is the most effective, but in many cases just training each view separately may yield better results. In a well-trained scenario, where there is at least one view which is already trained, both Co-EM and Teacher-Learner approaches work well. If no labeled data is available for the new sensing platform using PECO to bootstrap labels is almost as effective as if the ground truth labels were available. Finally, when selecting which views to include in the multi-view learning, if the accuracy of the systems are known, the most accurate systems should play the role of teacher. If the accuracy of the systems are unknown, combining the systems using an ensemble method as in PECO-E may help mitigate the risk of selecting an inaccurate system to play the role of teacher but may also result in lower accuracies scores than might otherwise be achieved.

Chapter 6

Conclusions

Activity recognition has many promising applications such as health-care and energy efficiency. One challenge that currently hinders the large-scale adoption of activity recognition systems is the need to gather labeled data on which the activity recognition system can be trained. Every time a new environment or a new sensing platform is encountered, new training data must be gathered. We have developed several heterogeneous transfer learning algorithms which solve the new environment problem by mapping the feature-space of the new environment to previously encountered feature-spaces. The techniques are able to outperform a manual mapping based upon sensor locations and in some cases outperforms a classifier which has been trained solely in the new environment. Using multiple source classifiers in a stacking ensemble is shown to produce even better classification accuracies.

We have also developed several heterogeneous transfer learning algorithms based on multi-view learning to solve the new sensing platform problem. In particular, we have shown that the PECO algorithm is able to bring new activity recognition systems online without the use of any manually labeled training data for the new recognition system. The accuracy and recall of the new system is nearly equivalent to that of a system which is provided ground truth labels in place of the bootstrapped labels. Furthermore, the accuracy and recall of the PECO trained system show significant improvement over a system which is trained only on the limited amount of ground truth labels.

In addition to the empirical results presented in Chapter 5, we also developed theoretical bounds on the accuracy of a learner trained under the teacher-learner model. We have shown how the teacher-learner model is a variation of multi-view learning and hence has the same PAC guarantees as other multi-view learning algorithms. Using the accuracy of the teacher and the level of agreement between the teacher and the learner we estimate the expected accuracy of the learner and we also develop upper and lower bounds on the accuracy of the learner.

In summary, the main contributions of this work include:

- A new class of heterogeneous transfer learning algorithms, FSR. The key feature of FSR is that, unlike most other heterogeneous transfer learning algorithms, FSR maps the target feature-space onto the source feature-space. This allows for more efficient evaluation of the empirical results produced by the mapping which can then be used to search for good mappings. The approach also facilitates the use of ensemble learners when combining multiple source domains.

- Novel heterogeneous transfer learning algorithms (GAFSR, GrFSR and SFSR) which do not rely on feature-feature, feature-instance, or instance-instance co-occurrence data. These algorithms can be used to solve the new environment problem and outperform the Manual mapping technique by about 5%. GAFSR and GrFSR have polynomial time complexities (cubed in the number of features and linear in the number of instances). SFSR also has a polynomial time complexity (squared in the number of features and linear in the number of instances).

- The ELFSR algorithms which use ensemble learning in conjunction with FSR to combine multiple source datasets. These techniques outperform the combined

Manual mapping technique by 20%.

- Framing the new sensing platform problem as a multi-view learning problem.
- A new class of uninformed multi-view transfer learning algorithms, PECO. The key feature of PECO is to use a trained activity recognition system as a teacher to bootstrap labels for a new activity recognition system. After the initial labels are bootstrapped, any other informed MVTL technique (such as co-Training or co-EM) can be applied to further improve the classification algorithm. These algorithms can be used to solve the new sensing platform problem. When compared to training a system without this data it results in performance improvements of 20%
- Novel algorithms for multi-view transfer learning using an ensemble to combine multiple views. These algorithms can be used to solve the new sensing platform problem and provide greater stability in the face of unknown initial accuracies.
- The novel application of multi-view learning to transfer knowledge between sensor modalities
- Positioning of the teacher-learner framework within transfer learning literature.
- PAC-style bounds to the teacher-learner framework.
- Formulas for the expected accuracy of the learner as well as upper and lower bounds on the accuracy. These can provide a good estimate of the accuracy of the learner when no labeled training data is available for evaluation.

Heterogeneous transfer learning is a hard problem in general. While successful, the techniques we presented here also have some important limitations which could be

addressed in future work. For example, the FSR techniques assume that although the features are semantically different they are syntactically similar. In other words, the features have similar values or ranges of values even if those values mean different things in the different domains. One possible technique which could be used to overcome this limitation is adding a normalization step so that all of the features have similar value ranges. The currently proposed uninformed FSR technique does not yet produce results comparable to the other techniques or to the baseline techniques. New meta-features could be explored to improve the accuracy of USFSR. Additionally, uninformed variants of GAFSR and GrFSR could be explored by using a fitness function which does not rely on labeled target data. One possible candidate function could be based upon preserving the distance between points in the original space and points in the mapped space.

The MVTL techniques rely on the assumption that two or more activity recognition systems can observe the same events and can communicate with one another. Our current work has focused on multiple sensor streams which are aligned using a common timestamp. However, in the future we plan to implement MVTL techniques which run in real-time allowing multiple sensing platforms to share label information through our middleware architecture. Another limitation of MVTL techniques is the need for a shared label space. Applying some of the transfer learning algorithms for handling different label spaces could be used to potentially overcome this challenge.

As the number of devices with sensing and computing capabilities increase, a personalized activity recognition ecosystem becomes possible. By developing techniques which solve the new environment problem and the new sensing platform problem, this work contributes to the goal of making such personalized ecosystems possible.

Bibliography

[1] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *Proceedings of the International Conference on Data Engineering*, pages 3–14, 1995.

[2] Hande Alemdar and Cem Ersoy. Wireless sensor networks for healthcare: A survey. *Computer Networks*, 54(15):2688 – 2710, 2010.

[3] A. Arnold, R. Nallapati, and W.W. Cohen. A comparative study of methods for transductive transfer learning. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, pages 77 –82, oct. 2007.

[4] Akin Avci, Stephan Bosch, Mihai Marin-Perianu, Raluca Marin-Perianu, and Paul Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *Architecture of Computing Systems (ARCS), 2010 23rd International Conference on*, pages 1 –10, Feb. 2010.

[5] S.M. Barnett and S.J. Ceci. When and where do we apply what we learn?: A taxonomy for far transfer. *Psychological bulletin*, 128(4):612–637, 2002.

[6] U. Blanke and B. Schiele. Remember and transfer what you have learned-recognizing composite activities based on activity spotting. In *Wearable Computers (ISWC), 2010 International Symposium on*, pages 1–8. IEEE, 2010.

[7] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistic*, 2007.

[8] John Blitzer, Ryan T. McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, 2006.

[9] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[10] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[11] J.P. Byrnes. *Cognitive development and learning in instructional contexts*. Allyn and Bacon, Boston, 1996.

[12] Alberto Calatroni, Daniel Roggen, and Gerhard Tröster. Automatic transfer of activity recognition capabilities between body-worn motion sensors: Training newcomers to recognize locomotion. In *Eighth International Conference on Networked Sensing Systems (INSS'11)*, Penghu, Taiwan, June 2011.

[13] Liangliang Cao, Zicheng Liu, and T.S. Huang. Cross-dataset action detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1998 –2005, June 2010.

[14] Marie Chan, Daniel Estve, Christophe Escriba, and Eric Campo. A review of smart homespresent state and future challenges. *Computer Methods and Programs in Biomedicine*, 91(1):55 – 81, 2008.

[15] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, May 2011.

[16] R. Chattopadhyay, N.C. Krishnan, and S. Panchanathan. Topology preserving domain adaptation for addressing subject based variability in semg signal. In *2011 AAAI Spring Symposium Series*, 2011.

[17] Liming Chen, J. Hoey, C.D. Nugent, D.J. Cook, and Zhiwen Yu. Sensor-based activity recognition. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(6):790–808, 2012.

[18] Jian Cheng and Kongqiao Wang. Active learning for image retrieval with co-svm. *Pattern Recognition*, 40(1):330–334, 2007.

[19] H.L. Chieu, W.S. Lee, and L.P. Kaelbling. Activity recognition from physiological data using conditional random fields. Technical report, Singapore-MIT Alliance (SMA), 2006.

[20] D. Cook. Learning setting-generalized activity models for smart spaces. *Intelligent Systems, IEEE*, PP(99):1, 2010.

[21] Diane J. Cook, Kyle D. Feuz, and Narayanan C. Krishnan. Transfer learning for activity recognition. *Knowledge and Information Systems*, 36:537–556, 2012.

