TRANSFER RULE LEARNING FOR BIOMARKER DISCOVERY FROM RELATED DATA SETS

by

Philip Ganchev

M.S., University of Pittsburgh, 2004

Submitted to the Graduate Faculty of

the Intelligent Systems Program in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2010

UNIVERSITY OF PITTSBURGH

INTELLIGENT SYSTEMS PROGRAM

This dissertation was presented

by

Philip Ganchev

It was defended on

December 10, 2010

and approved by

Vanathi Gopalakrishnan, Department of Biomedical Informatics

Robert Bowser, Department of Pathology

Shyam Visweswaran, Department of Biomedical Informatics

Fu-Chiang Tsui, Department of Biomedical Informatics

Dissertation Director: Vanathi Gopalakrishnan, Department of Biomedical Informatics

TRANSFER RULE LEARNING FOR BIOMARKER DISCOVERY FROM RELATED DATA SETS

Philip Ganchev, PhD

University of Pittsburgh, 2010

Biomarkers are a critical tool for the detection, diagnosis, monitoring and prognosis of diseases, and for understanding disease mechanisms in order to create treatments. Unfortunately, finding reliable biomarkers is often hampered by a number of practical problems, including scarcity of samples, the high dimensionality of the data, and measurement error. An important opportunity to make the most of these scarce data is to combine information from multiple related data sets for more effective biomarker discovery. Because the costs of creating large data sets for every disease of interest are likely to remain prohibitive, methods for more effectively making use of related biomarker data sets continues to be important.

This thesis develops TRL, a novel framework for integrative biomarker discovery from related but separate data sets, such as those generated for similar biomarker profiling studies. TRL alleviates the problem of data scarcity by providing a way to validate knowledge learned from one data set and simultaneously learn new knowledge on a related data set. Unlike other transfer learning approaches, TRL takes prior knowledge in the form of interpretable, modular classification rules, and uses them to seed learning on a new data set.

We evaluated TRL on 13 pairs of real-world biomarker discovery data sets, and found TRL improves accuracy twice as often as degrading it. TRL consists of four alternative methods for transfer and three measures of the amount of information transferred. By experimenting with these methods, we investigate the kinds of information necessary to preserve for transfer learning from related data sets. We found it is important to keep track of the relationships between biomarker values and disease state, and to consider during

learning how rules will interact in the final model. If the source and target data are drawn from the same distribution, we found the performance improvement and amount of transfer increase with increasing size of the source compared to the target data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

I want to express my heartfelt thanks to my academic advisor, Dr. Vanathi Gopalakrishnan, for her tireless encouragement, financial support, guidance, ideas and care over so many years. This thesis would not have been possible without her patience and protection.

It was Professor Bruce Buchanan who took me under his wing after my first advisor moved. He also introduced me to RL and, after he retired, he was my connection to Vanathi. I am deeply grateful for his inspiration, encouragement and care.

I am grateful to Dr. Robert Bowser who for several years generously funded my work that led to this thesis. I have learned much from the many discussions for which he was always so available. The opportunity to attend his research group meetings gave me invaluable understanding about mass spectrometry clinical studies. The data provided by the Bowser Lab are what motivated this thesis.

Many thanks go to Jiyan An for the countless clarifications about mass spectrometry and the ALS data; Dr. David Malehorn for providing the lung cancer data sets and for being so available for consultations about them; Dr. Kumar Kolli for providing the Breast Cancer data sets and funding part of my work.

My colleagues Mark Fenner, Eric Williams, Jonathan Lustgarten and Collin Lynch have provided many helpful ideas and suggestions over the years. Special thanks to Collin for his encouragement when the going was tough. My deepest gratitude goes to my brother Kuzman for many long discussions, invaluable ideas and moral support. Thanks to my parents for their regular nagging, unwavering encouragement and belief in me.

For my dear mother and father, my first teachers in this world.

1.0 INTRODUCTION

Biomarkers are a critical tool for the detection, diagnosis, monitoring and prognosis of diseases, and for the understanding of disease mechanisms in order to create treatments. A biomarker is any measurable indicator of a particular biological state of interest, particularly one relevant to the risk of contraction, the presence or the stage of disease [Rifai et al., 2006]. For example, a biomarker might be the concentration of a particular type of protein in the blood of a human subject or the presence of a particular gene in the tissue of the subject. Reliable biomarkers for a particular disease can be used to create screens for easy or early detection of the disease, which greatly improves the health outcome for the patient. Similarly, biomarkers can be used to monitor the disease progression and response to treatment. Another value of biomarkers is that they can be used in further research to understand the mechanism of disease, and ultimately create treatments. Therefore, the discovery of more accurate biomarkers has great utility to humankind.

The goal in biomarker discovery is to find a small set of biomarkers that can be measured and the measurements used together to accurately predict a biological state, such as a disease state. The discovery process typically begins with identifying the types of biological states that we seek to distinguish. This defines the groups of subjects that are used in the study, such as patients diagnosed with a particular disease, patients diagnosed with certain related diseases and healthy control subjects. It also helps to define the types of biological samples, such as blood or cerebrospinal fluid, that we will collect from each subject. Once the biological samples are collected, they are often then treated either with physical or chemical processes or with biological agents to prepare them for more accurate measurement. Variables associated with possible biomarkers are measured in the treated samples using molecular profiling. For example, the abundance of proteins in a particular range of masses

might be measured in each sample, creating a data set. The data set is analyzed using statistical techniques to find small subsets of the variables that discriminate between the groups of patients with high accuracy. Finally, these putative markers are linked to bio molecules such as proteins or genes.

There are a wide variety of technologies and techniques that are used for the measurement and the associated preparation of samples. For molecular profiling, those include:

- Matrix-assisted laser desorption-ionization time-of-flight mass spectrometry (MALDI-TOF MS),
- Surface-assisted laser desorption-ionization time-of-flight mass spectrometry (SELDI-TOF-MS),
- Gene expression profiling using DNA micro-arrays, and
- Immunoassays.

In the statistical analysis, computational variable selection aims to find small sets of variables that can discriminate well between the groups of instances. Machine learning is used to estimate how well those variables can discriminate between the groups. This is done by learning a classifier using those variables, and applying that classifier on unseen data.

1.1 THE PROBLEM

The statistical analysis of the data aims to find a small subset of variables that can discriminate between the groups of samples. However, this is particularly challenging because of several problems that underlie all biomarker discovery studies:

1. **Small data sets.** From a statistical point of view, the number of biological samples available for a clinical molecular profiling study is relatively small, typically in the tens or hundreds. This is because it is difficult to recruit suitable patients and control subjects matched by age, gender and other possibly important variables. It is also expensive to draw and store samples and process them to measure the potential biomarkers.

2. **Many uncontrolled variables.** Many variables among the subjects in a study that are possibly important, such as age and exposure to harmful chemicals, often are not properly controlled. This is because there are so many such variables that controlling all of them would require a very large number of subjects. Moreover, the exact variables to control for are not known in the first place.
3. **High dimensionality.** Because the mechanisms of the human body are so complicated, and because scientific understanding of them is currently incomplete, the number of measurable quantities that are candidates for biomarkers is enormous. In order to broaden the search for biomarkers, studies typically measure a wide range of variables for the biological samples. For example, proteomic mass profiling studies using high-throughput techniques such as SELDI-TOF and MALDI-TOF typically measure hundreds of thousands of m/z values. Gene expression studies measure the expression of thousands or tens of thousands of genes. This problem is smaller when using technologies such as immunoassays, where typically tens of potential markers are evaluated, but the other problems still exist.
4. **Measurement error.** Technologies such as SELDI-TOF and MALDI-TOF have an associated measurement error. This includes random errors due to measuring small quantities and physical limitations of the apparatus, as well as systematic errors that can increase or decrease certain measurements within a data set.

All these aspects make the statistical analysis prone to error, because among all the measured variables it is likely that by chance alone, some of them will appear to discriminate among the small number of samples. Such variables will not correspond to real biomarkers, that is, they will not generalize to the whole population.

On the other hand, there are often a number of similar small studies that examine the same population groups of subjects using the same technologies. Often this is due to having a pilot study with a small number of patients, and follow-up studies with additional patient samples or new patients. This results in having two or more related data sets. We consider data sets to be **related** if they represent the same variables. Then the question is, how to best use the combined information from these multiple data sets to discover more accurate biomarkers and classifiers?

1.1.1 Current Methods

In order to combine the information from multiple data sets, researchers typically analyze each data set separately, then compare the biomarkers discovered. Unfortunately, this might be sub-optimal because it does not consider the whole of the information in the two data sets at the same time. That is, there is no interaction between the data sets.

Another simple way to use all the data is to use the union of two data sets and analyze the whole set in the usual way. But such attempts are typically confounded by variability in sample processing and by systematic measurement error specific to each data set. For example, the same numerical measurement might mean a high abundance of some protein in one data set but a low abundance in another. Thus the variable will not appear discriminative in the union of the data sets, even if it actually is.

1.2 THE APPROACH

This thesis explores one approach to making better use of available data for biomarker discovery. The approach is an instance of the paradigm of transfer learning (also called "inductive transfer"). Transfer learning is the use of knowledge about one task to help in learning another task [Caruana, 1997]. In particular, this thesis uses a form of sequential transfer learning, where information is learned from one data set, the **source data**, and is used when learning on another data set, the **target data**.

Thus the aim of using the available data sets is translated to the following questions:

1. How to validate the existing information with new data?
2. How to learn new accurate information in conjunction with the old?

Current transfer learning methods are ill-suited for the small, high-dimensional data sets often encountered in biomarker discovery. Many of those methods learn a model using all the available variables and do not help in finding the most discriminative variables. Many of the models, such as ones that are based on artificial neural networks (ANNs) and Bayesian

belief networks require many training examples to learn an accurate classifier. ANNs also and require a long training time.

Instead, to address the questions above, the thesis develops a novel framework for transfer learning using classification rule learning: the Transfer Rule Learner framework (TRL). The framework is based on the rule learning algorithm Rule Learner (RL), which has been successfully used in biomarker discovery and verification studies for early detection of

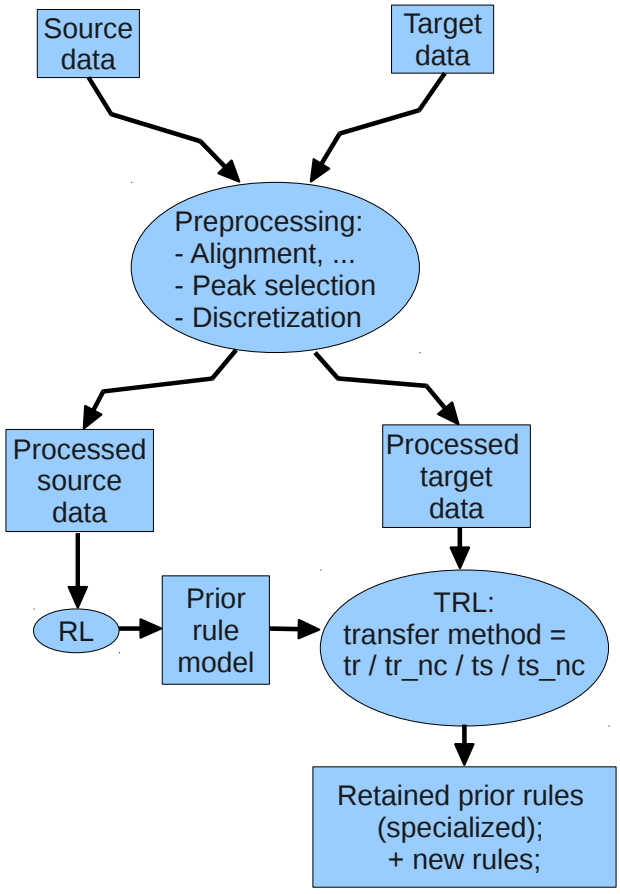


Figure 1: The Transfer Rule Learner (TRL) framework. The Rule Learner (RL) algorithm extracts prior rules, which are input into the transfer rule learning algorithm introduced in this thesis. "tr" is simple rule transfer; "ts" is structure transfer; "tr_nc" and "ts_nc" are the non-coverage variations of simple rule transfer and structure transfer.

amyotrophic lateral sclerosis (ALS) [Ranganathan et al., 2005, Gopalakrishnan et al., 2006, Ryberg et al., 2010]. RL learns classifiers that are sets of propositional rules (for an example, see Section 2.8). It has some advantages over other learners, including simplicity, understandability by domain experts and modularity of its learned classification models. RL requires discrete variable values, and several discretization algorithms are already available in the implementation. However, discretization provides a unique challenge for transfer learning, and TRL provides two methods to address that challenge.

The TRL framework is illustrated in Figure 1. First, the source and target data may optionally be preprocessed to aid in the transfer; then both data sets are discretized. Then the prior classification rule model is learned on the source data and is used in learning the target model from the target data. The transfer learning is the central part of the framework, and is implemented with four algorithms. These algorithms are differentiated by two aspects: (a) the type of information transferred and (b) the way prior rules interact with new rules during learning. Thus for the type of information transferred, TRL has two options:

1. Simple rule transfer (tr), which transfers entire rules, versus
2. Structure transfer (ts), which transfers only the sets of variables but not their values

And for the interaction of rules TRL has two options:

1. With coverage and on the beam
2. Non-coverage (nc), only the beam

The framework is evaluated by its classification performance and 3 novel measures of amount of transfer. Using these measures, the feasibility of the four methods is evaluated on 13 pairs of real-world clinical data sets from biomarker profiling or verification studies. Classification performance is compared to the baseline condition of learning on the target data set alone.

We are now ready to state the thesis.

1.2.1 Thesis

The central thesis of this dissertation is that the TRL framework is sufficient for evaluating the effect of transfer between two related biomarker discovery data sets. Based on the

experiments performed on 13 pairs of real-world molecular profiling data sets, we make the following claims.

General claims:

1. TRL provides a mechanism for transfer learning between two related data sets that increases classification performance on the target data much more often than not, compared to learning without transfer.
2. The change in performance after transfer and the amount of transfer are highly variable and depend on the data sets used for transfer.
3. When the source and target data are drawn from the same distribution, for all transfer methods, *both* the classification performance and the amount of transfer depend on the accuracy of the source model and baseline target model:
   a. The more accurate the source model compared to the model learned on the whole data alone, the greater the improvement in accuracy and the greater the amount of transfer.
   b. When the target data is too inaccurate, there is very little or no transfer and very little change in performance.

Specific claims:

4. With TRL under resource constraints, variable values are important information to transfer.
5. With TRL under resource constraints, it is important during learning to take into account the interaction that will occur between prior rules with newly learned rules during inference.