[22] Diane J Cook, Narayanan C Krishnan, and Parisa Rashidi. Activity discovery and activity recognition: A new partnership. *Cybernetics, IEEE Transactions on*, 43(3):820–828, 2013.

[23] DJ Cook, M Schmitter-Edgecombe, et al. Assessing the quality of activities in a smart environment. *Methods of information in medicine*, 48(5):480, 2009.

[24] Wenyuan Dai, Yuqiang Chen, Gui-Rong Xue, Qiang Yang, and Yong Yu. Translated learning: Transfer learning across different feature spaces. In *Advances in Neural Information Processing Systems*, pages 353–360, 2008.

[25] Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 210–219. ACM, 2007.

[26] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 193–200, New York, NY, USA, 2007. ACM.

[27] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Self-taught clustering. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 200–207, New York, NY, USA, 2008. ACM.

[28] Hal Daumé, Abhishek Kumar, and Avishek Saha. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems*, pages 478–486, 2010.

[29] Hal Daumé, III and Daniel Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intillegence Research*, 26(1):101–126, May 2006.

[30] Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting Association of Computing Linguistics*, pages 256–263, 2007.

[31] Jesse Davis and Pedro Domingos. Deep transfer via second-order markov logic. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 217–224, New York, NY, USA, 2009. ACM.

[32] Thomas G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15. Springer-Verlag, 2000.

[33] Lixin Duan, Dong Xu, I.W. Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1959 –1966, June 2010.

[34] Charles Elkan. The foundations of cost-sensitive learning. In *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 2*, IJCAI'01, pages 973–978, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[35] Ali Farhadi and Mostafa Tabrizi. Learning to recognize activities from the wrong view point. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision ECCV 2008*, volume 5302 of *Lecture Notes in Computer Science*, pages 154–166. Springer Berlin / Heidelberg, 2008.

[36] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.

[37] Dehong Gao, Furu Wei, Wenjie Li, Xiaohua Liu, and Ming Zhou. Cotraining based bilingual sentiment lexicon learning. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[38] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[39] Tao Gu, Shaxun Chen, Xianping Tao, and Jian Lu. An unsupervised approach to activity recognition and segmentation based on object-use fingerprints. *Data and Knowledge Engineering*, 69(6):533–544, June 2010.

[40] Hirotaka Hachiya, Masashi Sugiyama, and Naonori Ueda. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, 80(0):93 – 101, 2012. Special Issue on Machine Learning for Signal Processing 2010.

[41] K.Z. Haigh and H. Yanco. Automation as caregiver: A survey of issues and technologies. In *AAAI-02 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care*, pages 39–53, 2002.

[42] L.K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

[43] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[44] D.H. Hu and Q. Yang. Transfer learning for activity recognition via sensor mapping. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[45] D.H. Hu, V.W. Zheng, and Q. Yang. Cross-domain activity recognition via transfer learning. *Pervasive and Mobile Computing*, 7(3):344 – 358, 2010.

[46] Sham M Kakade and Dean P Foster. Multi-view regression via canonical correlation analysis. In *Learning Theory*, pages 82–96. Springer, 2007.

[47] Eunju Kim, Sumi Helal, and D. Cook. Human activity recognition and pattern discovery. *Pervasive Computing, IEEE*, 9(1):48–53, 2010.

[48] Zsolt Kira. Inter-robot transfer learning for perceptual classification. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 13–20, 2010.

[49] Svetlana Kiritchenko and Stan Matwin. Email classification with co-training. In *Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research*, pages 301–312. IBM Corp., 2011.

[50] N.C. Krishnan. *A Computational Framework for Wearable Accelerometer-Based*. PhD thesis, Arizona State University, 2010.

[51] N.C. Krishnan, P. Lade, and S. Panchanathan. Activity gesture spotting using a threshold model based on adaptive boosting. In *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pages 155 –160, July 2010.

[52] N.C. Krishnan and S. Panchanathan. Analysis of low resolution accelerometer data for continuous human activity recognition. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 3337 –3340, April 2008.

[53] M. Kurz, G. Hölzl, A. Ferscha, A. Calatroni, D. Roggen, and G. Tröster. Real-time transfer and evaluation of activity recognition capabilities in an opportunistic system. In *ADAPTIVE 2011, The Third International Conference on Adaptive and Self-Adaptive Systems and Applications*, pages 73–78, 2011.

[54] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. In *Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data*, pages 10–18, 2010.

[55] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119. ACM, 2001.

[56] Antony Lam, Amit Roy-Chowdhury, and Christian Shelton. Interactive event search through transfer learning. In Ron Kimmel, Reinhard Klette, and Akihiro Sugimoto, editors, *Computer Vision - ACCV 2010*, volume 6494 of *Lecture Notes in Computer Science*, pages 157–170. Springer Berlin / Heidelberg, 2011.

[57] K. Lang et al. News weeder: Learning to filter netnews. In *12th International Conference of Machine Learning*, pages 331–339, 1995.

[58] Jonathan Lester, Tanzeem Choudhury, Nicky Kern, Gaetano Borriello, and Blake Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *Proceedings of the 19th international joint conference on Artificial intelligence*, pages 766–772, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[59] Lin Liao, Dieter Fox, and Henry Kautz. Location-based activity recognition using relational markov networks. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 773–778, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[60] Jingen Liu, M. Shah, B. Kuipers, and S. Savarese. Cross-view action recognition via view knowledge transfer. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3209 –3216, june 2011.

[61] Shenghua Liu, Wenjun Zhu, Ning Xu, Fangtao Li, Xue-qi Cheng, Yue Liu, and Yuanzhuo Wang. Co-training and visualizing sentiment evolvement for tweet events. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 105–106. International World Wide Web Conferences Steering Committee, 2013.

[62] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Munguia Tapia, and Stephen Intille. A long-term evaluation of sensing modalities for activity recognition. In *Proceedings of the 9th international conference on Ubiquitous computing*, pages 483–500, Berlin, Heidelberg, 2007. Springer-Verlag.

[63] U. Maurer, A. Smailagic, D.P. Siewiorek, and M. Deisher. Activity recognition

and monitoring using multiple sensors on different body positions. In *International Workshop on Wearable and Implantable Body Sensor Networks*, April 2006.

[64] L. Mihalkova, T. Huynh, and R.J. Mooney. Mapping and revising markov logic networks for transfer learning. In *Proceedings of the national conference on artificial intelligence*, volume 22, page 608. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[65] L. Mihalkova and R.J. Mooney. Transfer learning by mapping with minimal target data. In *Proceedings of the AAAI-08 Workshop on Transfer Learning for Complex Tasks*, July 2008.

[66] Lilyana Mihalkova and Raymond J. Mooney. Transfer learning from minimal target data by mapping across relational domains. In *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, pages 1163–1168, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[67] Melanie Mitchell. *An Introduction to Genetic Algorithms (Complex Adaptive Systems)*. A Bradford Book, third printing edition, February 1998.

[68] Thomas M. Mitchell. *Machine Learning*, chapter Bayesian Learning, pages 154 – 200. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[69] Fabian Nater, Tatiana Tommasi, Helmut Grabner, Luc van Gool, and Barbara Caputo. Transferring activities: Updating human behavior analysis *(both first authors contributed equally)*. In *ICCV WS on Visual Surveillance*, 2011.

[70] Kamal Nigam and Rayid Ghani. Analyzing the effectiveness and applicability of

co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 86–93. ACM, 2000.

[71] Paulito Palmes, Hung Keng Pung, Tao Gu, Wenwei Xue, and Shaxun Chen. Object relevance weight pattern mining for activity recognition and segmentation. *Pervasive and Mobile Computing*, 6(1):43–57, Feb. 2010.

[72] J.J. Pan, Q. Yang, H. Chang, and D.Y. Yeung. A manifold regularization approach to calibration reduction for sensor-network based tracking. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 988, 2006.

[73] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[74] Sinno Jialin Pan, James T Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *AAAI*, volume 8, pages 677–682, 2008.

[75] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web*, pages 751–760, 2010.

[76] S.J. Pan, J.T. Kwok, Q. Yang, and J.J. Pan. Adaptive localization in a dynamic wifi environment through multi-view learning. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 1108, 2007.

[77] S.J. Pan, D. Shen, Q. Yang, and J.T. Kwok. Transferring localization models across

space. In *Proceedings of the 23rd national conference on Artificial intelligence*, volume 3, pages 1383–1388, 2008.

[78] S.J. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[79] S.J. Pan, V.W. Zheng, Q. Yang, and D.H. Hu. Transfer learning for wifi-based indoor localization. In *Association for the Advancement of Artificial Intelligence (AAAI) Workshop*, page 6, 2008.

[80] Weike Pan, Erheng Zhong, and Qiang Yang. Transfer learning for text mining. In *Mining Text Data*, pages 223–257. Springer, 2012.

[81] Matthai Philipose, Kenneth P, Mike Perkowitz, Donald J. Patterson, Dieter Fox, Henry Kautz, and Dirk Hhnel. Inferring activities from interactions with objects. *IEEE Pervasive Computing*, 3:50–57, 2004.

[82] William Phillips and Ellen Riloff. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 125–132. Association for Computational Linguistics, 2002.

[83] David Pierce and Claire Cardie. Limitations of co-training for natural language learning from large datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 1–9, 2001.

[84] Ronald Poppe. A survey on vision-based human action recognition. *Image Vision Comput.*, 28(6):976–990, June 2010.

[85] Peter Prettenhofer and Benno Stein. Cross-lingual adaptation using structural correspondence learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(1):13, 2011.

[86] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, March 1986.

[87] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 759–766, New York, NY, USA, 2007. ACM.

[88] P. Rashidi and D.J. Cook. Transferring learned activities in smart environments. In *5th International Conference on Intelligent Environments*, volume 2, pages 185–192, 2009.

[89] P. Rashidi and D.J. Cook. Activity recognition based on home to home transfer learning. In *AAAI Workshop on Plan, Activity, and Intent Recognition*, 2010.

[90] P. Rashidi and D.J. Cook. Multi home transfer learning for resident activity discovery and recognition. In *KDD Knowledge Discovery from Sensor Data*, pages 56–63, 2010.

[91] P. Rashidi and D.J. Cook. Activity knowledge transfer in smart environments. *Pervasive and Mobile Computing*, 7(3):331–343, 2011.

[92] P. Rashidi, D.J. Cook, L.B. Holder, and M. Schmitter-Edgecombe. Discovering activities to recognize and track in a smart environment. *IEEE Transactions on Knowledge and Data Engineering*, 23(4):527 –539, April 2011.

[93] Yuanfang Ren, Yan Wu, and Yanbin Ge. A co-training algorithm for {EEG} classification with biomimetic pattern recognition and sparse representation. *Neurocomputing*, (0):–, 2013.

[94] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Forster, Gerhard Troster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, et al. Collecting complex activity datasets in highly rich networked sensor environments. In *Networked Sensing Systems (INSS), 2010 Seventh International Conference on*, pages 233–240. IEEE, 2010.

[95] Daniel Roggen, Kilian Frster, Alberto Calatroni, and Gerhard Trster. The adarc pattern analysis architecture for adaptive human activity recognition systems. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–18, 2010. 10.1007/s12652-011-0064-0.

[96] Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (3rd ed.)*. Pearson Education, 2010.

[97] H. Sagha, S.T. Digumarti, J. del R Millan, R. Chavarriaga, A. Calatroni, D. Roggen, and G. Troster. Benchmarking classification techniques using the opportunity human activity dataset. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 36–40, Oct 2011.

[98] Yasamin Sahaf. *Comparing Sensor Modalities for Activity Recognition*. PhD thesis, Washington State University, 2011.

[99] Dairazalia Sánchez, Monica Tentori, and Jesús Favela. Activity recognition for the smart hospital. *Intelligent Systems, IEEE*, 23(2):50–57, 2008.

[100] Shiliang Sun. A survey of multi-view machine learning. *Neural Computing and Applications*, 23(7-8):2031–2038, 2013.

[101] M.E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, 10:1633–1685, 2009.

[102] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[103] E. Thorndike and R.S. Woodworth. The influence of improvement in one mental function upon the efficiency of other functions.(i). *Psychological review*, 8(3):247–261, 1901.

[104] S. Thrun. *Explanation-based neural network learning: A lifelong learning approach*. Kluwer Academic Publishers, 1996.

[105] S. Thrun and L. Pratt. *Learning to learn*. Kluwer Academic Publishers, 1998.

[106] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[107] T. van Kasteren, G. Englebienne, and B. Kröse. Recognizing activities in multiple contexts using transfer learning. In *AAAI AI in Eldercare Symposium*, 2008.

[108] T. van Kasteren, G. Englebienne, and B. Kröse. An activity monitoring system for elderly care using generative and discriminative models. *Personal and Ubiquitous Computing*, 14(6):489–498, Sept. 2010.

[109] T. van Kasteren, G. Englebienne, and B. Kröse. Transferring knowledge of activity recognition across sensor networks. In Patrik Floren, Antonio Krger, and Mirjana Spasojevic, editors, *Pervasive Computing*, volume 6030 of *Lecture Notes in Computer Science*, pages 283–300. Springer Berlin / Heidelberg, 2010.

[110] A. Venkatesan. *A Study of Boosting based Transfer Learning for Activity and Gesture Recognition*. PhD thesis, Arizona State University, 2011.

[111] A. Venkatesan, N.C. Krishnan, and S. Panchanathan. Cost-sensitive boosting for concept drift. In *International Workshop on Handling Concept Drift in Adaptive Information Systems 2010*, pages 41–47, 2010.

[112] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18:77–95, 2002.

[113] Xiaojun Wan. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 235–243. Association for Computational Linguistics, 2009.

[114] Chang Wang and Sridhar Mahadevan. Manifold alignment using procrustes analysis. In *Proceedings of the 25th international conference on Machine learning*, pages 1120–1127. ACM, 2008.

[115] Liang Wang, Tao Gu, Xianping Tao, and Jian Lu. Sensor-based human activity recognition in a multi-user scenario. In *Ambient Intelligence*, pages 78–87. Springer, 2009.

[116] Zheng Wang, Yangqiu Song, and Changshui Zhang. Transferred dimensionality reduction. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 550–565. Springer Berlin / Heidelberg, 2008.

[117] B. Wei and C. Pal. Heterogeneous transfer learning with rbms. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[118] David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[119] Chen Wu, Amir Hossein Khalili, and Hamid Aghajan. Multiview activity recognition in smart homes with spatio-temporal features. In *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras*, ICDSC '10, pages 142–149, New York, NY, USA, 2010. ACM.

[120] Lin Xian-ming and Li Shao-zi. Transfer adaboost learning for action recognition. In *IT in Medicine Education, 2009. ITIME '09. IEEE International Symposium on*, volume 1, pages 659 –664, aug. 2009.

[121] Jun Yang, Rong Yan, and Alexander G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *Proceedings of the 15th international conference on Multimedia*, MULTIMEDIA '07, pages 188–197, New York, NY, USA, 2007. ACM.

[122] Q. Yang. Activity recognition: linking low-level sensors to high-level intelligence. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 20–25. Morgan Kaufmann Publishers Inc., 2009.

[123] Qiang Yang, Yuqiang Chen, Gui-Rong Xue, Wenyuan Dai, and Yong Yu. Heterogeneous transfer learning for image clustering via the social web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 1–9. Association for Computational Linguistics, 2009.

[124] Weilong Yang, Yang Wang, and Greg Mori. Learning transferable distance functions forhuman action recognition. In Liang Wang, Guoying Zhao, Li Cheng, and Matti Pietikinen, editors, *Machine Learning for Vision-Based Motion Analysis*, Advances in Pattern Recognition, pages 349–370. Springer London, 2011.

[125] Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1855–1862. IEEE, 2010.

[126] Z. Zhao, Y. Chen, J. Liu, and M. Liu. Cross-mobile elm based activity recognition. *International Journal of Engineering and Industries*, 1(1):30–38, 2010.

[127] Z. Zhao, Y. Chen, J. Liu, Z. Shen, and M. Liu. Cross-people mobile-phone based activity recognition. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[128] V.W. Zheng, D.H. Hu, and Q. Yang. Cross-domain activity recognition. In *Ubicomp*, volume 9, pages 61–70, 2009.