1.3 SIGNIFICANCE

To my knowledge, this work is the first study of transfer learning for biomarker discovery. This is an important problem because of the difficulty and expense of collecting biological data sets, and because in some cases multiple related data sets for a discovery

task are available but it is not clear how to best use them to discover accurate biomarkers. Previous studies suggest that transfer is possible because they find reproducibility of information learned from spectra collected in different sessions using the same protocols [Pelikan et al., 2007, Semmes et al., 2005].

Unfortunately many previous transfer learning frameworks produce models that are difficult to interpret and that use a large number of variables [Caruana, 1997, Blitzer, 2008], making them less useful as a method for the discovery of biomarkers, as described in Chapter 2. The methods we consider are based on the Rule Learner algorithm [Clearwater and Provost, 1990], and they produce classification rule models. Rule models have the advantage that variable selection is embedded in the learning algorithm, and the resulting model uses only a few of the many measured variables to explain the data, unlike artificial neural networks. Also, unlike artificial neural networks, they are understandable by domain experts and can be used to form biological hypotheses for further experimentation.

This thesis develops a transfer rule learning framework, TRL, which includes four variations of transfer learning and measures for evaluation of the transfer.

Two learning tasks are considered related if there is a mapping from the variables in the source task set to the variables of the target task. As currently implemented, TRL uses only the trivial mapping of equivalence between source and target variable. However, by defining other mappings, the framework allows more general transfer between different types of data.

TRL is evaluated on 26 real world clinical data sets (13 data set pairs), focusing mostly on SELDI-TOF and MALDI-TOF data partly because those were the most available. We also include experiments in pre-processing the data with the aim of improving the transfer.

In addition to introducing transfer learning for the domain of biomarker discovery and a framework capable of performing transfer learning in this domain, we also investigate what kinds of information it is important to preserve from the source data set in order to best learn a model for the target data set. This contribution can guide future research on transfer learning for biomarker discovery, even under different frameworks. Specifically, we find in Chapter 5 that it is important to preserve the information about how biomarkers relate to disease state in the source data set. When this information is suppressed, and all

possible relationships are considered for the target data set, performance declines (Claim 4). Secondly, it is important to consider how prior information interacts with new information learned on the target domain. When we omit these interactions during learning, the effect of transfer learning on performance decreases (claim 5).

This gives rise to many research questions that can be addressed in future work. See Chapter 6 – Conclusion.

1.4 DISSERTATION OVERVIEW

The rest of this document is organized as follows. Chapter 2 provides background information about the concepts and techniques used in this thesis and other relevant techniques; this includes biomarkers, classification learning, variables and variable selection methods, rule learning with RL, and a quick overview of the transfer learning literature and current methods. Chapter 3 presents in detail the Transfer Rule Learning (TRL) framework, which was implemented and evaluated in this thesis as a solution to the problem described in 1.1 above. Chapter 4 describes the experiments done to evaluate the TRL framework, including the data sets used, the baseline used for comparison, and the evaluation measures. Chapter 5 presents the major results from the experiments with the real-world data sets. Chapter 6 presents conclusions and discusses further development of the methods presented here.

2.0 BACKGROUND

This section introduces the techniques relevant to the thesis. This includes an introduction to biomarkers and the particular importance of protein biomarkers.

Most of the data sets used to evaluate TRL are proteomic mass spectra. Therefore this chapter gives an overview of the technique used for generating that type of data, mass spectrometry, as well as the pre-processing of mass spectra that before statistical analysis. Then this chapter gives an overview of variable selection, including EBD, which is the method for discretization and selection of variables used in experiments presented in this thesis to evaluate TRL.

After that, the chapter introduces Rule Learner, the machine learning method that was used in the experiments. Finally, we overview the field of transfer learning, which is the overall approach taken in the thesis.

2.1 BIOMARKERS

A biomarker is a measurable indicator of a specific biological state, particularly one relevant to the risk of contraction, the presence or the stage of disease. Thus, biomarkers allow easy or early detection of a disease or monitoring of disease progression, and provide a factor measurable across populations.

Early search for biomarkers focused on genes, but more recently shifted to proteins. Protein biomarkers can be much more sensitive and specific to particular diseases than gene marker, because there are many more proteins than genes. This is because proteins are produced in complex processes that begin with genes but have intermediate steps that are

affected by various factors such as differential splicing of the mRNAs representing the proteins, posttranslational modifications of the proteins, and temporal and functional regulation of gene expression [Anderson and Anderson, 2005]. Thus, a disease process that alters intermediate steps cannot be detected as a gene biomarker. Disease processes change the protein constitution of the diseased cells, through altered gene expression, differential protein modification, changes in specific activity and aberrant localization, all of which may affect cellular function. Proteomic techniques allow such protein changes to be identified.

2.2 PROTEOMIC MASS SPECTROMETRY

One technique that has become very popular recently for discovery of proteomic biomarkers has been whole-sample mass spectrometry. Mass spectrometry (MS) is a technique for analyzing substances by measuring the relative concentrations of their molecules and molecular fragments. In the search for protein biomarkers, MS is used to analyze the proteins and peptides in biological samples. High-throughput, whole-sample MS techniques measure a large amount of data for each sample, and are used as a relatively cheap and fast way to search for biomarkers by their approximate mass. Although this data is relatively inaccurate, it is useful for finding potential biomarkers.

Such high-throughput techniques include matrix-assisted laser desorption and ionization (MALDI) time-of-flight (TOF) and surface-enhanced laser desorption and ionization (SELDI) TOF. The techniques are illustrated in Figure 2. In TOF MS, the molecules in the samples are separated by mass and their abundance is recorded. Specifically, in MALDI and SELDI, the molecules are converted to gaseous ions by the energy of a focused laser beam. In MALDI, the energy indirectly reaches the analyte, after being converted to heat by energy absorbing compounds called a matrix. Before ionization, the samples are mixed with an energy-absorbing chemical called a matrix [Sem, 2007, p 103]. The ions fly into a vacuum and are subjected to an electrical field, which exerts a force on them proportional to their charge. Lighter ions accelerate faster than heavier ones, and so reach the end of the vacuum tube sooner. There, a detector periodically records the accumulated charge caused

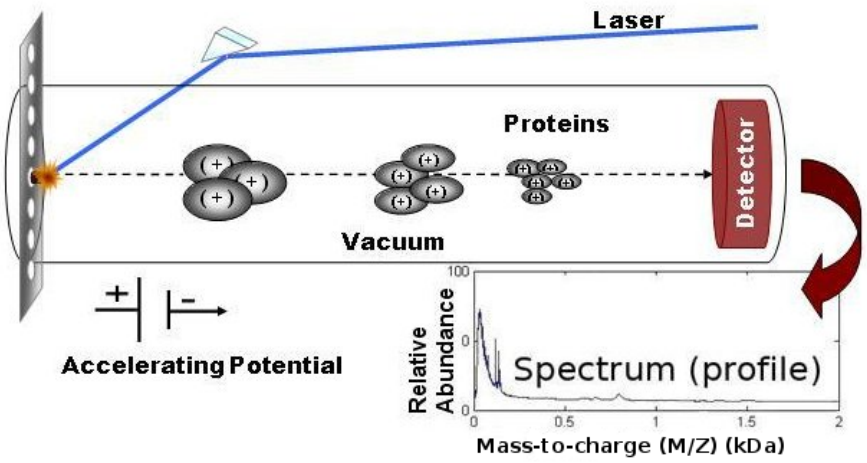


Figure 2: The working of a time-of-flight mass spectrometer. (Adapted from [Ranganathan, 2004].)

by the ions. The time-of-flight ranges are then converted to approximate mass-to-charge ratio (m/z) of ions, and the accumulated charges are recorded as the intensity for the particular m/z range. The intensity then represents the their approximate relative abundance of particles (proteins) whose mass is in that range.

The collection of the measured intensities for all the m/z's is then a mass spectrum or mass profile. It can be plotted with m/z on the x-axis and relative abundance on the y-axis. The mass is measured in Daltons (Da), defined as 1/12 of the mass of an unbounded carbon-12 nucleotide at rest and in its ground state, or approximately 1.6605 exp -27 kg. The set of spectra generated in one batch can then be used to compare the intensities for a given m/z across all the spectra. In the discovery process, we want to find m/z's for which one group of individuals (e.g. lung cancer patients) have consistently higher intensities than other groups. This is the function of supervised feature selection and machine learning methods. However, because the intensities represent *relative* abundances, we cannot compare the absolute intensity of a spectrum from one batch with that of a spectrum from another batch.

SELDI is a modification of MALDI that aims to address the challenges of separation, purification and detection of proteins by MS [Sem, 2007, 102]. In particular, it helps to separate complex mixtures of analytes into simpler mixtures, which are easier to analyze. This is done by coating the chip array surface with a chemical that binds to specific functional groups. Thus, analytes that have those functional groups are adsorbed on the chip surface and processed with MALDI, while the others are washed away during the sample preparation process. Using different surface chemicals allows the selective analysis of different groups of proteins and peptides. Also, this technique allows contaminants, such salts and detergents, which interfere with the creation of ions, to be removed. Commercially, SELDI has been realized in spectrometers and protein chips manufactured by Ciphergen Biosystems Inc.

2.3 BIOMARKER DISCOVERY

A study aiming to find biomarkers for particular biological states begins by defining the population to be studied and subgroups in it, which are to be distinguished by the biomarkers. For example, there may be two groups: women who suffer from breast cancer and healthy women. Then, clinical samples, such as blood plasma, are obtained from individuals in each group. One or more spectra are created from each sample and analyzed using machine learning (ML) techniques to search for combinations of m/z's that distinguish samples of a given group. The spectra are usually first pre-processed in various ways to reduce the measurement errors and ease the search for biomarkers. Candidate markers are considered to be represented by variables. For example, for mass spectrometry data, variables are either individual m/z's or regions of m/z's such as areas or intensities of peaks found after peak selection. Sets of variables are selected using variable selection (described below), and evaluated by training and testing classifiers to discriminate between the groups.

A major obstacle to the analysis is that MS data sets typically contains few instances but many variables. Usually only on the order of 100 samples are available for study, due to the scarcity of individuals of the groups of interest and the cost of obtaining samples. By contrast, on the order of 10,000 m/z measurements are made because any one of them may

provide biological information. This scarcity of instances makes the learning problem very underspecified and therefore difficult.

This is made worse by measurement errors in creating the mass spectra. There are both systematic errors (biases) and random variations, in both mass and abundance. For example, having many peptides of very similar mass can cause errors in reading the mass and abundance.

Dimensionality reduction, variable selection and variable construction are fundamentally tied to learning, because using the correct variables makes learning easy, and finding the correct variables requires learning. But variable selection is central to MS analysis, because the goal is a small but discriminative set of biomarkers. Moreover, to be useful in this aim, a predictive model and its variables must be interpretable in a biological context. Therefore, a black-box predictive model is less useful, as are dimensionality reduction techniques such as principal component analysis which map the original variables to new variables is difficult to interpret. Of course an accurate predictive model that generalizes to other spectra is desirable but usually too difficult to obtain, due to the difficulties with the data.

2.4 PRE-POSSESSING OF SPECTRA

Mass spectra are usually pre-processed before machine learning is applied, to reduce the effects of random variation and bias in the measurements. Typical operations are intensity normalization, baseline correction, smoothing, peak finding, and alignment.

Intensity normalization tries to correct an error in intensity (abundance) measurement throughout the sample. One method is the use of an internal calibrant substance mixed in the same concentration in all samples (Wong 2006). After the spectra are created, each one is scaled linearly so that the intensity corresponding to the molecules in the calibrant are equal. Another common approach to intensity normalization is total ion current (TIC) normalization. This assumes that all the samples contain the same total concentration of the molecules, and so the total charge that flows to the detector (the total current) should be equal in all spectra. Thus the each spectrum is rescaled by multiplying its intensity values

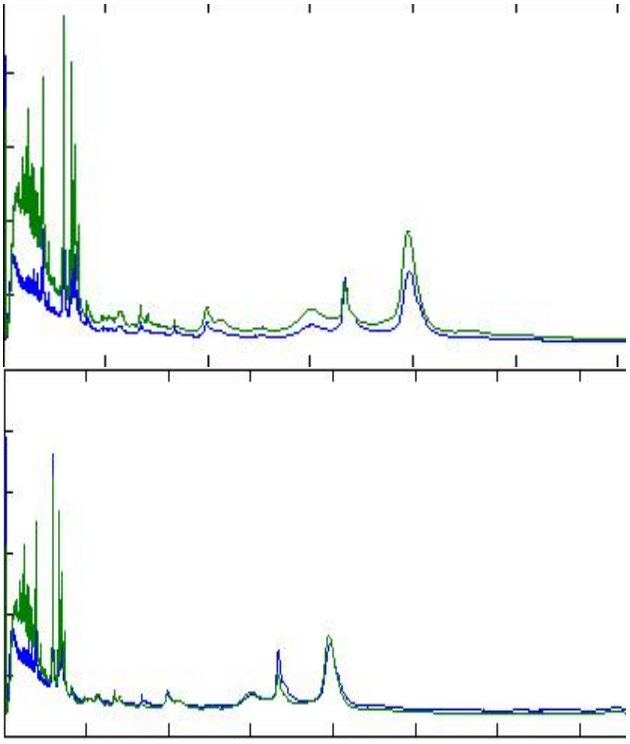


Figure 3: Two spectra before and after TIC normalization.

by the average signal for all spectra and dividing by the total signal for the spectrum.

Finding peaks in the spectra is used in some other pre-processing operations, and often as an operation in itself because it achieves a dramatic reduction of the variable space. Ideally, all peaks representing a molecule should be found, while peaks due to noise or random ions ignored; but in practice this is not guaranteed. The peaks are assumed to be common among all spectra. Morris et al. (2005) detect peaks in the average spectrum of the spectra in the data set. The averaging reduces the noise, so detecting the true peaks is more reliable. Peaks found in the mean spectrum are used to find peaks in individual spectra. The peaks in the average spectrum are defined as those that form a local maximum in a window and satisfy other criteria such as relative height and minimum intensity. For each sample spectrum, peaks are assigned as the closest local maximum that is closest to a local maximum in the average spectrum. If two peaks map to one maximum, the farther peak is mapped to another

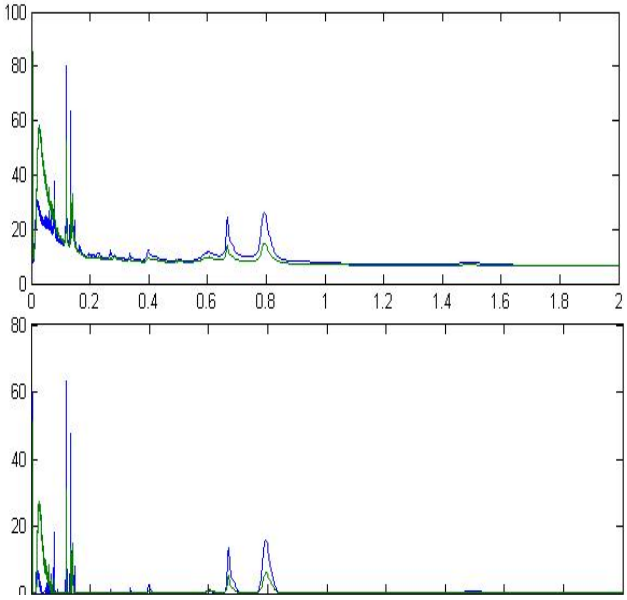


Figure 4: Two spectra before and after baseline correction.

maximum.