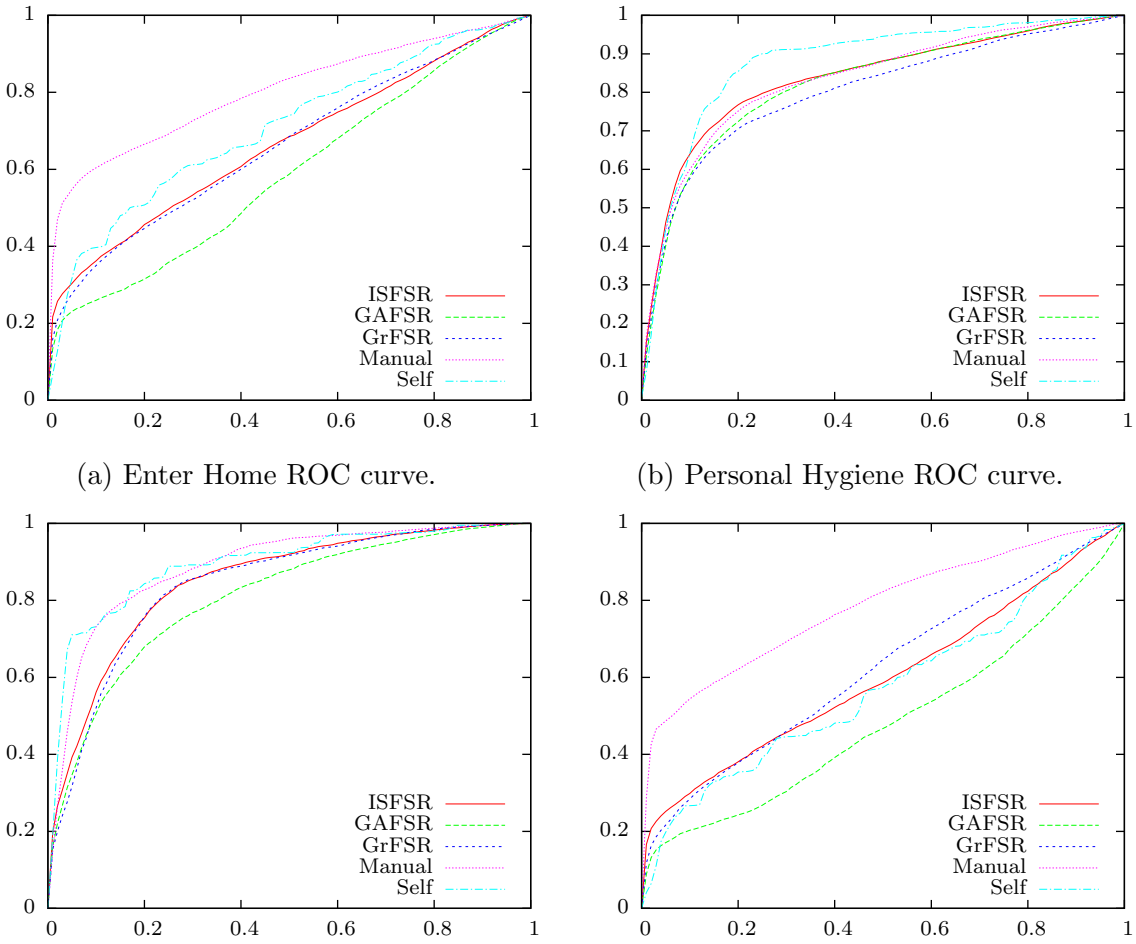
[129] V.W. Zheng, S.J. Pan, Q. Yang, and J.J. Pan. Transferring multi-device localization models using latent multi-task learning. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 1427–1432, 2008.

[130] Wenming Zheng, Xiaoyan Zhou, Cairong Zou, and Li Zhao. Facial expression recognition using kernel canonical correlation analysis (kcca). *Neural Networks, IEEE Transactions on*, 17(1):233–238, 2006.

[131] Erheng Zhong, Wei Fan, Jing Peng, Kun Zhang, Jiangtao Ren, Deepak Turaga, and Olivier Verscheure. Cross domain distribution adaptation via kernel mapping. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1027–1036. ACM, 2009.

Appendix A

Individual Class ROC Curves



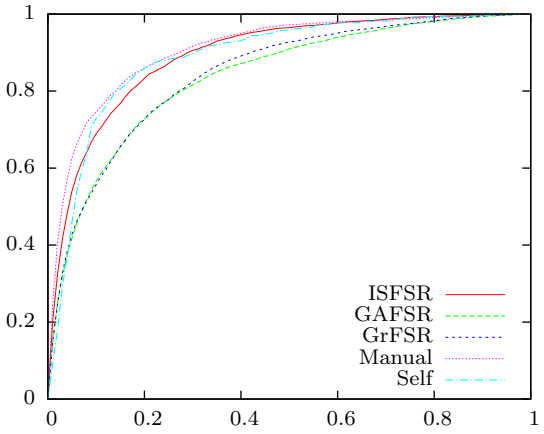
(a) Enter Home ROC curve.

(b) Personal Hygiene ROC curve.

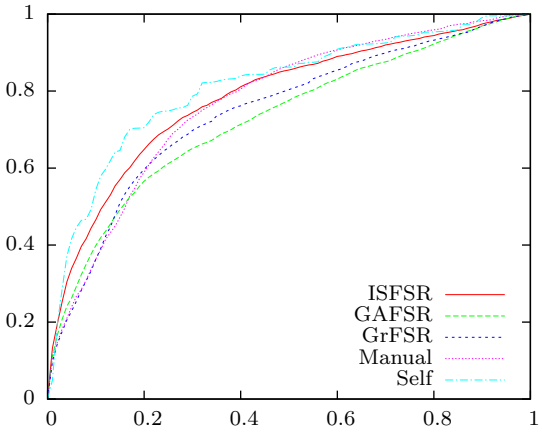
(c) Eat Lunch ROC curve.

(d) Leave Home ROC curve.

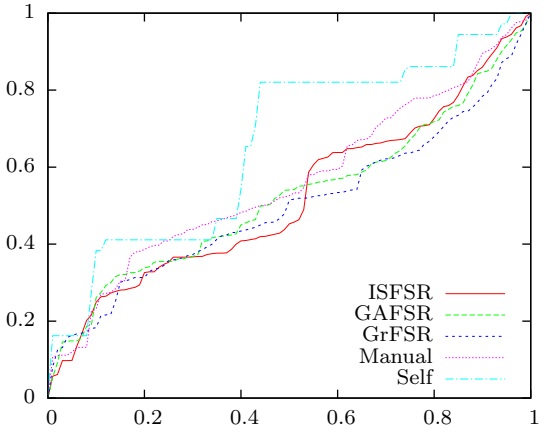
Figure 42: Individual Class ROC Curves



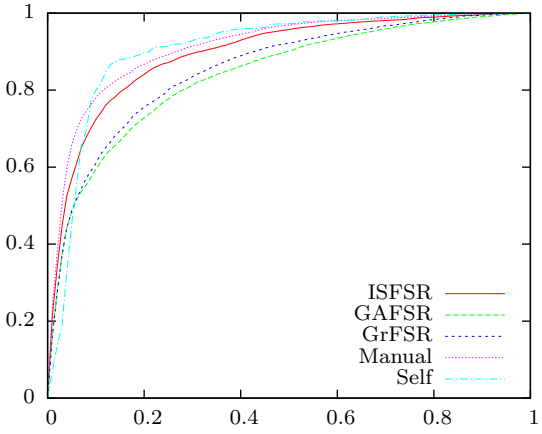
(a) Cook Dinner ROC curve.



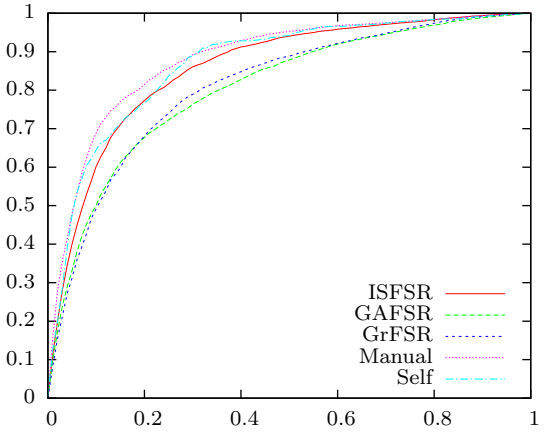
(b) Eat Dinner ROC curve.



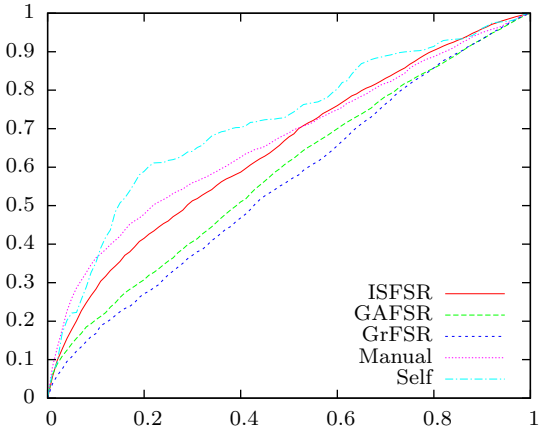
(c) Exercise ROC curve.



(d) Cook Lunch ROC curve.

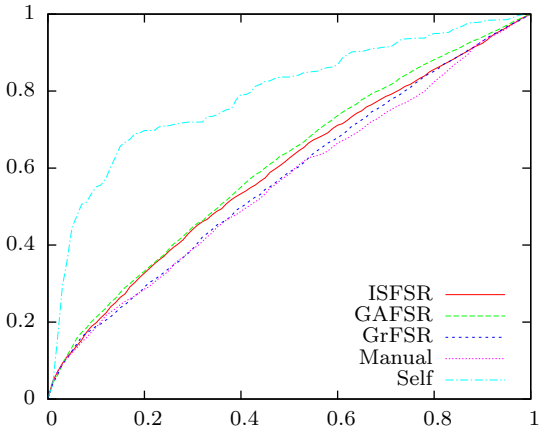


(e) Wash Dinner Dishes ROC curve.

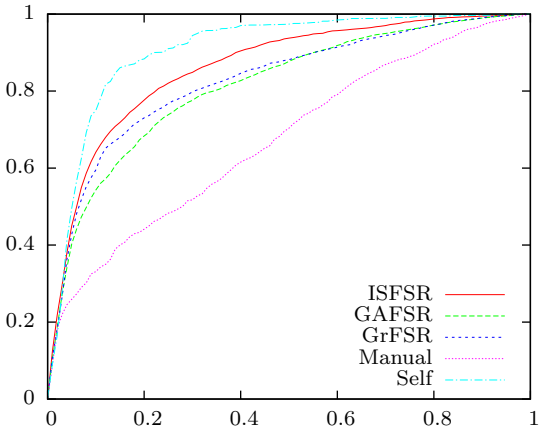


(f) Relax ROC curve.

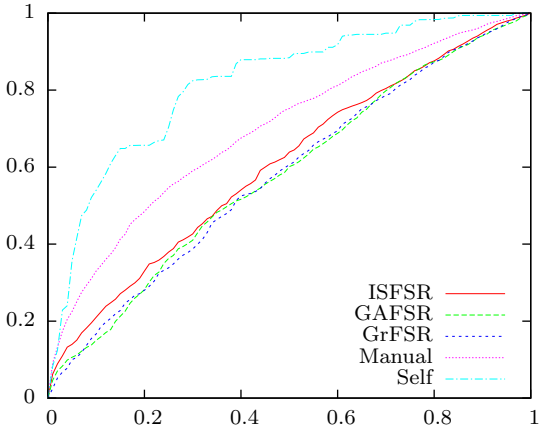
Figure 43: Individual Class ROC Curves (cont.)



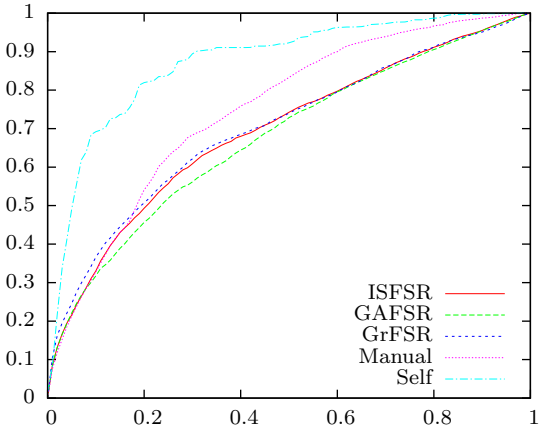
(a) Read ROC curve.



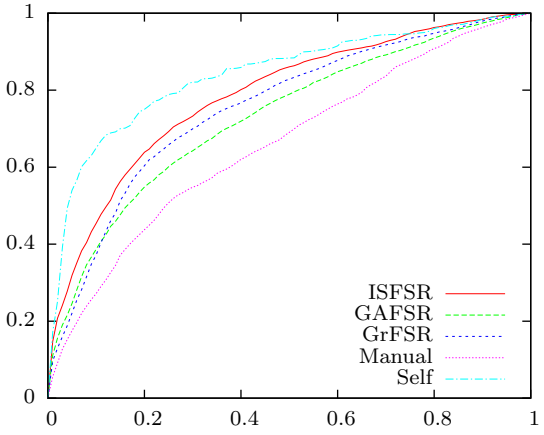
(b) Wash Lunch Dishes ROC curve.



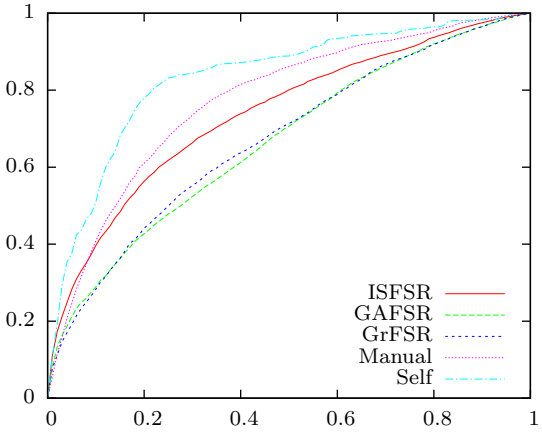
(c) Phone ROC curve.



(d) Evening Meds ROC curve.

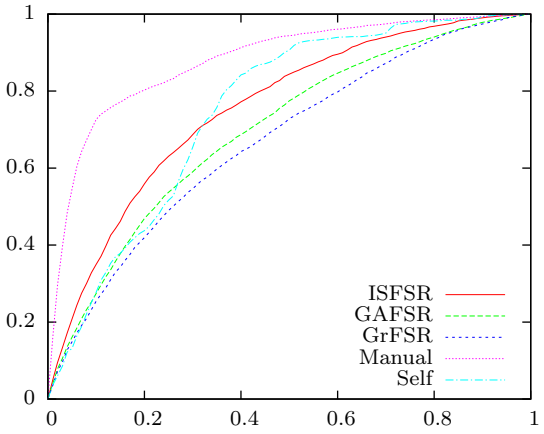


(e) Eat Breakfast ROC curve.

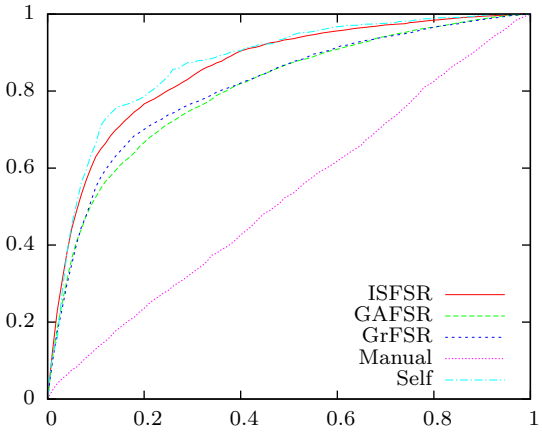


(f) Watch TV ROC curve.

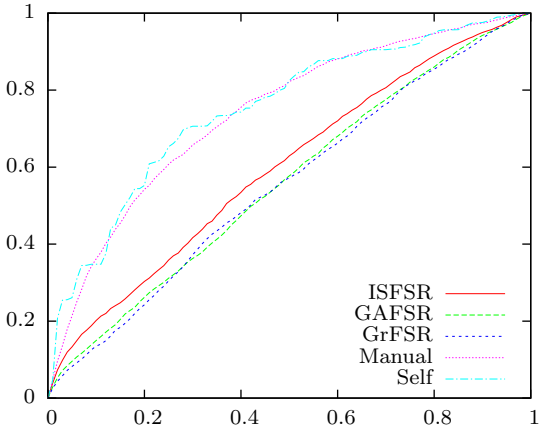
Figure 44: Individual Class ROC Curves (cont.)



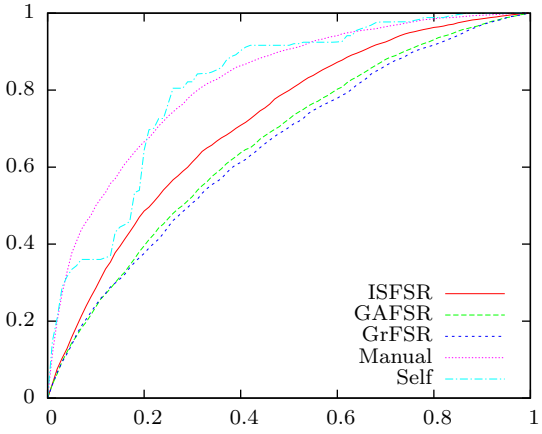
(a) Cook ROC curve.



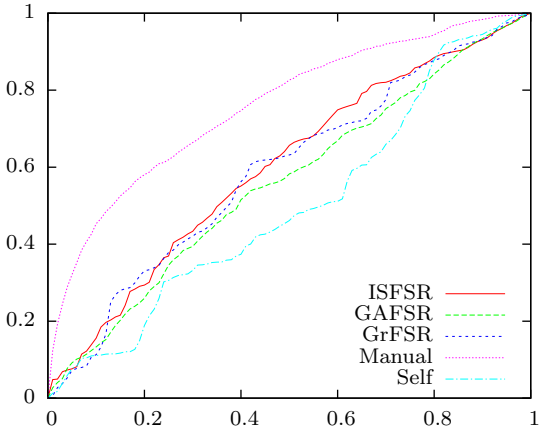
(b) Wash Breakfast Dishes ROC curve.