Baseline correction aims to correct intensity errors that occur in a portion of a spectrum due to saturation of the spectrometer and background signal due to matrix. Because matrix ions have low mass, the spectra typically have a high peak there, which tapers. This baseline should be removed to leave intensity due to molecules of interest (illustrated in Figure 4). There are many methods of detecting the baseline. A popular method is that of [Coombes et al., 2003]. First, peaks are found and removed from each spectrum, then the baseline is estimated from the modified spectrum using the moving average of local minima. The size of the moving window determines the sensitivity of the baseline estimation.

Smoothing and de-noising aim to reduce small signals due to noise but preserve true signals (Figure 5). There are many smoothing algorithms, including moving average, and fitting a smooth function inside a moving window. Fitting a function tends to better preserve relative maxima, minima and width in the spectrum [Wong et al., 2005]. A simple and popular fitting algorithm for smoothing is the Svarsky-Golay algorithm, which fits a polynomial using a least-squares regression. Similar to smoothing, de-noising aims to remove noise

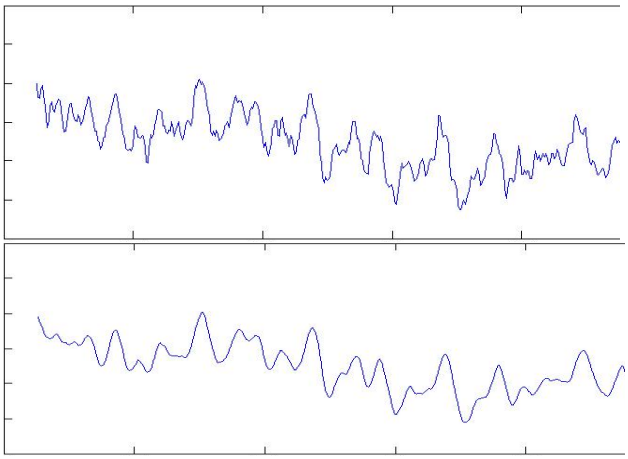


Figure 5: A spectrum before and after smoothing.

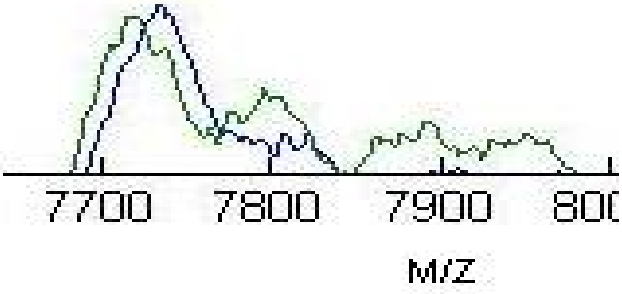


Figure 6: A part of two spectra, showing a possible mis-alignment

completely. De-noising is often done by wavelet transform, where the signal is expressed as a series of coefficients representing different resolutions of the signal, then coefficients for low resolutions (that is, high frequency) are reduced or removed [Coombes et al., 2005].

Alignment aims to correct errors in mass measurement (Figure 6). Errors can occur both systematically for a whole spectrum, and only in part of a spectrum. In SELDI, the variance may be +-0.2% of any m/z [Yutaka et al., 1900].

2.5 VARIABLE SELECTION

Broadly speaking, biomarker discovery from mass spectral data is based on finding a small set of variables, candidate biomarkers, whose measurement can discriminate the classes. The problem of finding discriminative variables is called variable selection and has been extensively studied in statistics and machine learning. The problem is framed as a prediction task, namely to find a small set of variables that can be used to build a model that can accurately classify unseen instances.

There are three general approaches to variable selection: wrapper, filter and embedded. Wrapper algorithms use a search algorithm (such as best-first search) to iteratively select promising candidate sets of variables, and evaluate them by learning and testing a classifier. At the end of the process the best-performing set of variables is selected. Since they evaluate sets of variables together, they can find a set that performs well as a whole, even if no single variable in the set performs very well. On the other hand, they are computationally expensive, and intractable with a large number of variables. Furthermore, they can be prone to over-fitting when the search space is very large. For example exhaustive search needs to consider a space that is exponential in the number of variables, and hence requires too many data points for reliable learning when the number of variables is large.

Filter approaches evaluate the variables or variable sets in a single pass, without training a classifier on sets. This is typically much more computationally efficient, but may yield a poorer set of variables. The simplest cases of filter approaches are univariate measures of correlation with the class variable, such as Pearson correlation, Wilcoxon correlation and Chi-squared (χ^2) correlation [Liu et al., 2002, Liu and Setiono, 1995]. Univariate measures do not consider the correlation of different variables with each other, and can select variables that do not add any information. For example if the data can be explained well by just three variables, but each of them is repeated many times, then univariate measures will select many copies of the most discriminative variable, and would rank the third variable needed for good performance very low on the list. Another form of filter approach, Correlation-based Feature Selection [Hall, 1999], uses pairwise variable correlations to mitigate this problem.

In the embedded approach to variable selection, variables are selected during the process

of learning a model. For example, when learning a linear model, L-1 regularization can be used to directly encourage the model to be sparse i.e. use a small number of variables. Another example is the use of an ensemble to learn a set of models with different input variables. The importance of different variables can be assessed by the frequency of their use in accurate models. Finally, sparsity can be a byproduct of the learning method as in coordinate descent algorithms such as AdaBoost.

2.5.1 Efficient Bayesian Discretization

Discretization is the process of converting real values into intervals of values; the set of intervals are called a discretization policy. This is important for learners such as RL which only work with discrete-valued data, because many techniques for molecular profiling, such as mass spectrometry and immunoassays produce real-valued measurements. Another benefit of discretization is that it act as a variable selection for the input data: variables for which discretization produces only a single interval $(-\infty, \infty)$ are removed from the data set.

One discretization algorithm that has been found to work well with RL is Efficient Bayesian Discretization (EBD) [Lustgarten, 2009]. EBD is a supervised algorithm, meaning that it makes use of the class variable. Like most published discretization algorithms, EBD is univariate: that is, it discretizes one variable at a time.

EBD takes as input a vector of n pairs (X, Z) of values for the continuous input variable X and class variable Z. For example the input might be <(1.2, T), (3.2, F), (2.3, T), (4.3, F)>. EBD sorts the input variable vector, splits it into k intervals representing a discretization policy, and scores the policy using a Bayesian score. EBD considers all policies that have up to K intervals and outputs the one with the highest score. The scoring function used by EBD is:

$$score_{EBD} = P(D|M)P(M), \tag{2.1}$$

where $P(D|M)$ is the posterior probability of the data D given the discretization policy M and $P(M)$ is the prior probability of the discretization policy. This quantity has a closed form solution under the following assumptions:

1. The values of the values of the target variable were sampled independently (iid)from the distribution $P(Z|X)$, which is modeled as a multinomial distribution;
2. Prior belief about the distribution $P(Z|X = x_i)$ is independent of prior belief about the distribution $P(Z|X = x_j)$ for all values x_i and x_j of X, such that $i \neq j$; and
3. For all values x_j of X, prior belief about the distribution $P(Z|X = x)$ is a Dirichlet distribution with hyperparameters α_i and $\alpha_{i,j}$.

EBD avoids recomputing sub-problem solutions. For example, if EBD has found the best policy for instances 1 to 4 is a single interval, then when considering instances 1 to 10, it never considers policies that split instances 1 to 4 into separate intervals. EBD has a computational complexity $O(n^2m)$ where n is the number of instances and m is the number of variables in the data set.

2.5.2 ReliefF

ReliefF is a variable ranking and selection algorithm that takes account of variable interactions, unlike most filter methods, most of which evaluate variables based on the impurity of the class value distribution [Robnik-Šikonja and Kononenko, 2003]. Instead, ReliefF computes a score for each variable, based on not only the difference in variable values, but also the distance between instances. Distance is computed in the feature space. The variable scores are iteratively refined by promoting variables that differentiate instances of different classes, and demoted if they differentiate instances of the same class. Specifically, in each iteration ReliefF randomly chooses a training instance and finds its nearest neighbor from the same class and the nearest neighbor from the opposite class.

2.5.3 Correlation-based feature selection

Correlation-based Feature Selection (CFS) [Hall, 1999] is a filter-based variable selection algorithm that takes account of weak variable interactions. It evaluates subsets of variables based on the variables' individual correlation with the target variable, as well as their mutual correlation. Good variable sets are assumed to contain variables that are highly correlated with the target but not correlated with each other. The space of variable subsets is searched

using a heuristic search such as best-first search (BFS). BFS starts with an empty set of variables and generates all possible single variable subsets. The subset with the highest score is chosen and grown in the same way by adding a single variable. If expanding a set does not improve the score, the search goes back to the next best unexpanded set. BFS eventually explores the whole space of feature subsets, but CFS stops the search after five consecutive fully unexpanded non-improving subsets.

2.6 COMPUTATIONAL VALIDATION OF CANDIDATE BIOMARKERS

Sets of variables can be evaluated by learning a classification model on a randomly chosen subset of the data set, and evaluating the model on the rest of the data. There are various measures of performance of classification models. These include **accuracy** (or **error**, which equals 1 minus accuracy), **sensitivity** (true positive rate), **specificity** (true negative rate), **positive predictive value** (PPV), **negative predictive value** (NPV), **balanced accuracy** and **relative classifier information** (RCI). See Appendix (Glossary) for details.

When data is scarce, it is more likely that the randomly chosen test set has a different distribution than the general population, and so the evaluation of the model can be an over-estimate or an under-estimate. There are a number of approaches to reduce this effect. Two popular approaches are resampling validation and cross-validation. In **resampling validation**, the procedure of choosing a training set and learning and evaluating a model is repeated a number of times.

In **cross-validation**, the whole data set is partitioned into a number n of subsets of equal size, called "folds". Each fold is in turn used for testing the model learned on union of the other $n - 1$ folds. Finally, the classification performance of all n folds are averaged to produce an estimate of the performance of a classifier learned on the whole data set. To reduce the effect of a possible lucky or unlucky partition of folds, cross-validation can be repeated a number of times, each time with a different partition into folds. The commonly used scheme is 10-fold cross validation repeated 10 times. It is known that cross-validation leads to a higher bias of the performance estimate, while resampling validation leads to a

higher variance.

2.7 TRANSFER LEARNING

As explained in Section 1, this thesis uses transfer learning to improve biomarker discovery from mass spectral data. Transfer learning is the use of knowledge about one task to help in learning another task. For instances, learning the task of recognizing the edge of the road can be used to learn to better learn the task of steering a vehicle [Caruana, 1997]. Inductive transfer appears to be fundamental to human learning [Ellis, 1965]. In machine learning, variations of transfer learning have been studied under the names "inductive transfer", "knowledge transfer", "life-long learning", "learning to learn" [Thrun, 1996], "meta-learning" [Vilalta and Drissi, 2002] and "cumulative learning". The assumption in using transfer learning is that the tasks are different but related, so that pooling their instances does not make sense, but treating them as independent fails to use the information from one to benefit the other [Xue et al., 2007]. Intuitively, learning models for the tasks using a shared representation can allow them to provide domain information to one-another.

A survey of transfer learning is given by [Pan and Yang, 2010]. The authors define a learning domain and a learning task as follows. A **learning domain** D consists of two components: a variable space χ and a marginal probability distribution $P(X)$, where $X = x_1, ..., x_n \in \chi$. For example, if the learning task is document classification, and each text term is a binary variable representing whether the term is present in the document, then χ is the space of all possible term vectors, x_i is the ith term vector, and X is a subset of vectors sampled from χ and used for learning. If two domains are different, then either the feature spaces or the probability distributions are different.

Given a learning domain, a **learning task** in the domain consists of two components: a space Y of classification labels and a conditional probability distribution $P(y|x)$ that can be used to predict the label y of an instance x. Thus if two tasks are different, then either their label spaces are different or their conditional probability distributions are different.

In this thesis we consider the case where there is one source domain, D_s, and one target

domain D_t, which is the most popular in the research literature. Let the source domain data be denoted $Z_s = (x_{s_1}, y_{s_1}), ..., (x_{s_{n_s}}, y_{s_{n_s}})$, where $x_{s_i} \in \chi_s$ is a data instance (data vector), and $y_{s_i} \in Y_s$ is its classification label. Similarly, let the target domain data be denoted $Z_t = (x_{t_1}, y_{t_1}), ..., (x_{t_{n_t}}, y_{t_{n_t}})$

Finally, adapting the definition from [Pan and Yang, 2010], we can define **transfer learning** as follows. Given a source domain D_s, a source learning task T_s, a target domain D_t, a target learning task T_t, and a learning performance measure a, transfer learning aims to improve the performance of the learned classifiers on target predictive function $f_t()$ in D_t using knowledge in D_s and T_s, where $D_s \neq D_t$ or $T_s \neq T_t$.

We can consider how these definitions apply to common scenarios of biomarker discovery from data as described in Chapter 1, Introduction. The motivating scenario for attempting transfer learning for biomarker discovery was the availability of multiple data sets of the same population of subjects, same sample type and same measurement technique, but different batches of measurement. This setting is most likely to result in positive transfer because the source doman and target domain are similar in most aspects. In this scenario, the difference between the source domain and target domain is the marginal distribution of samples due to systematic measurement errors in both batches of measurement. This means that the source and target domains are different. The difference in distribution are especially marked with mass spectrometry data.

Learning domains can also differ in their respective sets variables. Some measurement techniques, such as mass spectrometry, invariably measure slightly different variables when measuring the same biological samples in different measurement batches. Thus, the original source and target domains are composed of slightly different sets of variables. However, it is common practice to eliminate such differences using alignment procedures, as descrived above in Section 2.4, Pre-processing. The result is that the source and target data sets have the same sets of variables, that represent approximately the physical quantities that were measured (e.g. abundances of particles of a given mass-to-charge ratio). This thesis does not attempt transfer learning between learning domains having sets of variables. The TRL framework developed in the thesis allows transfer between domains that have some variables in common. Intuitivey, it can be hypothesized that the greater is the proportion of variables

in common, the grater the expected effect of transfer. However, in all of the experiments done to evaluate the framework (Chapter 4) the source and target data sets always had the same set of variables.

If the source and target data sets are generated from different populations of subjects, the source and target domains will have different marginal probability distributions and so will be different. For example, populations can be humans and rats; or human stage 1 lung cancer patients and healthy individuals on the one hand, and human stage 3 lung cancer patients and healthy individuals on the other hand. The marginal probability distributions would also be different if the source and target domains represent different types of biological samples, such as blood on the one hand and urine on the other hand. In this case, the marginal probability distributions of the instance vectors would be different because the variables in the data represent different mixtures of physical particles in the samples. In this thesis, we do not attempt transfer learning across different populations of subjects or sample types. The marginal probability distributions can also be different if the data sets are generated using different measurement technologies. For example, mass spectrometry using different type of "ProteinChip" will create different marginal distributions of mass spectra because the chips are sensitive to different classes of proteins. In this thesis, we have attempted transfer learning across MALDI and SELDI technoligies in one pair of data sets. See Chapter 4, Experiments.

Based on their definitions, [Pan and Yang, 2010] show the relationships between traditional (base) learning and different types of transfer learning, as shown in Table 2. The authors categorize transfer learning into inductive transfer learning, transductive transfer learning, and unsupervised transfer learning based on the source and target domains and tasks.

1. *Inductive transfer learning* is where the source and target tasks are different and related, while the source and target domains may be the same or different. Therefore, some labelled target data are needed to learn a predictive model (classifier) $f_t(x)$ for use on the target domain. Inductive transfer can further be divided into two cases:
   a. Labeled source data are available. This setting is similar to multi-task learning.
   b. No labeled source data are available. This setting is similar to self-taught learning [**?**],

Type of learning		Source, target domain	Source, target task
Base learning		The same	The same
Transfer learning	Inductive transfer & Unsupervised transfer leraning	The same	Different but related
		Different but related	Different but related
	Transductive learning	Different but related	The same

Table 1: Relationship among traditional (base) learning and various types of transfer learning.

in which the set of labels in the source and target domains are very different from each other.

2. *Transductive transfer learning* is where the source and target tasks are the same, while the domains are different but related.

3. *Unsupervised transfer learning* is where the source and target tasks are different and related, similar to inductive transfer (1), but the learning tasks are unsupervised, such as clustering and dimensionality reduction.

Transfer learning can also be classified by *what type information* is being transferred:

1. *Transfer of knowledge of instances* or *instance-based transfer learning* assumes that some parts of the source domain can be re-used. [Pan and Yang, 2010] consider only the case where source instances are re-weighted. Howerver, TRL is an example where *information* about the source domain is implicitly transferred in the prior rules seeding RL's heuristic beam search, without re-weighting.

2. *Transfer of feature representations* aims to find a good representation of features to reduce divergence between the source and target domain. This involves variable construc-

Transfer learning setting	Related areas	Source domain labels	Target domain labels	Tasks
Inductive transfer	Multi-task learning	Available	Available	Regression, Classification
	Self-taight learning	Unavailable	Available	Regression, Classification
Transductive transfer	Domain adaptation, Sample selection bias, Co-variate shift	Available	Unavailable	Regression, Classification
Unsupervised transfer learning	-	Unavailable	Unavailable	Regression, Dimensionality reduction

Table 2: Settings for transfer learning based on availability of source and target labels.

tion, and is similar to common-variavle learning. TRL does not do any variable construction, unless the conjunction of variable-value pairs in rule antecendents are considered new feeatures. However, it is possible to extend TRL to perform supervised variable construction, as discussed in Sectionr̃efsect:conclusion, Conclusion.

3. *Transfer of shared parameters* aims to discover common hyperparameters between the distributions of the source and target tasks. The transferred knowledge are the shared hyperparameters.

4. *Relational knowldge transfer* or *relational transfer learning* deals with transfer learnig for relational domains, where the data is not "independent and identically distributed" (*i.i.d.*) as is traditionally assumed in machine learning. The data can be represented by multple relations, such as social networking data. Relational transfer learning tries to transfer the relations between the data from the source domain to the target domain.

Transfer learning can improve generalization performance of the learned information, the speed of learning and the intelligibility of learned models [Caruana, 1997]. If it improves the desired aspect, it is called **positive transfer**, and if it degrades it, it is called **negative transfer**. This thesis focuses on improving performance on one classification learning task. In this thesis, we consider a **learning task** to be learning a classifier with discriminatory markers from clinical data set collected one set of experimental conditions, e.g. a set of SELDI proteomic mass spectra collected from samples in one session. A related task will be

considered any other data set where we might expect to see the same information learned from the two data sets. Operationally this will be, for example, a data set collected in a different session, but from the same type of samples (e.g. blood serum), same spectrometer platform (e.g. SELDI IMAC), and the same or very similar measurement conditions.

Inductive transfer may not be beneficial to learning if the transfer mechanism is not appropriate for the tasks at hand; it may even degrade learning performance. See for example [Caruana, 1997]. Indeed, finding an appropriate mechanism is a difficult research problem for any source and target domains.

The benefits of transfer learning have been demonstrated in a number of theoretical and experimental studies. Theoretical results include [Baxter, 1995, Baxter, 1998, Ben-David et al., 2002, Ben-David et al., 2003]. Those studies prove that, given a number of data sets from different tasks, it is possible in theory to more effectively learn a classification function for those tasks or one of the tasks than using data from just one task. The proofs are based on the idea that the additional data sets can be used to reduce the hypothesis space of the learning tasks. [Baxter, 2000] casts inductive transfer as the search for a learning bias, in particular, a set of variables, that is appropriate for all learning tasks in an environment; he proves that the sample complexity (the bound for number of instances) per task for learning new instances is smaller than when using one task, and the sample complexity for learning a new task is smaller after learning the bias of the environment. The number of instances required to accurately estimate the error of a hypothesis depends inversely on the number of tasks. [Baxter, 2000] shows that variable sets can be learned using a single-hidden-layer ANN.

[Baxter, 1998] presents a hierarchical Bayes model [Gelman et al., 2004] of learning to learn, where Q is an "objective prior" for sampling the task distributions P_θ. The learner's bias is represented by the set of candidate distributions, $R = \{P_\theta | \theta \in \Theta\}$, and a "subjective prior" P_π parametrized by $\pi \in \Pi$. The learner's goal is to find the task-sampling distribution Q among the set of candidate prior distributions, $\{P_\pi | \pi \in \Pi\}$, given a hyper-prior P_Π on Π, and assuming Q is among the candidates. This model is used to derive bounds for the sample complexity of the ANN learning of [Baxter, 2000].

Most of the existing experimental results in transfer learning involve artificial neural

networks (ANNs). However, ANNs are not well-suited for biomarker discovery because:

1. It is difficult to extract the discriminative variables (potential biomarkers) among all input variables of the learned ANN.
2. Training an ANN requires many training instances
3. Training an ANN is computationally intensive

There are several studies that have explored transfer with other learning methods. [Caruana, 1997] proposed a method for transfer with decision tree learning. However, this method is based on multi-task learning, which does not apply to the problem of transfer among mass spectrometry sessions. [Thrun and O'Sullivan, 1996] use a nearest neighbor approach to selectively transfer from the most similar tasks when there are many source tasks.

The study that is most similar to this thesis is [Reid, 2007], which developed the DEFT system. DEFT is a system for transfer learning of first-order logic rules, transferring evidence among similar rules. Two rules are considered similar if they use the same variables. The evidence transferred is the conditional probability table (CPT), namely the number of true positives, false positives and negatives of the rule on the source data. This prior CPT is then combined with CPT from the target data to arrive at the posterior estimate of the rule's quality. This is equivalent to our "union baseline" as explained in Section 4. Unlike DEFT, which transfers the evidence for the prior rules, TRL transfers the rules and fills in the prior evidence from the target data only. DEFT does nothing to modify the learner's learning bias or search space. By contrast, TRL directly affects the learning bias by seeding the search with the prior rules. This means TRL starts the search from a point in the search space that is assumed to be close to a good solution because the learning tasks (the data sets) are assumed to be similar. One effect of this seeding is that it makes it more likely that the prior rules will be used in the new model, if they are accurate on the target data. This acts as a confirmation of the prior knowledge.

Also, unlike DEFT, TRL uses propositional learning, which makes it more suitable for biomarker discovery from highly multi-dimensional data and scarce training instances.

Much of the existing experimental work in transfer learning deals with scenarios where

the data is partially shared between learning tasks. For example, [Caruana, 1997] and [Silver and Mercer, 2002] use ANNs that share inputs and hidden representations, but have different output; for example in [Caruana, 1997], (see Appendix), every data vector contains values for the input variables that represent an image of a driver's view of a road, and values for the output variables that specify (a) which way the car should be steered and (b) the location of the edge of the road. Similarly, in Abu Mostafa's "learning from hints" [Abu-Mostafa, 1990], the input patterns are the same for all tasks. However, this scenario does not apply to the current thesis because we have tasks whose input variables represent roughly the same inputs that have a real-world meaning (the approximate m/z of various peptides in the clinical sample), but have the same output variable (the clinical group of the individual; e.g., healthy or diseased with lung cancer). In particular, finding discriminative variables from data obtained from each experimental session will be considered one learning task. The goal will be to find a small set of variables that are discriminative for all tasks.

2.8 CLASSIFICATION RULE LEARNING WITH RL

As seen in Section 1, the TRL transfer learning framework is based on the classification rule learner RL [Clearwater and Provost, 1990]. Rule learning has a number of advantages for biomarker discovery, specifically understandability, and RL was chosen because it has been successfully used in biomarker discovery from gene-expression and proteomic mass spectrometry data [Gopalakrishnan et al., 2006, Ranganathan et al., 2005, Ryberg et al., 2010].

This section provides an overview of RL, which will be useful in understanding transfer learning described in Sections 3.1–3.2.

RL [Clearwater and Provost, 1990] is a classification learning algorithm that outputs a rule model classifier. RL's input is a set of training instances, where each instance is a vector of values for the input variables, and a class value. The classifier comprises a set of rules of the form:

```
IF <condition> THEN <consequent>
```

where the condition is a conjunction of one or more variable-value pairs, and consequent is a prediction of the class variable. For example, a rule learned from proteomic mass spectra might be:

```
IF ((MZ_2.05 = 1.30..inf) AND (MZ_9.65 = 0.15..0.23)) THEN Class=Control
```

Parameters: C, constraints on rules
Input : X, a data set of training instances
$\beta_0 \leftarrow$ `MakeBeam`$(\emptyset \Rightarrow \texttt{class}_1, \emptyset \Rightarrow \texttt{class}_2 \ldots)$;
model $\leftarrow$ [] ;
for *iteration* $i = 0, 1, \ldots$ **do**
  if β_i *is empty* **then return** model $\beta_{i+1} \leftarrow$ `MakeBeam`() ;
  for *each rule* $\rho \in \beta_i$ **do**
    if `satisfies`(ρ, C, X) **then**
      model $\leftarrow$ model + $[\rho]$;
    end
    if not `isincorrigible`(ρ, C, X) **then**
      $\beta_{i+1} \leftarrow \beta_i +$ `specialize`(ρ);
    end
  end
end
return model;

Algorithm 1: RL. Function `MakeBeam` creates a beam sorted by certainty factor. The initial rules have no conditions, so they apply to all data. Function `satisfies` checks if the rule satisfies the constraints. Function `incorrigible` checks if a specialization of the rule might satisfy the constraints.

A classification rule is an assertion that a data instance matching the variable-value pairs in the condition has the class specified in the consequent. The rule in the example above can interpreted as "the label should be Control if the variable for m/z 2.05 kDa was measured at above 1.30 units and the m/z 9.65 was measured between 0.15 and 0.23." RL uses these intervals of real values as discrete values which the input variables can take on. If some input variable has values in the data that are real numbers, such as MZ_2.05 = 1.30, then the variable values must be converted to intervals before RL can start learning. This process of conversion is called **discretization**. If the variable is categorical, the value is a discrete category, such as the value "female'. The class variable is also discrete.

RL rules have an associated score, called a **certainty factor**. The term comes from rule-based expert systems, where rules were created to represent expert knowledge and the score corresponds to the expert's certainty in each rule [Shortliffe Bruce and Edward, 1975, Buchanan and Feigenbaum, 1978]. By contrast, in rule induction algorithms such as Rule Learner [Clearwater and Provost, 1990], the certainty factors are calculated from the training

instances. Various functions can be used as a certainty factor, such as positive predictive value (PPV), signal-to-noise ratio (S/N) and likelihood ratio (LR). The certainty factor functions used by Rule Learner are defined in Appendix .

A rule **matches** an data instance if the instance's values logically satisfy the rule condition. The coverage of a rule is the number of instances that match the condition. A **true positive** (TP) is an instance that matches both the condition and the consequent; a **false positive** (FP) is an instance that matches the antecedent but not the consequent; a **true negative** (TN) is one that matches neither the condition not the consequent; a **false negative** (FN) is one that does not match the condition but matches the consequent.

A classification rule model is a set of rules with an associated way of applying them, called an **evidence gathering method**, for breaking ties when several rules match but predict different class values. The experiments in this thesis used the default evidence gathering method, "weighted voting": each rule that fires votes for the class it predicts and the votes are weighted by the rules' certainty factors. Another evidence gathering method is to always apply the rule with the highest certainty factor among matching rules.

The RL algorithm pseudo code is shown in Algorithm 1. It defines constraints on acceptable rules in terms of a number of quantities defined with respect to a rule and a data set. The constraints are minimum coverage, min. certainty factor, maximum false positive rate, and inductive strengthening. **Coverage** is the fraction of training instances for which the rule condition is satisfied. **Certainty factor** (CF) is a measure of the rule's accuracy; we used the true positive rate: the ratio of number of instances the rule predicts correctly divided by the number of instances it matches. **False positive rate** is the ratio of number of instances the rule predicts incorrectly divided by the number of instances it matches. **Inductive strengthening** is the minimum number of previously uncovered instances that a proposed rule must cover.

The algorithm proceeds as a heuristic beam search through the space of rules from general to specific [Provost et al., 1999]. Starting with all rules containing no variable-value pairs, it iteratively specializes the rules by adding conjuncts to the condition. It evaluates the rules and inserts promising rules onto the beam, sorted by decreasing certainty factor. Beam search is used to limit the running time and space of the algorithm.

If specializing rule R_1 produces rule R_2, then R_2 is called the **child** of R_1 and R_1 is called the **parent** of R_2. A child rule matches a subset of the instances that its parent matches, and that subset may have more homogeneous class values. Thus the child may have a higher certainty factor value. On the other hand, when the rule matches too few instances, it is unlikely to generalize well, and should not be included in the classification model. Thus when a rule's coverage drops below the coverage threshold, it is not specialized any more. To reduce the chance of over fitting, the instances are not sampled with complete replacement. Instead new rules must cover at least σ previously uncovered training instances; σ is a user-specified parameter called "inductive strengthening".