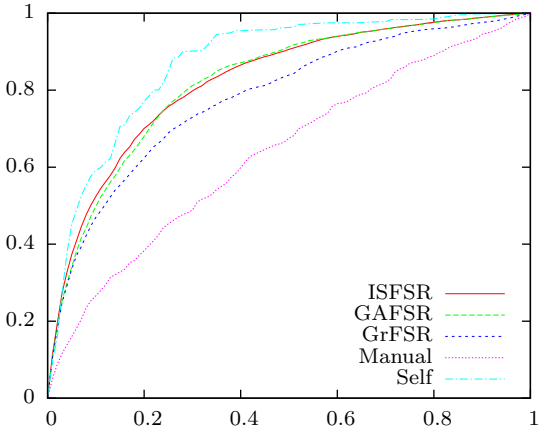
(c) Eat ROC curve.



(d) Wash Dishes ROC curve.

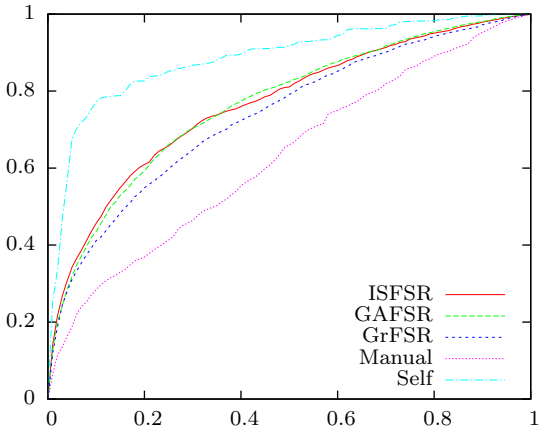


(e) Housekeeping ROC curve.

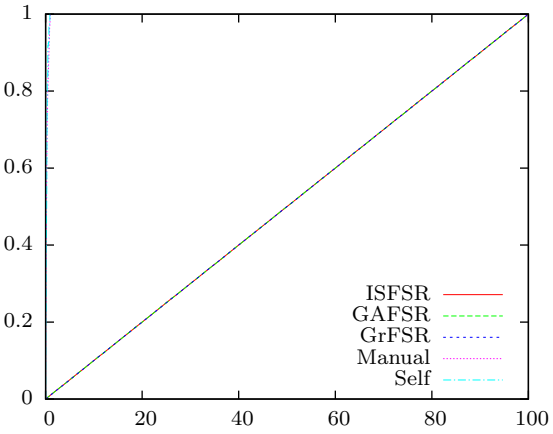


(f) Bathe ROC curve.

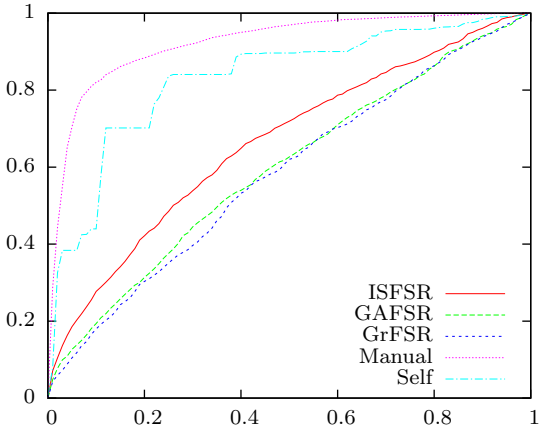
Figure 45: Individual Class ROC Curves (cont.)



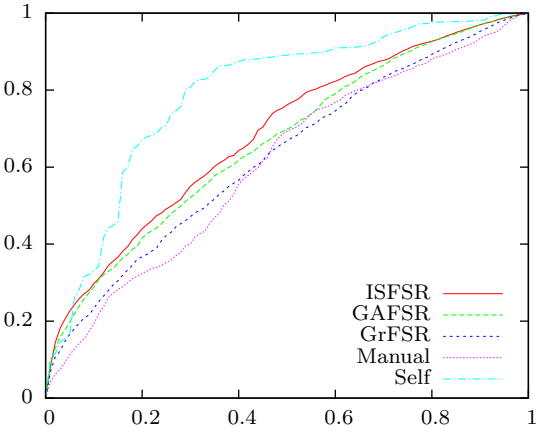
(a) Groom ROC curve.



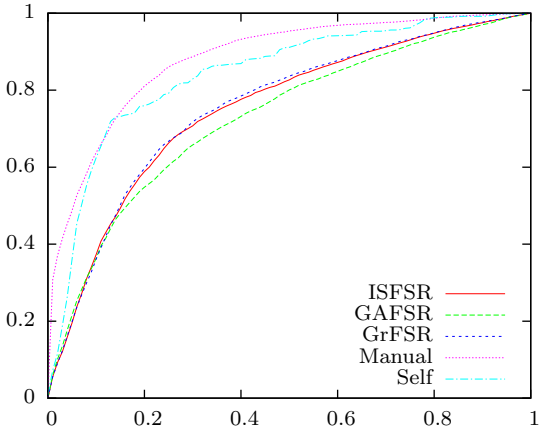
(b) Work at Desk ROC curve.



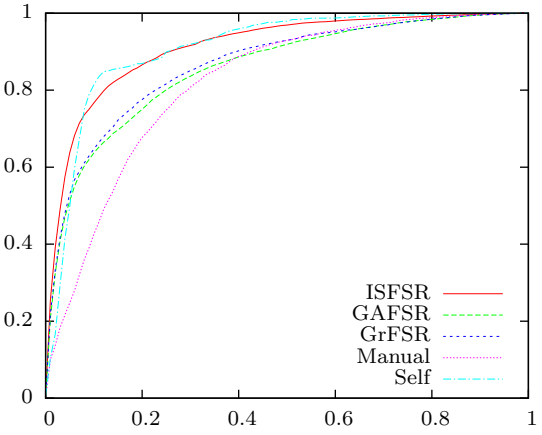
(c) Sleep out of Bed ROC curve.



(d) Work at Table ROC curve.

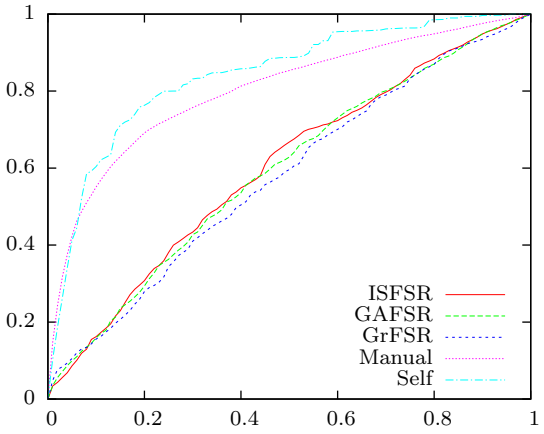


(e) Morning Meds ROC curve.

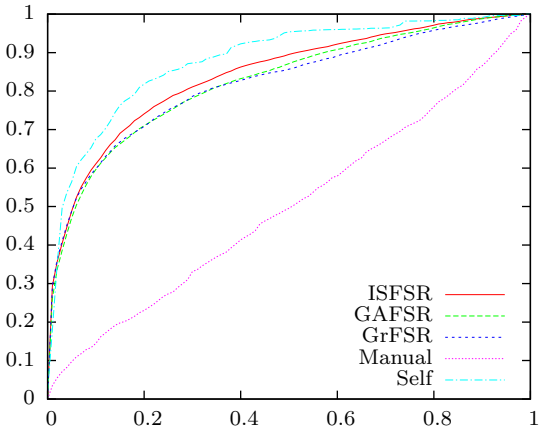


(f) Cook Breakfast ROC curve.

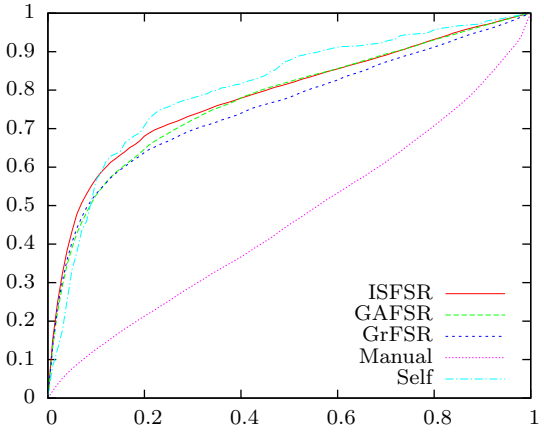
Figure 46: Individual Class ROC Curves (cont.)