After the search terminates, the set of found rules that specify user-specified constraints, together with a predefined evidence gathering method, is the induced classifier (the model) and can be used to classify unobserved instances. The evidence gathering method defines which rule should apply in case the instance matches two or more rules with different consequents. A simple method is to apply the rule with the highest certainty factor value. Another, commonly used, method is weighted voting, where each matching rule is weighted by its CF.

RL has several advantages over other rule learners and decision tree learners. First, its simplicity and flexibility allow the user to use domain knowledge to explicitly set an appropriate learning bias. Second, RL's rules can represent nonlinear relationships such as XOR. This is not possible in decision tree-based learners such as RIPPER [Cohen, 1995], where each attribute is considered only once for classification. Third, RL covers data with replacement, unlike most rule and decision tree learners, such as C4.5 [Quinlan, 1993], CART and RIPPER, which cover data without replacement. Covering with replacement is an advantage if data are scarce, because it leaves more instances to provide statistical support for newly discovered rules. We have observed this in some experiments where RL was more accurate than C4.5 Fourth, RL can handle hierarchical attributes and hierarchical values for attributes. Fifth, RL's model may abstain from making a prediction if it is not confident in the prediction. This is an implicit way of avoiding costly errors. RL also has an explicit way to handle classification errors, allowing the user to specify their relative costs.

Any learner needs a **bias**: a set of assumptions that allow it to generalize from

observed data instances in order to be able to make predictions on unobserved instances [Mitchell, 1980]. The bias includes the choice of hypothesis space, which must be large enough to contain the target concept, but small enough to ensure good generalization. RL has 12 parameters that modify its bias and can be set by the user to reflect her prior knowledge about the learning problem. (See Sections and .) The main four parameters are:

1. Certainty factor function: a function (described above)
2. Minimum certainty factor value: the minimum CF which any rule must have in order to stay on the beam
3. Beam width: the number of rules kept on the beam after each iteration of evaluating all rules.
4. Maximum number of conjuncts: the maximum number of variable-value pairs in any rule's condition.

RL can perform a search over the space of parameter values for an appropriate learning bias. It iterates with several values for each parameter; for each set of values for all the parameters, it learns a rule-based classifier on part of the training data, then evaluates the model by making predictions on the remainder of the training data and scoring the accuracy of those predictions. Finaly, it chooses the parameters that gave the most accurate model. This iterative evaluation is called **bias space search** or **learning parameter optimization**. The learning is done over ver cross validaion of the training data.

3.0 TRANSFER RULE LEARNING FRAMEWORK

The transfer scheme we propose is to load the prior rules into the beam along with the initial rules, before the first evaluate-specialize iteration. Figure 7 illustrates the algorithm.

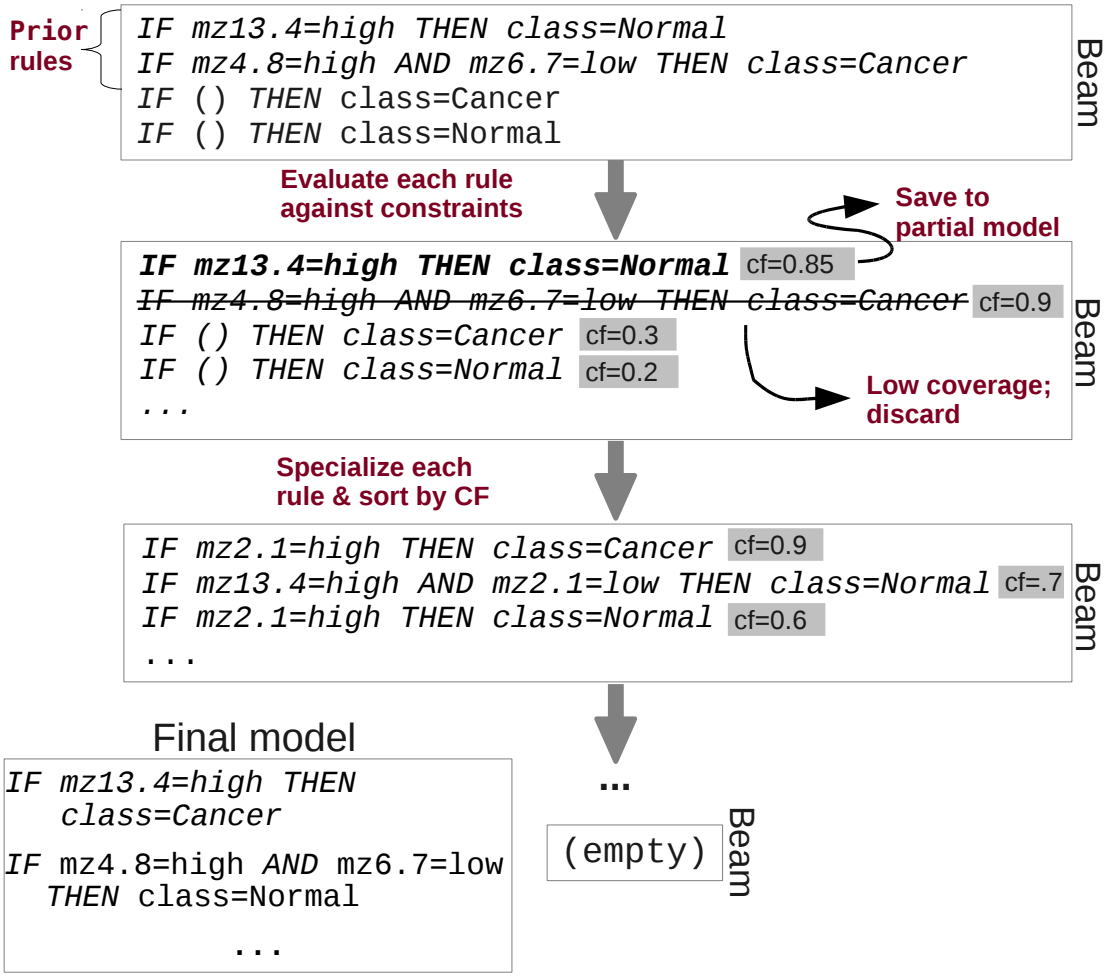


Figure 7: Illustration of the TRL algorithm.

Source data set

Attributes	mz4.8	mz13.4
Values	High	High
	Low	Low

Target data set

Attributes	mz4.8	mz13.4
Values	High	(none)
	Middle	
	Low	

Figure 8: Illustration of a mismatch between the discretization of the source data set and the discretization of the target data set.

RL's input data must be discrete, that is, each variable has a small number of values in the data. Many molecular profiling methods create real-valued data, so these data must be discretized for use with RL; that is, each real-valued measurement must be assigned to a range. A challenge in rule transfer is to make sure that the ranges have the same meaning across the two data sets. To transfer prior rules, we can discretize one data set and apply the same discretization to the other one (simple rule transfer), or transform the prior rules to use the new discretization (rule structure transfer). A further consideration is how prior rules will affect RL's search process. This is discussed in Section 3.3. Algorithm 2 shows the transfer learning algorithm. The difference from Algorithm 1 is that we initialize the beam with a list or prior rules π as described in the next sections. The set of constraints C is also sometimes modified, as described in the sequel.

3.1 SIMPLE RULE TRANSFER

The simplest way to avoid possible differences in the discretization between source and target is to ensure that they are discretized identically. Specifically, we discretized the target data and impose the same discretization on the source before running RL to compute prior rules.

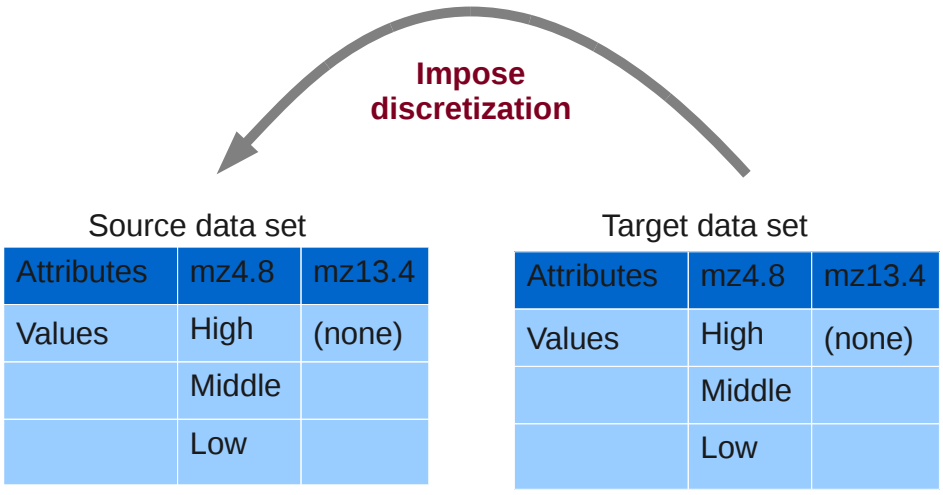


Figure 9: Simple rule transfer: imposing the target data discretization onto the undiscretized source data set.

This guarantees that the discretization makes sense on the target data, even if it means that the prior rules will be less accurate on the target data.

3.2 RULE STRUCTURE TRANSFER

Because of systematic differences between source and target data, it might not be optimal to use the same discretization for them. For example, two sets of proteomic mass spectra might have different baselines caused by the state of the measurement equipment. Even with post-processing such as baseline subtraction, numerical values in one data set might not correspond to numerical values in the other. To overcome such differences, we explore transferring only the structure of rules. Specifically, we remove feature values from the prior rules and re-instantiate them for the target task.

For example, a prior rule:

```
IF (MZ_7.23 = High) THEN (Group = Cancer)
```

is converted to a prior rule structure:

```
IF (MZ_7.23 = ?) THEN (Group = ?)
```

The rule is then instantiated from the target data discretization as:

```
IF (MZ_7.23 = High) THEN (Group = Cancer)
IF (MZ_7.23 = Low) THEN (Group = Cancer)
IF (MZ_7.23 = High) THEN (Group = Healthy)
IF (MZ_7.23 = Low) THEN (Group = Healthy)
```

We consider all classes in addition to all discrete feature values because the relationship in the data might be stronger in the dual rule:

```
IF (MZ_7.23 = Low) THEN (Group = Healthy)
```

and we want to count that as a retained prior rule because it represents the same relationship between variable values.

This approach has the additional advantage that the source data set is not needed during the transfer learning. The prior rules can be used when the source data set is not available, for example by extracting them from literature.

3.3 EFFECT ON THE SEARCH

To avoid over fitting, RL checks that any new rules added to the model cover at least σ previously uncovered training instances, where σ is the inductive strengthening. When prior rules are added to the beam, they may cover some training instances from the new training data (the target data) that are otherwise covered by other rules. Thus some rules that would otherwise be included in the final model might be excluded if prior rules are used. This may reduce the classification performance of the final model on the new data. To reduce this effect, we created a variation of the algorithm that we called NC (for "non-covering"). In this variation, the training instances covered by prior rules are ignored for purposes of inductive strengthening. However, prior rules still affect the search through the beam: they and their specializations may displace rules from the beam that would have otherwise stayed on the beam. Also, prior rules can still produce different predictions, because the classifier includes prior rules in addition to the newly learned rules.

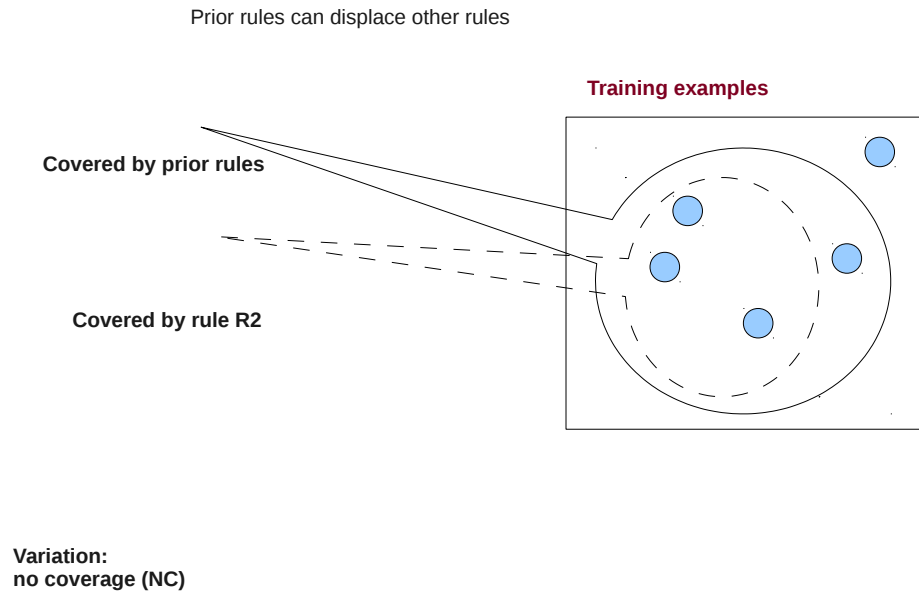


Figure 10: Illustration of displacement of a rule by prior rules. The examples covered by rule R2 have already been covered by prior rules, so R2 is not admissible. The nc transfer algorithms are intended to reduce that effect.

3.4 SCALING OF THE DATA SETS

When taking a batch of measurements like those typically found in a biomarker discovery data set, there is often systematic measurement artifacts that affect all the data points measured, but are specific to that batch of measurements. A classic example of this is a set of mass spectra generated in one batch (one "session"). As explained in Section 2.2, the intensities represent only the relative abundances, relative to the MS session. These systematic measurement artefact's can interfere with transfer learning, and in particular with the discretization. For example, suppose in the source data set the intensities measured for some mass to charge ratios are offset by background radiation, but in the target data set there is no such offset. When we perform discretization using the target training data, the source data might all end up in the "low" bin for the offset mass to charge ratios. Even if these are discriminative features for the source data set, we would not be able to learn any

rules that are applicable to the target data. However, if we have a sufficiently large sample, we can normalize the source and target data to have the same mean and variance. If they are drawn from the same distribution of biological samples, and the data set sizes are large enough, this should counteract the systematic measurement errors.

Parameters: C, constraints on rules

Input : X, a data set of training instances

* **Input** : π, *a list of prior rules*

* $\beta_0 \leftarrow$ `import`(π) + `MakeBeam`$(\emptyset \Rightarrow$ `class`$_1, \emptyset \Rightarrow$ `class`$_2 \ldots)$;

model $\leftarrow$ [] ;

for *iteration* $i = 0, 1, \ldots$ **do**

  if β_i *is empty* **then**

    return model

  end

  $\beta_{i+1} \leftarrow$ `MakeBeam`() ;

  for *each rule* $\rho \in \beta_i$ **do**

    if `satisfies`(ρ, C, X) **then**

      model $\leftarrow$ model + $[\rho]$;

    end

    if not `isincorrigible`(ρ, C, X) **then**

      $\beta_{i+1} \leftarrow \beta_i +$ `specialize`(ρ);

    end

  end

end

return model;

Algorithm 2: Rule Learning with Rule Transfer. The lines that differ from base RL (Algorithm 1) are prepended with an asterisk (*). Function `MakeBeam` creates a beam sorted by certainty factor, but now the prior rules in π are considered first. Function `satisfies` checks if the rule satisfies the constraints. Function `is_incorrigible` checks if a specialization of the rule might satisfy the constraints. Function `import` takes each prior rule and removes any variable that cannot be mapped to a variable in the target data (i.e., it generalizes the rule).

4.0 EXPERIMENTS AND EVALUATION

This section describes the experiments done to evaluate the transfer methods implemented in the TRL framework. Section 4.1 describes the data sets used. Section 4.2 describes the evaluation measures

4.1 DATA SETS

In order to evaluate transfer learning performance, we need to consider classification performance for pairs of compatible data sets. We obtained 30 data sets as summarized in Table 3: 13 MALDI sets, 11 SELDI, 5 luminex and 1 gene expression. Within each pair, each data set was produced from the same overall clinical population and the same type of clinical samples, and using the same protocols and measurement platform (e.g. "ProteinChip" type). The only exception is the experiment of transferring between the ALS MALDI data set and the ALS 2004 data set. Within each pair, data set was used once as the target when the other one was the source. Most of the spectra were generated with replicates, that is multiple spectra for the same biological sample. In the experiments for evaluating TRL, all replicate spectra were averaged before learning.

The gene expression data from [Golub et al., 1999] is a well- known data set with a good classification performance, and we used it to evaluate behavior of the transfer framework on a pair of data sets that are known to be be drawn from the same distribution. Because of its good performance, it allowed us to evaluate the behavior of the methods with different sizes of source and target data sets (and thus a range of performances of the learned classifiers).

The three ALS IMAC SELDI data sets were collected by the lab of Dr. Robert Bowser

T	Disease	Name	Size	#R	#Vars	Vrange	Reference
G	leukemia	train	27+11	-	7,131	-	[Golub et al., 1999]
G	leukemia	test	20+14	-	7,131	-	[Golub et al., 1999]
P	ALS	MALDI 22	23+29	-	73,539	1-70k	Unpublished
P	ALS	IMAC Zn 2004	13+9	-	73,539	2-100k	[Ryberg et al., 2010]
P	ALS	IMAC Zn 2005	44+22	-		0-200k	[Ryberg et al., 2010]
P	ALS	IMAC Zn 2007	53+33	-		0-200k	[Ryberg et al., 2010]
P	lung cancer	WCX UPCI	95+90	2	51,745	2-100k	[Pelikan et al., 2007]
P	lung cancer	WCX Vand	114+88	2	51,745	2-100k	[Pelikan et al., 2007]
P	lung cancer	IMAC UPCI	95+90	2	51,749	2-100k	[Pelikan et al., 2007]
P	lung cancer	IMAC Vand	114+89	2	51,749	2-100k	[Pelikan et al., 2007]
P	lung cancer	luminex train 13	56+56	-	13+1	*	[Bigbee et al., 2011]
P	lung cancer	luminex test 13	10+82	-	13+1	*	[Bigbee et al., 2011]
P	lung cancer	luminex blind 13	30+30	-	13+1	*	[Bigbee et al., 2011]
P	lung cancer	luminex train 70	56+56	-	70	**	[Bigbee et al., 2011]
P	lung cancer	luminex test 70	10+82	-	80	**	[Bigbee et al., 2011]
P	lung cancer	MALDI hic8L 1a	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 1b	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 2a	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 2b	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 3a	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 3b	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 4a	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 4b	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 5a	14+8	2	87,952	0-17k	Unpublished
P	lung cancer	MALDI hic8L 5b	14+8	2	87,952	0-17k	Unpublished
P	breast cancer	HICL 2008	14+8	3	87,952	0-17k	[Kolli et al., 2009]
P	breast cancer	HICL 2009	14+8	3	87,952	0-17k	[Kolli et al., 2009]
P	breast cancer	WCXL 2008	14+8	3	87,952	0-17k	[Kolli et al., 2009]
P	breast cancer	WCXL 2009	14+8	3	87,952	0-17k	[Kolli et al., 2009]

Table 3: Data sets used in the experiments. Horizontal lines delimit groups of data sets containing pairs used for transfer. Column "T" shows whether the data set is proteomic of genomic. "Size" is the number of positive clinical samples (cases) and negative clinical samples (controls); for the [Golub et al., 1999] data set, these are ALL and AML respectively. "#R" is the number of technical replicate spectra measured for each sample in the data set. "#Vars" is the number of variables in the data set, such as number of m/z's measured for SELDI and MALDI data sets, number of proteins selected for luminex data, and number of genes expressed for gene expression data. "Vrange" is the range of m/z's measured for SELDI and MALDI data sets.

at the Department of Pathology, University of Pittsburgh. The samples were collected over a number of years, and were used to create the sets of spectra in 2004, 2005 and 2007 respectively, with small overlaps of the samples among the data sets. The ALS MALDI data set was generated by Dr. Jonathan Lustgarten at the University of Pittsburgh from a small subset of the ALS SELDI 2004 samples, using a Ciphergen ProteinChip spectrometer with a gold place ProteinChip. The exact protocol is not available.

For the lung cancer SELDI (IMAC and WCX) data, separate samples were accrued at the University of Pittsburgh Cancer Institute (UPCI) and Vanderbilt University, as described in [Pelikan et al., 2007] and [Yildiz et al., 2007] respectively. SELDI analysis was done concurrently and with the same conditions as described in [Pelikan et al., 2007], using two types of ProteinChip: WCX and IMAC.

The "lung cancer luminex" data were generated from blood sera of patients with with adeno lung carcinoma, patients with squamous carcinoma, clinical controls and PLuSS controls. The variables are the concentrations of 70 cancer-related proteins measured by Luminex xMAP multiplexed immunoassays. The training set was from samples from 56 patients with biopsy-proven primary adenocarcinoma or squamous cell carcinoma of the lung diagnosed in a clinical setting, and sera from 56 age-, sex- and smoking-matched CT- screened controls who were known to be cancer-free after a minimum 3 year follow-up. The "lung cancer luminex 70" data sets (marked with a ** in Table 3) have 70 markers: G-CSF, HGF, TNF-RI, IL-6, EOTAXIN, MCP-1, TNF-a_T, IP-10, IL-2R, IL-8, Cytokera, ErbB2, Fas_sFa, EGFR, CA72-4, AFP, Kallikre, Mesothel, IGFBP-1, sE-Selec, sV-CAM, sI-CAM, MPO, tPAI1, MIF, FSH, LH, TSH, PROLACTI, GH, ACTH, HE4_XXX, MMP-1, MMP-7, MMP-8, MMP-12, Leptin, NGF, sFasL, Thrombos, Angiosta, CD40L(T, ULBP-1, ULBP-2, MICA, SCC, SAA, TTR, Resistin, MMP-9, Adiponec, TNF-RII, EGF, IL-6R, DR5, IL-1Ra, RANTES, GROa, MCP-3, HSP70, CEA(Fuj, VEGF, LIF, bFGF, PDGF-BB, SCF, TRAIL, M-CSF, SCGF-B, SDF-1a. The "lung cancer luminex 13" data sets (marked with a * in Table 3) include 13 of the previous 70 markers: Cytokeratin19, ErbB2, CEA, sE-Selectin, tPAI1, MIF, PROLACTIN, GH, Thrombospondin, SAA, TTR, RANTES, PDGF-BB. In addition, experiments included the demographic variable, age. Those 13 markers are ones that were found to be discriminative, on the training data set with the 70. Therefore, train-

ing a classifier and testing it on the examples from the original train data set will produce an over-estimate of classification performance. However, this is not important for our purposes of comparing the amount of transfer and classification performance with and without transfer.

The lung cancer MALDI data were collected for the Lung SPORE project, by Dr. David Malehorn at the University of Pittsburgh Cancer Institute (UPCI) Clinical Proteomics Facility. They were created from 15 pools of samples from lung cancer patients, 8 from clinical controls and 8 from controls from the Pittsburgh Lung Screening Study (PLuSS). PLuSS is a community-based study of lung cancer screening with low-dose multi-detector helical computed tomography (CT), funded by the NCI SPORE in Lung Cancer at the University of Pittsburgh and approved by the IRB. Only the cancer and PLuSS control spectra were used for our experiments. The protocol was as follows. Samples were normalized to 5.7 mg for processing (except for 3 pools). Each pooled sample was immunodepleted. Then they were fractionated using anion exchange into 11 fractions: Frx0, Frx1A/1B, Frx2A/2B, Frx3A/3B, Frx4A/4B, Frx5A/5B. Each of the 11 fractions for each sample was purified on HIC8 magnetic beads. Each magnetic prep was then spotted with CHCA matrix. Each spotting was done in duplicate. Spotted samples were read on a Bruker MALDI spectrometer in linear mode, with different mass ranges.

Those 30 data sets were grouped into 17 pairs (13 groups) of compatible data sets for transfer among the sets within each pair. Within each pair, each data set was produced from the same overall clinical population and the same type of clinical samples, and using the same protocols and measurement platform (e.g. "ProteinChip" type). The protocols are described in the published papers.

The 17 pairs of data sets included 14 pairs of mass spectrometry data sets that were pre-processed using baseline subtraction, total-ion-current (TIC) normalization, alignment and, in some cases, other procedures. The procedures are described in Table 4, Legend of Operations, below, and are represented in the names of the resulting data sets in Table 6, Section 5.1. The preprocessing without peak selection resulted in a total of 8 data set pairs (Table 5), an peak selection resulted in a further 17 pairs (Table 6).

After alignment, the two data sets in each pair had the same sets of variables. Thus the

experiments evaluated transfer learning for the case when the source and the target domain share all their variables.

For discretization, we used EBD with λ=1, which has been shown to perform well on biomedical data [Lustgarten, 2009]. RL parameter settings were: min. CF: 85%; min. coverage: 4 instances; max. FP: 10%; inductive strengthening: 1 example.

4.2 EVALUATION MEASURES

The purpose of the transfer rule learning framework is to (1) evaluate the agreement of new data with the prior information, and simultaneously (2) learn new information that incorporates as much of the prior knowledge as useful. High agreement would mean much prior information is retained and is accurate on the new data set. To evaluate the agreement, we measured several variables:

1. The performance of the learned classifier, namely:
   a. accuracy
   b. sensitivity
   c. specificity
2. The amount of information transferred, defined by three measures of the amount of transfer:
   a. rr/pr: number of rules retained as a proportion of prior rules
   b. rr/lr: number of rules retained as a proportion of the total number of rules in the new rule model
   c. ra/la: number of variables (attributes) in the retained rules, as a proportion of number the variables in the new rule model.

The number of rules and variables have to to normalized in this way in order to make it possible to compare them across data sets of different sizes. We recorded those measures for cross-validation folds, and for the final model learned on all the target data.

All experiments are performed in $m \times n$-fold cross-validation, usually 10×10-fold cross-validation. (See Chapter 2.6.)

As a baseline condition, we used learning on the target data alone.

Vand+UPCI	Create a data set as the union of the instances (clinical samples or spectra) in data set "Vand" and the data set "UPCI", first making sure that they have the same list of m/z's
1.5k+, 1.5k-	Select m/z's greater than 1kDa, up till the highest m/z' in the raw data set
2k57k, 2k-67k	m/z's between 1kDa and 67kDa
WCX	Ciphergen SELDI Weak cation exchange
WCXL	Bruker MALDI weak cation exchange, linear mode
IMAC+WCX	Create the union of the m/z's in the data set from the IMAC platform and the WCX platform. The data sets must have the same set of instances (clinical samples)
bs10	Subtract baseline signal with window of 10 m/z's (SpecAlign default)
rn	Remove negative intensities. This is useful after because baseline subtraction.
ri	Relative intensities: scale the intensities for each m/z to the range [0, 1]
tc	Normalize by total ion current
pi	Select peaks using the default Spec Align parameters, and create variables from the peak intensities
pi0.2,10,1.1	Select peaks with parameters 0.2, 10, 1.1, then create variables from the peak intensities
sc	Scale each variable to mean 0, standard deviation 1
cv10	Perform 10-fold cross validation
bss10	Perform bias space search with 10-fold internal cross validation
bss5cv10	Perform bias space search with 5-fold internal cross-validation, repeated over 10-fold external cross-validation
d71	Discretize variables using EBD [Lustgarten, 2009] (RL default)
d03, gaus	Discretize variables using Gaussian discretization with 3 intervals
d62, fi	Discretize variables using Fayyad and Irani Minimum Description Length (MDL) discretization [Fayyad and Irani, 1993]

Table 4: Legend of operations performed on the data sets.

5.0 RESULTS

This chapter presents and discusses the results of the experiments performed to evaluate the proposed methods of transfer. Section 5.1 presents the baseline performance results which are to be compared with the proposed methods, and the rationale for including some of the available data sets and excluding others from the comparison. Section 5.2.1 describes the results of transfer between two subsets of the same data set that was collected in one batch.

5.1 BASELINE PERFORMANCES

This section discusses the classification performance of RL, without transfer, on the available data sets. These baseline performances are then used to calculate a relative improvement in performance when using transfer; these changes in performance are presented in later sections. However, the baseline classification performances are presented as absolute performances.

Unfortunately many of the performances are very low, and especially many data sets result in very low specificity. This is partly because RL made many abstentions when asked to predict the negative data instances (the controls), and our metric considers abstentions as an incorrect prediction. For this reason, we additionally consider the results if RL had predicted control instead of abstaining. That is, we treated abstentions as predictions for control. This altered inference method can be considered a different evaluation measure, and we present it alongside the usual performance measures (accuracy, sensitivity, specificity).