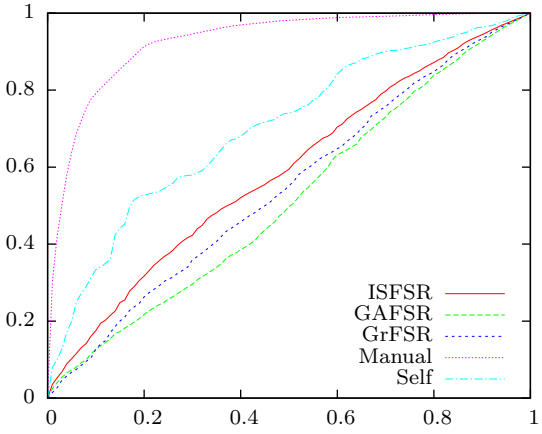
(a) Take Medicine ROC curve.



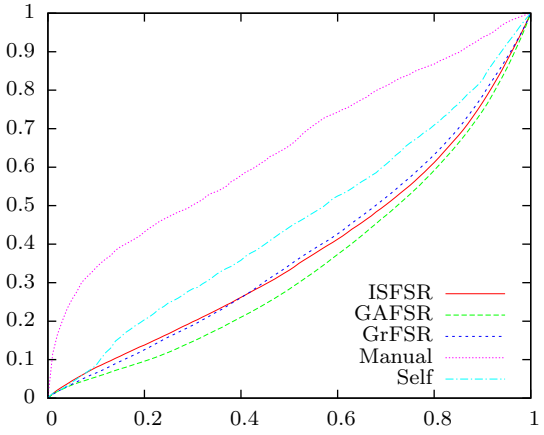
(b) Bed-Toilet Transition ROC curve.



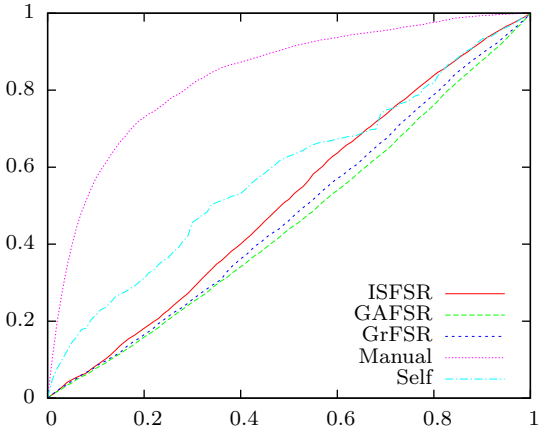
(c) Toilet ROC curve.



(d) Work ROC curve.

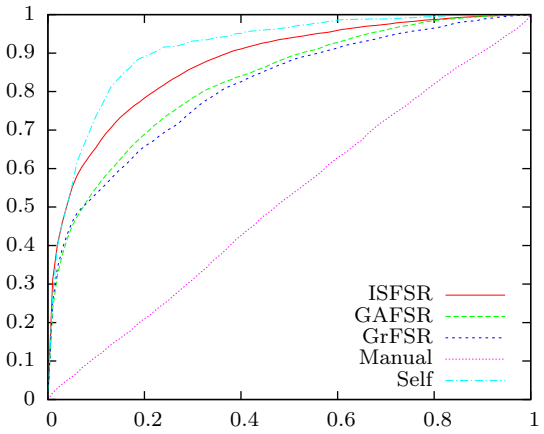


(e) Other Activity ROC curve.

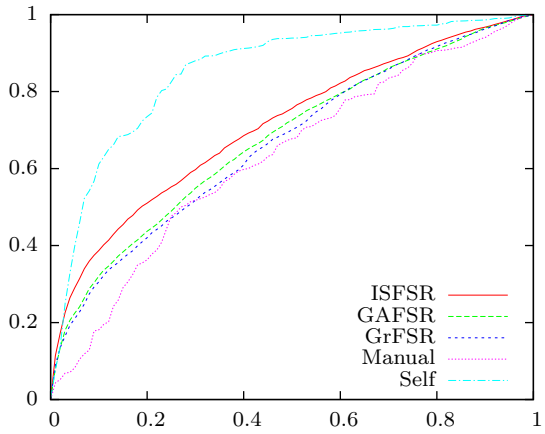


(f) Entertain Guests ROC curve.

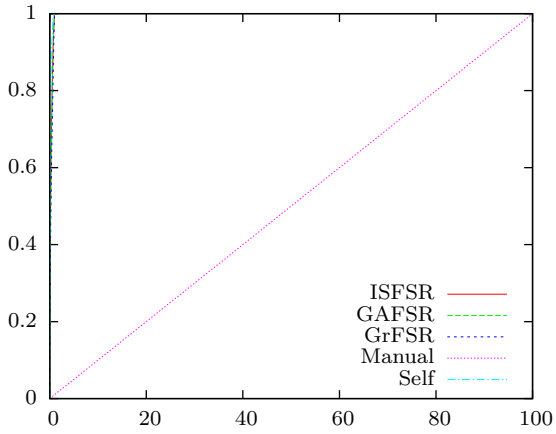
Figure 47: Individual Class ROC Curves (cont.)



(a) Sleep ROC curve.



(b) Work on Computer ROC curve.



(c) Dress ROC curve.

Figure 48: Individual Class ROC Curves (cont.)

Appendix B

Accuracy and Recall of the Source (Teacher) View

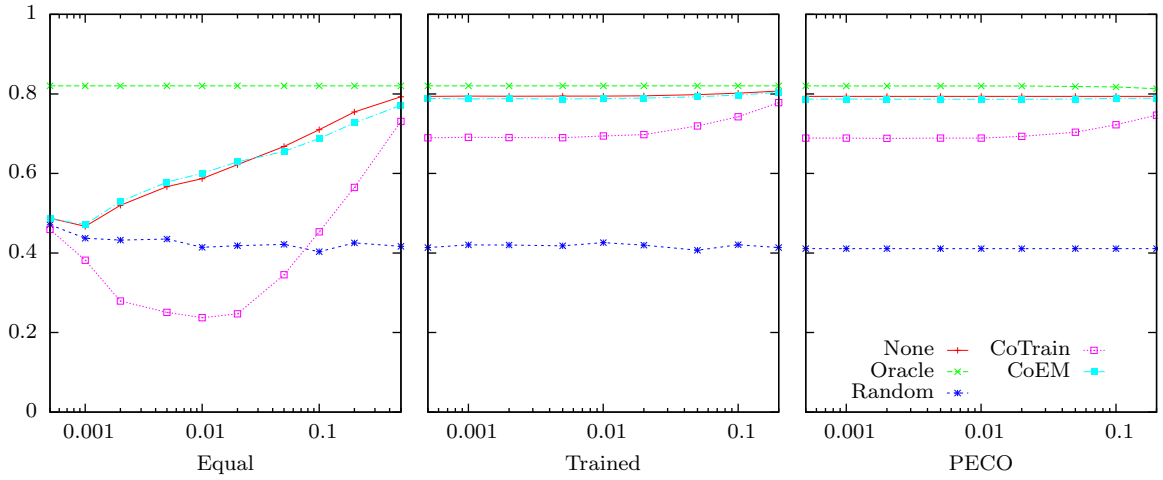


Figure 49: Teacher classification accuracy vs. labeled data using View 2 and View 3 where View 2 is the teacher. This corresponds to the learner accuracies in Figure 25.

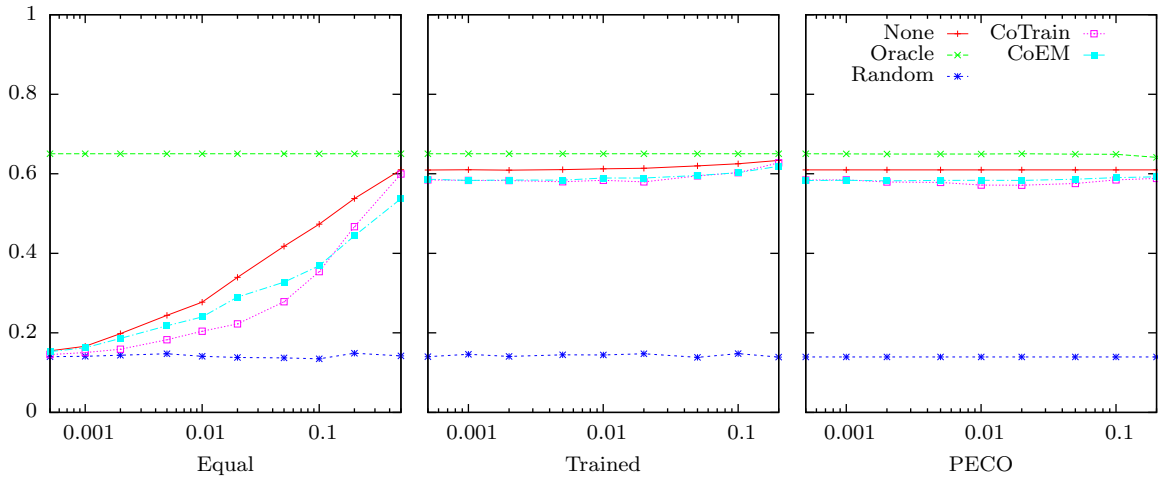


Figure 50: Teacher average recall vs. labeled data using View 2 and View 3 where View 2 is the teacher. This corresponds to the learner recall scores in Figure 26.