We do not examine transfer across data sets for which RL performs very poorly because the comparison would be meaningless, and such classifiers would not be very useful. Indeed,

Data set	Acc	SN	SP	Ab	Acc	SN	SP	CVF
Golub train	**0.92**	0.82	0.96	0	0.92	0.96	0.82	20
Golub test	**0.85**	0.90	0.79	0	0.85	0.90	0.79	20
ALS MALDI22	**0.55**	0.77	0.22	5	0.73	0.77	0.67	10
ALS IMAC Zn 2004	**0.56**	0.39	0.69	3	0.58	0.39	0.72	10
Lung WCX UPCI bss10	**0.59**	0.63	0.54	5	0.61	0.63	0.59	bss10
Lung WCX Vand bss10	**0.77**	0.70	0.82	7	0.80	0.82	0.76	bss10
Lung WCX UPCI 2k-67k	0.53	0.64	0.41	8	0.56	0.64	0.48	10
Lung WCX Vand 2k-67k	0.73	0.81	0.64	15	0.78	0.81	0.75	10
Lung IMAC UPCI bss10	**0.59**	0.60	0.59	13	0.64	0.60	0.68	bss10
Lung IMAC Vand bss10	**0.71**	0.84	0.54	6	0.73	0.84	0.60	bss10
Lung Luminex train 13	**0.85**	0.86	0.84	0	0.85	0.86	0.84	20
Lung Luminex test 13	**0.86**	0.02	0.94	0	0.87	0.20	0.95	20
Lung Luminex train+test 13	**0.75**	0.61	0.81	18	0.92	0.92	0.92	20
Lung Luminex blind 13	**0.77**	0.77	0.77	0	1	1	1	20
Lung Luminex train 70	**0.77**	0.68	0.86	0	0.77	0.68	0.86	20
Lung Luminex test 70	**0.87**	0.1	0.96	4	0.87	0.1	0.96	20

Table 5: Performance on baseline experiments *without peak selection* (rule learning without transfer). The data sets are named for the initial data set in Table 3, see the legend of data processing operations. "*" indicates the results is for the data set before it is imported with another data set for equalizing the sets of features (m/z's). The data sets achieving a greater than 55% accuracy are in bold.

as we anticipated, and as we will see later, transfer to or from a data set with a very low performance can rarely improve performance, and in that case it might be better to use the classifier learned from the better-performing data set. Instead, to make a meaningful comparison, we use only data sets that have a baseline accuracy of 0.55 or greater. The baseline results are shown in Tables 5 and 6. Thirteen data set pairs (26 data sets) meet the accuracy ≥ 0.55 threshold. These include: the gene expression data (1 data set pair); the ALS SELDI 2004 data and MALDI data (1 data set pair); the lung cancer SELDI WCX data sets with two types of pre-processing (peak-selected and non-peak selected; 2 data set pairs); the lung cancer SELDI IMAC data sets with three types of pre-processing (3 data set pairs); the lung cancer Luminex data sets with 13, 14 and 70 variables (4 data set pairs); and two of the lung cancer MALDI data set pairs, namely 1 and 3 (2 data set pairs).

Data set	Acc	SN	SP	Ab	Acc	SN	SP	CV
ALS MALDI 22 pi *	0.73	0.77	0.67	1	0.73	0.77	0.67	10
ALS IMAC Zn 2004 rn pi *	0.75	0.83	0.69	3	0.73	0.61	0.83	10
ALS MALDI22 pi	0.55	0.77	0.22	2	0.73	0.77	0.67	10
ALS IMAC Zn 2004 pi	0.31	0.34	0.26	16	0.54	0.26	0.76	10
ALS MALDI22 rafft pi	0.45	0.62	0.22	3	0.59	0.62	0.56	10
ALS IMAC Zn 2004 rafft pi	0.29	0.22	0.34	18	0.54	0.22	0.79	10
ALS 2004_rn_ri_rafft_pi_cv10	0.27	0.39	0.17	18	0.52	0.39	0.62	10
ALS 2005_2k-19k_bs_rn_ri_rafft_pi_cv10	0.43	0.59	0.00	9	0.59	0.57	0.64	10
ALS IMAC Zn 2004 pi	0.77	0.90	0.61	3	0.79	0.61	0.93	10
ALS IMAC Zn 2007 pi	0.15	0.00	0.23	64	0.42	0.23	0.73	10
ALS 2004 IMAC Zn ac_pi_cv20	0.71	0.83	0.56	2	0.73	0.57	0.86	20
ALS 2007 IMAC Zn ac_pi_cv20	0.11	0.03	0.16	69	0.39	0.16	0.79	20
ALS 2004 IMAC Zn pi_cv10	0.69	0.93	0.39	2	0.69	0.39	0.93	10
ALS 2007 IMAC Zn pi_cv10	0.12	0.06	0.16	72	0.43	0.16	0.88	10
Lung WCX_Vand+UPCI_2k_pi_sc	0.55	0.77	0.28	75	0.68	0.77	0.58	10
Lung WCX_Vand+UPCI_2k_pi	0.63	0.79	0.44	56	0.73	0.79	0.65	10
Lung IMAC UPCI pi bss10	**0.75**	0.89	0.59	2	0.76	0.89	0.61	bss10
Lung IMAC Vand pi bss10	**0.68**	0.95	0.35	15	0.74	0.95	0.48	bss10
Lung IMAC UPCI pi	**0.63**	0.85	0.40	23	0.74	0.85	0.62	10
Lung IMAC Vand pi	**0.61**	0.94	0.19	19	0.69	0.94	0.38	10
Lung IMAC_Vand+UPCI_2k_pi	0.53	0.84	0.17	82	0.68	0.84	0.50	10
Lung IMAC_Vand+UPCI_2k_pi_sc	0.45	0.79	0.05	108	0.65	0.79	0.49	10
Lung MALDI hic8L_1a_900+_tc_rafft_pi	**0.68**	0.71	0.62	1	0.73	0.71	0.75	22
Lung MALDI hic8L_1b_900+_tc_rafft_pi	**0.73**	0.86	0.50	1	0.77	0.86	0.63	22
Lung MALDI hic8L_2a_900+_bs10_rn_rafft_pi	0.50	0.64	0.25	1	0.55	0.64	0.38	22
Lung MALDI hic8L_2b_900+_bs10_rn_rafft_pi	0.55	0.79	0.12	3	0.64	0.79	0.38	22
Lung MALDI hic8L_3a_900+_bs10_rn_rafft_pi	**0.59**	0.86	0.12	3	0.68	0.86	0.38	22
Lung MALDI hic8L_3b_900+_bs10_rn_rafft_pi	**0.73**	0.79	0.62	0	0.73	0.79	0.63	22
Lung MALDI hic8L_4a_900+_bs10_rn_rafft_pi_capl	0.00	0.00	0.00	19	0.36	0	1	22
Lung MALDI hic8L_4b_900+_bs10_rn_rafft_pi_capl	0.59	0.57	0.62	4	0.59	0.57	0.63	22
Lung MALDI hic8L_5a_900+_bs20_rn_rafft_pi_capl	0.36	0.57	0.00	0	0.36	0.57	0	22
Lung MALDI hic8L_5b_900+_bs20_rn_rafft_pi_capl	0.68	0.79	0.50	0	0.68	0.79	0.5	22
Breast HICL_2008_1600+_bs10_ri_rafft_pi_d03_loo	0.42	0.22	0.62	9	0.53	0.44	0.63	64
Breast HICL_2009_1600+_bs10_ri_rafft_pi_d03_loo	0.35	0.35	0.35	20	0.44	0.48	0.40	64
Breast WCXL_2008_1700+_bs10_tc_rafft_pi_d03_ppv	0.47	0.59	0.34	7	0.53	0.72	0.34	30
Breast WCXL_2009_1700+_bs10_tc_rafft_pi_d03_ppv	0.59	0.73	0.45	5	0.64	0.55	0.73	30

Table 6: Performance on baseline experiments *with peak selection* (rule learning without transfer). The data sets are named for the initial data set in Table 3, see the legend of data processing operations. "*" indicates the results is for the data set before it is imported with another data set for equalizing the sets of features (m/z's). The data sets achieving a greater than 55% accuracy are in bold.

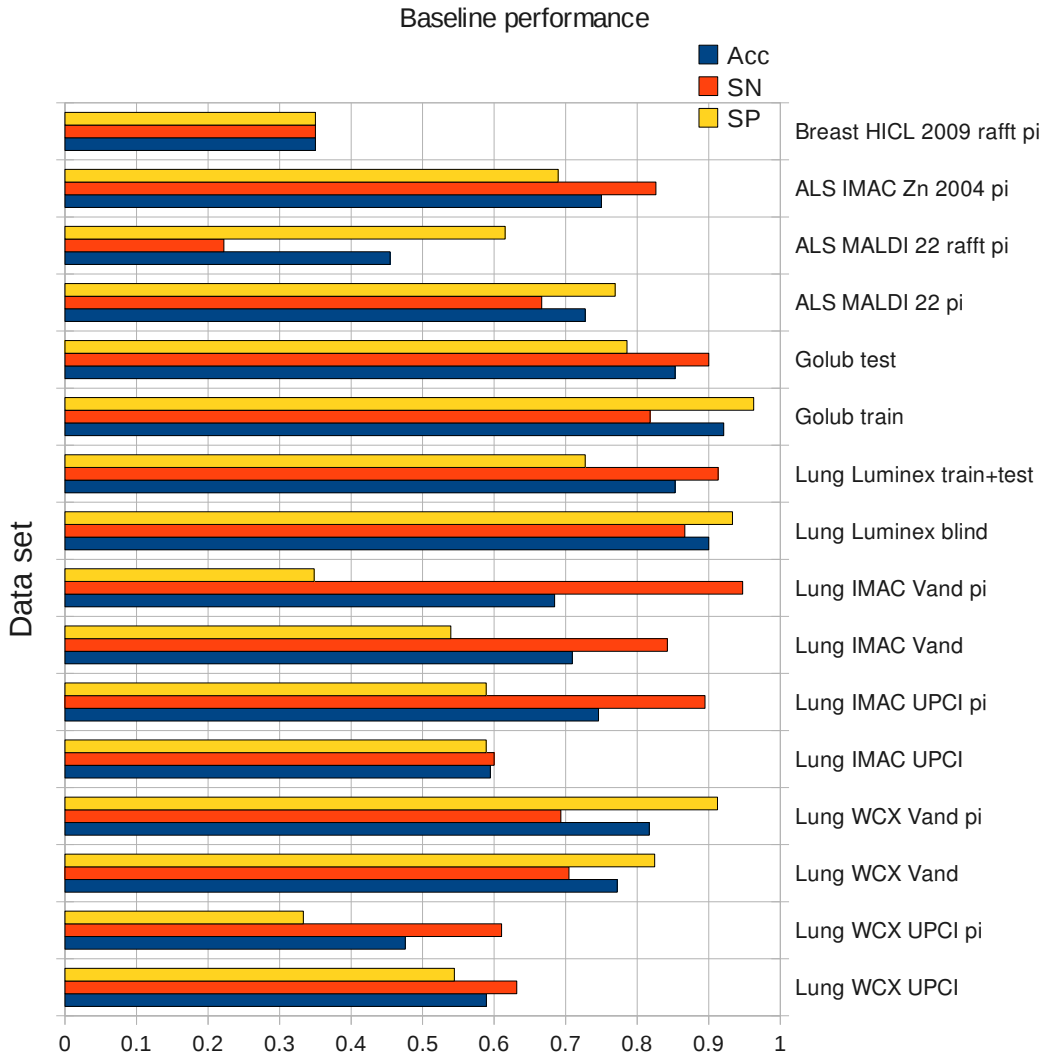


Figure 11: Baseline performances with RL and 10-fold cross validation. For the meaning of the operations, see Table 4.

5.2 SIMPLE RULE TRANSFER

This section presents the results of experiments with transfer of whole rules, including performance compared to the baselines, and amount of transfer. We begin with transfer experiments between subsets sampled from the same data (intra-set transfer), and then present results across data sets.

5.2.1 Varying the relative Sizes of the Source and Target Data

To check that transfer improves performance when the source and target data are known to be from the same distribution, we performed transfer between subsets of the same data set. We call this *intra-set transfer*. We used the genomic data from [Golub et al., 1999] because it is a well-known data set with a good classification performance. Moreover, we wanted to explore how the performance after transfer and the amount of transfer are affected by the relative sizes of the source and target data sets. For this purpose, successive target sets of different sizes were sampled at random from the whole data set, and the remainder of the data was used as source data. The sizes of the source and target data sets represent a more general issue of the generalization performance of the classifiers learned from the data sets. Indeed, in a practical application of transfer, the source and target classifiers might have different generalization performance due to size or different amounts of noise in the data sets. As an illustration, the absolute performance with different subset sizes sampled from all the data, is shown in Figure 12. The data sizes range ranging from 10% to 90% of all the data, and the performance is the average of 10 x 5-fold cross-validation. (Using 10-fold cross-validation was not possible with 10% of the data because there are only 72 instances.) We see a progressive increase in accuracy from 50% (with 10% of the data) to 90% (70% of the data), and a dramatic increase in specificity from 10% to 80%. Thus, when doing transfer from a 80% subset to a 20% subset, this represents transfer from a model with 90% accuracy to a model with 50% accuracy.

Figure 13 shows the change in performance after the simple rule transfer compared to using only the target data. As expected, this works well because the source and target

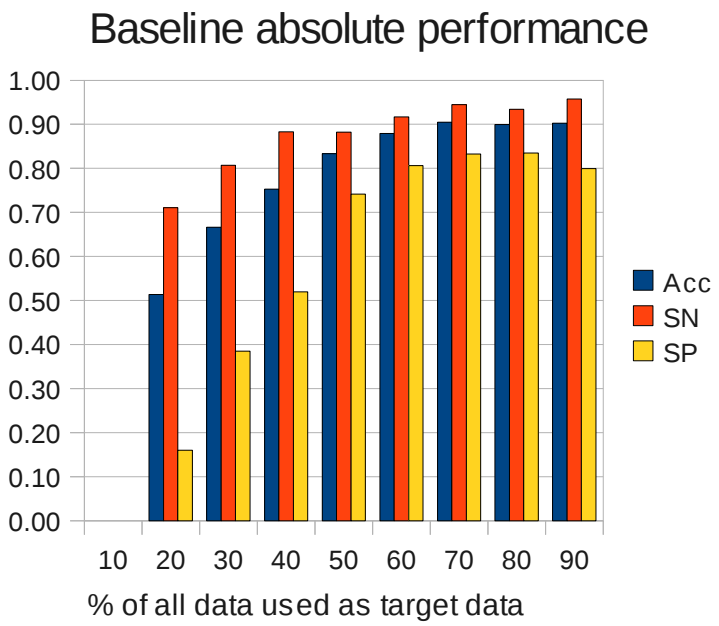


Figure 12: Performance on subsets of different sizes, sampled from the gene expression data [Golub et al., 1999]. Horizontal axis: percentage of data set used as target. Averages of 10 random source/target splits, with 5-fold cross-validation. Vertical axis: performance.

data are from the same distribution. Transfer never decreases performance, and significantly increases performance when the source data set is large compared to the target data, that is when the source classifier generalizes better than the classifier learned on the target data. The improvement decreases as the size of the target data increases. An exception to that trend occurs with 10% of the target data; in this case there is no benefit from transfer because the target data is too small to accurately evaluate the goodness of a classifier, and all the transferred rules are discarded. In fact, no rules are learned at all on the target data set of 7 examples, regardless whether transfer is used. This should not be very surprising because we have a minimum coverage requirement of 4 examples for new rules to be added to the rule set. When we have only 7 examples, a feature would have to predict more than half of them exactly correctly in order to be retained. However, trying to learn such a high

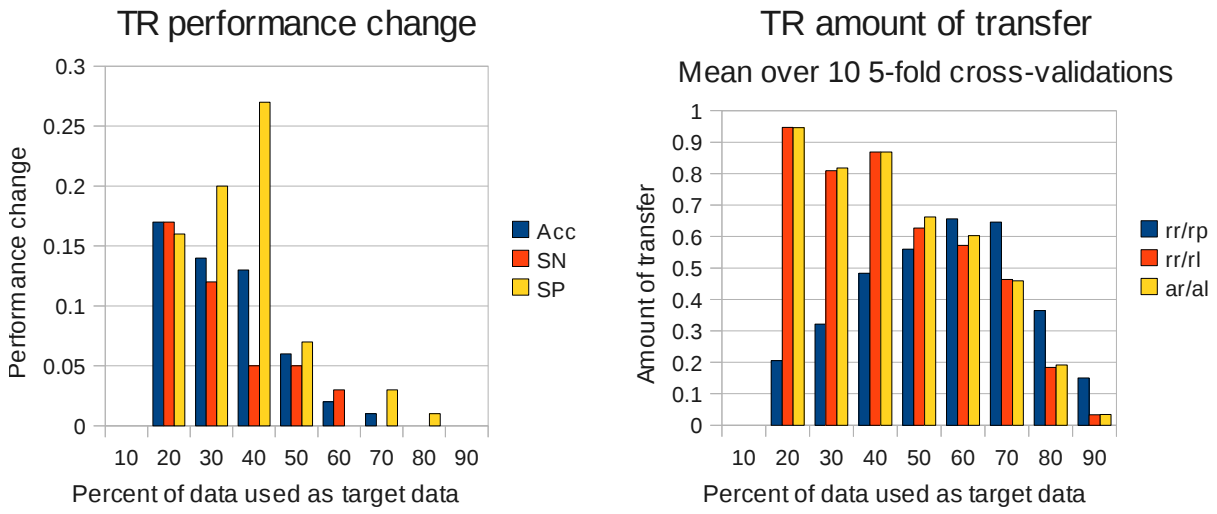


Figure 13: Performance change and amount of transfer in gene expression data [Golub et al., 1999] with simple rule transfer compared to target data only. Left panel: change in performance. Right panel: amount of transfer. Target data was sampled randomly, and remaining data was used as source. All numbers are averages of 10 random source/target splits, with 5-fold cross-validation for the target data.

dimensional problem with only 7 training examples is unlikely to ever be very effective, since the learning procedure would be swamped by noise.