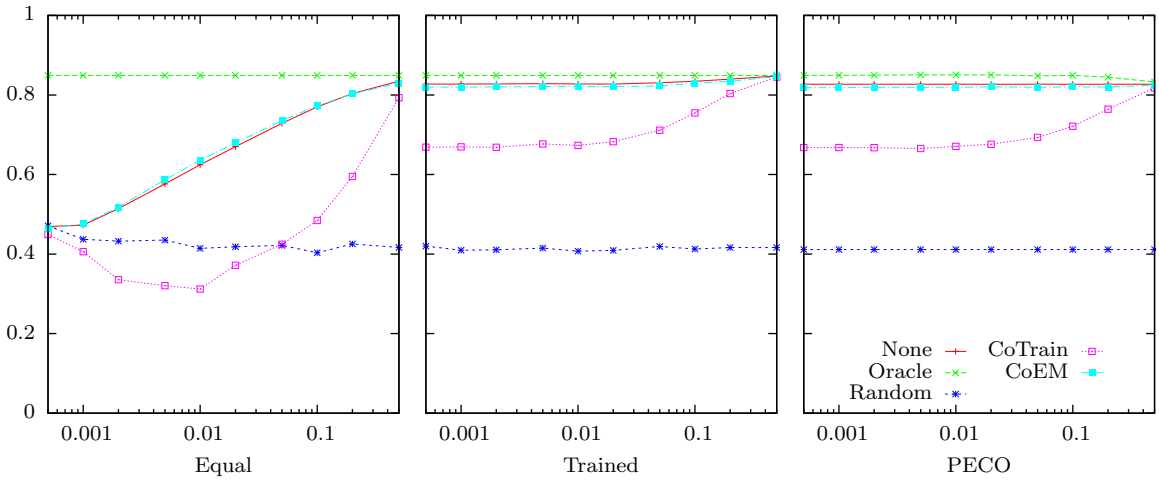


Figure 51: Teacher classification accuracy vs. labeled data using View 1 and View 2 where View 1 is the teacher. This corresponds to the learner accuracies in Figure 31.

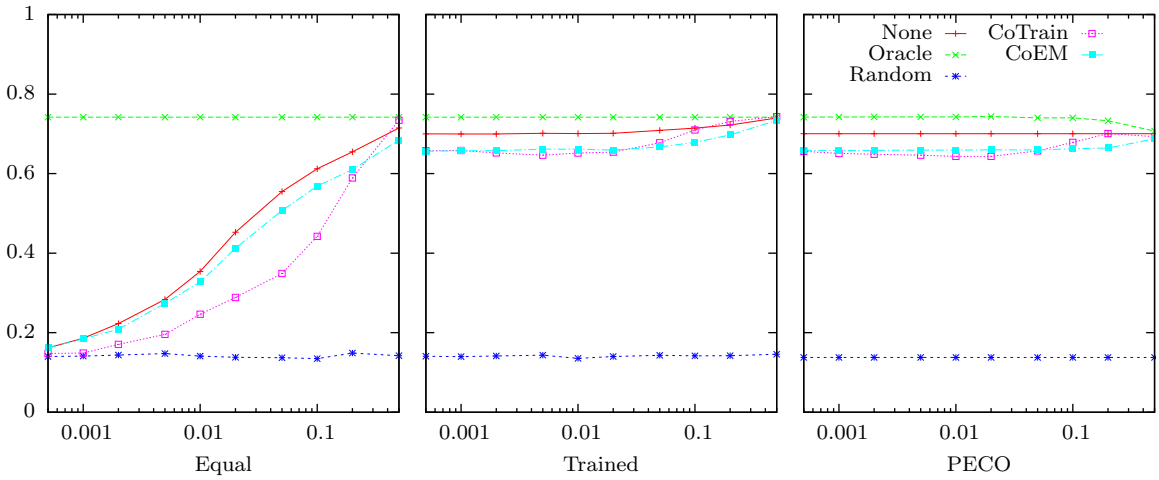


Figure 52: Teacher average recall vs. labeled data using View 1 and View 2 where View 1 is the teacher. This corresponds to the learner recall scores in Figure 32.

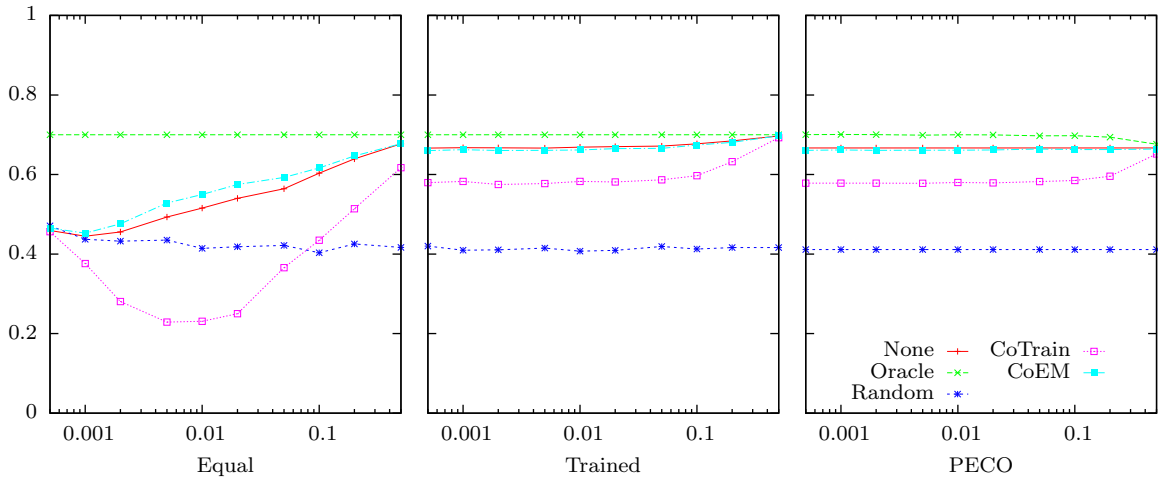


Figure 53: Teacher classification accuracy vs. labeled data using View 3 and View 2 where View 3 is the teacher. This corresponds to the learner accuracies in Figure 33.

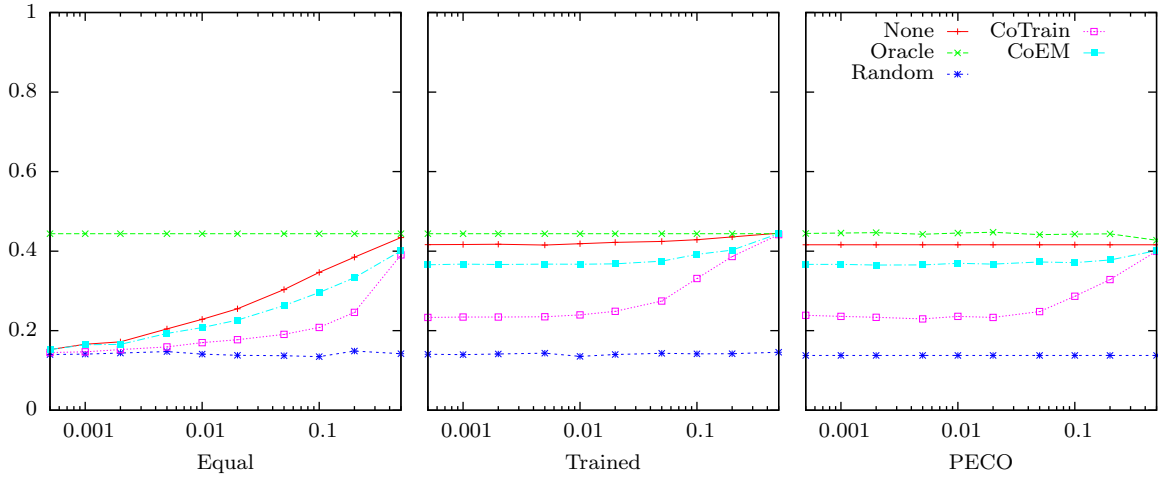


Figure 54: Teacher average recall vs. labeled data using View 3 and View 2 where View 3 is the teacher. This corresponds to the learner recall scores in Figure 34.