A similar trend is seen with the number of rules and variables transferred in the right panel of Figure 13. Recall that rr/rp represents number of rules retained as a proportion of all prior rules; rr/rl: the number of rules retained as a proportion of all rules learned; and ar/al: the number of attributes in the retained rules as a proportion of all attributes in the learned rules. When the target data are small, the proportion of all learned rules that were retained from the source classifier (rr/rl) is large, and decreases as the source data decreases relative to the target data. With 80% of the data used as source, as much as 90% of the rules learned after transfer are prior rules learned from the source data. This is because the source data allows the learner to learn many good rules, and most of the rules we learn from the target data come from prior rules. The same trend is seen for the fraction

of attributes in the target classifier that were in the source classifier (ar/al), for the same reason. By contrast, the proportion of prior rules that are retained (rr/rp) is greatest when the source and target data sets are about equal. This is because the large source data sets give rise to large prior models, and so the fraction of those prior rules retained is small. As the size of the source data decreases and the number of prior rules decreases, increasing the ratio rr/rp. As the size of the source data decreases further, the quality of the rules learned also decreases and even though some rules are learned, many of them are rejected when the target data are considered again leading to a small rr/rp ratio.

5.2.2 Simple Rule Transfer Across All Data Sets

Next we consider the results of simple rule transfer across data sets where the source and target data that are generated from independent measurements. The data sets to be used were selected in Section 5.1 and described in Section 4.1. The mean changes in performance with different transfer methods are shown in Figures 14 and 15. Figure 14 considers abstentions as errors, while Figure 15 considers abstentions as predictions for control. Because Figures 14 and 15 contain a lot of numbers, the trends might not be entirely clear. If we compute the average change in performance for each of the performance measures: accuracy, sensitivity and specificity we see that on average, transfer learning improved all measures of performance. However, the improvement was very small, about 0.5% increase in accuracy. The increase is greater for sensitivity than for specificity. Additionally, the standard deviation of the change is several times larger than the increase, and is greatest for specificity where the standard deviation is 5%. In several cases, transfer substantially increases the number of abstentions.

Notably, whenever transfer decreases performance, transfer in the opposite direction never decreases performance. Moreover, the average is lowered by two significant drops in performance, namely ALS MALDI (22 examples, -0.09) transferred from IMAC (2004 data, 52 examples, -0.12) and Lung Cancer IMAC UPCI peak-selected transfer from Vanderbilt. In both cases, there was a large increase in performance when transferring in the other direction. To remove the effect of these outliers, we can consider the ratio of the number of

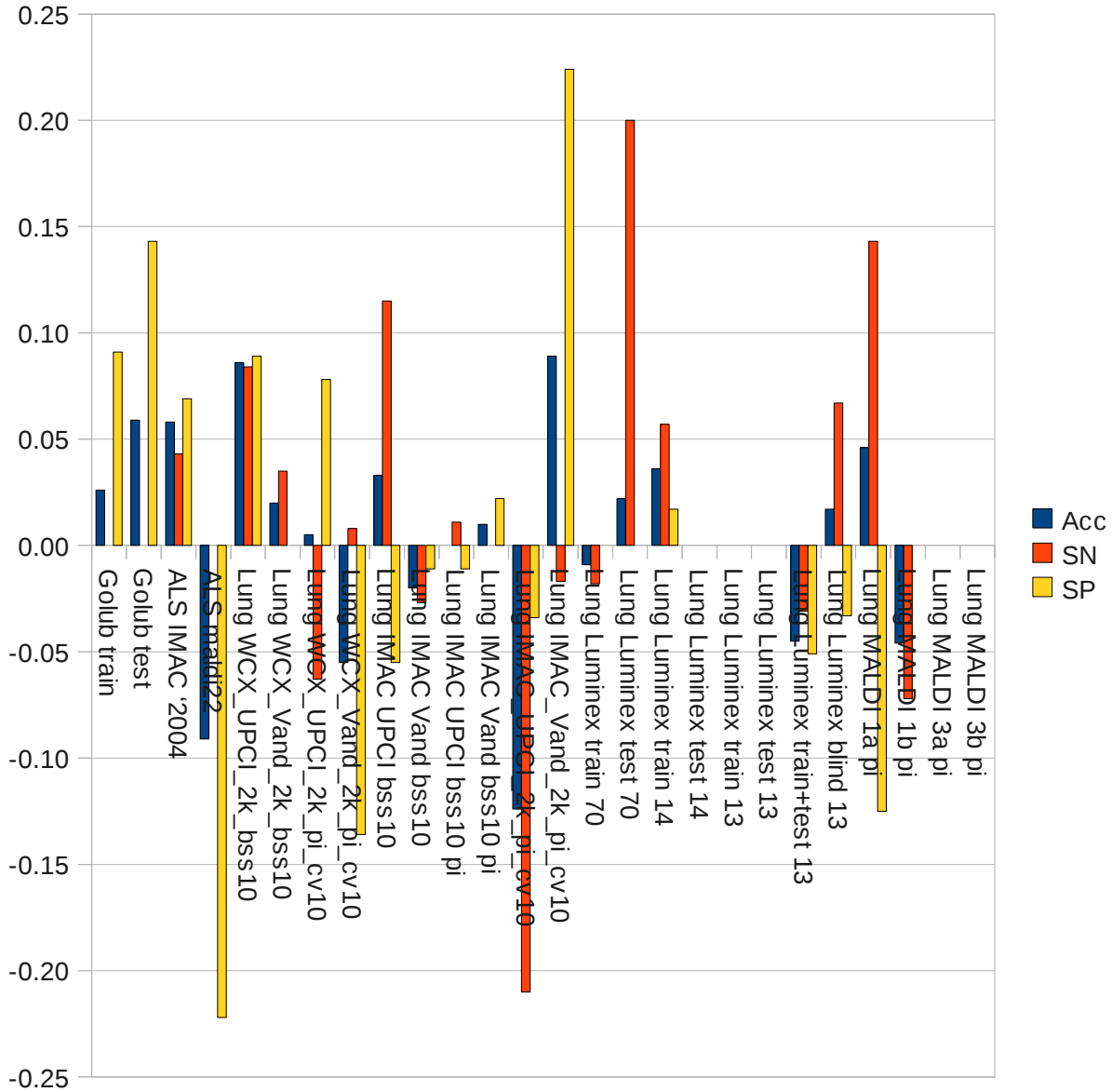


Figure 14: Mean change in performance when abstentions are considered errors. Change for each of the 26 data sets, compared to training on the target data set alone, for each data set. Means are for cross-validation as listed in Table 5, but usually 10-fold.

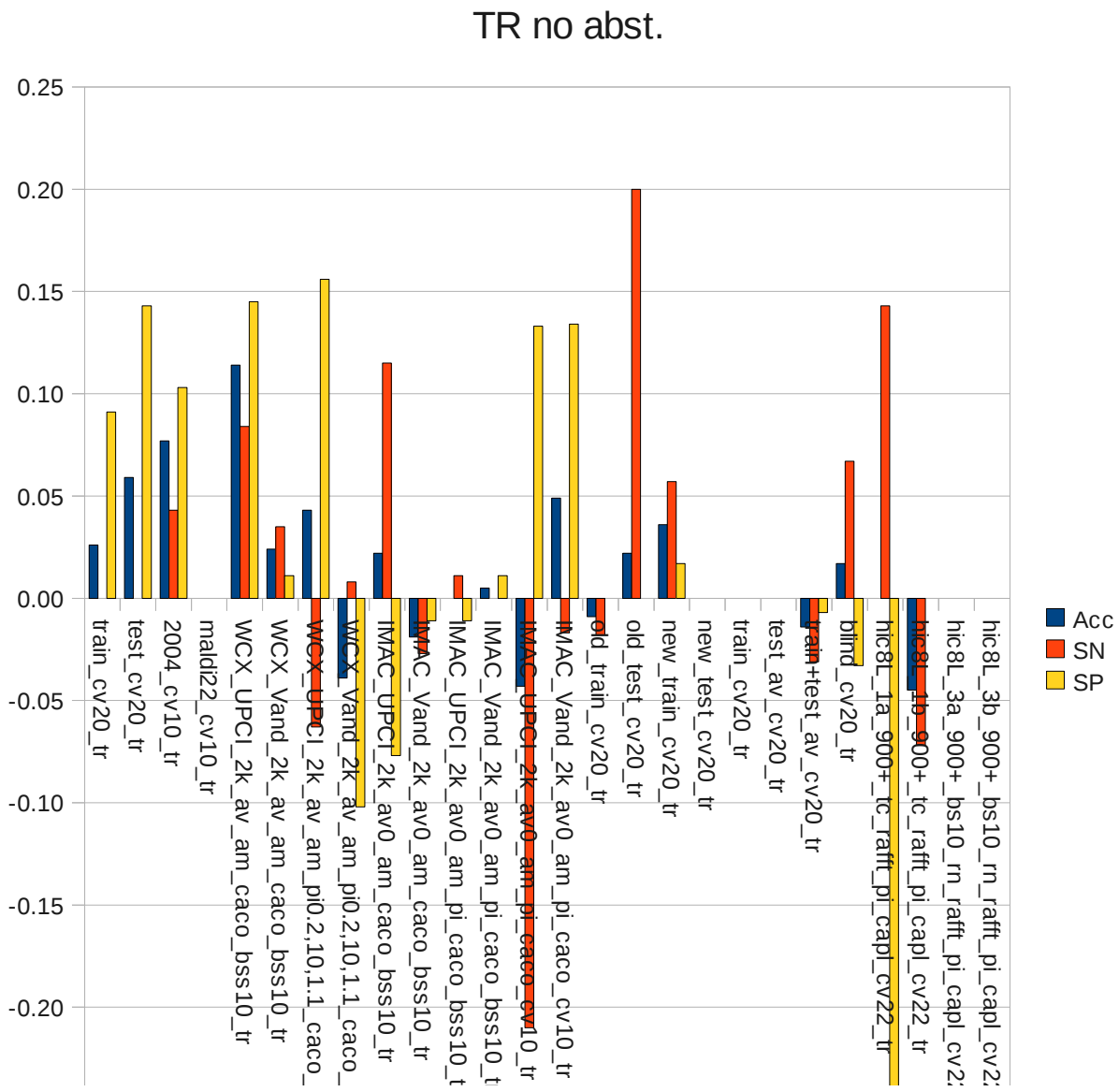
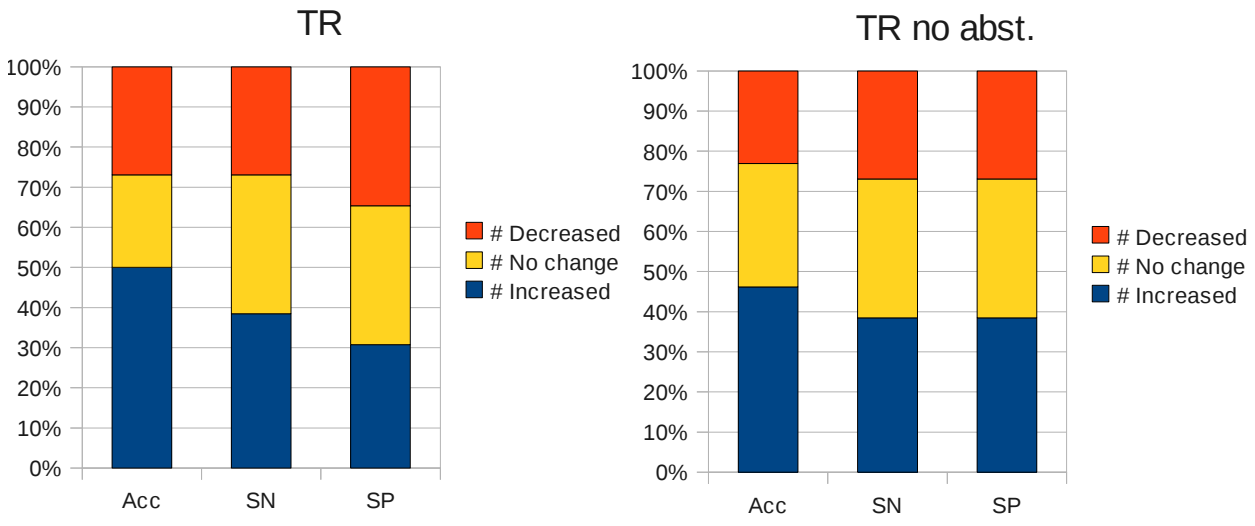


Figure 15: Mean change in performance when abstentions are considered predictions for Control. Change for each of the 26 data sets, compared to training on the target data set alone, for each data set. Means are for cross-validation as listed in Table 5, but usually 10-fold.

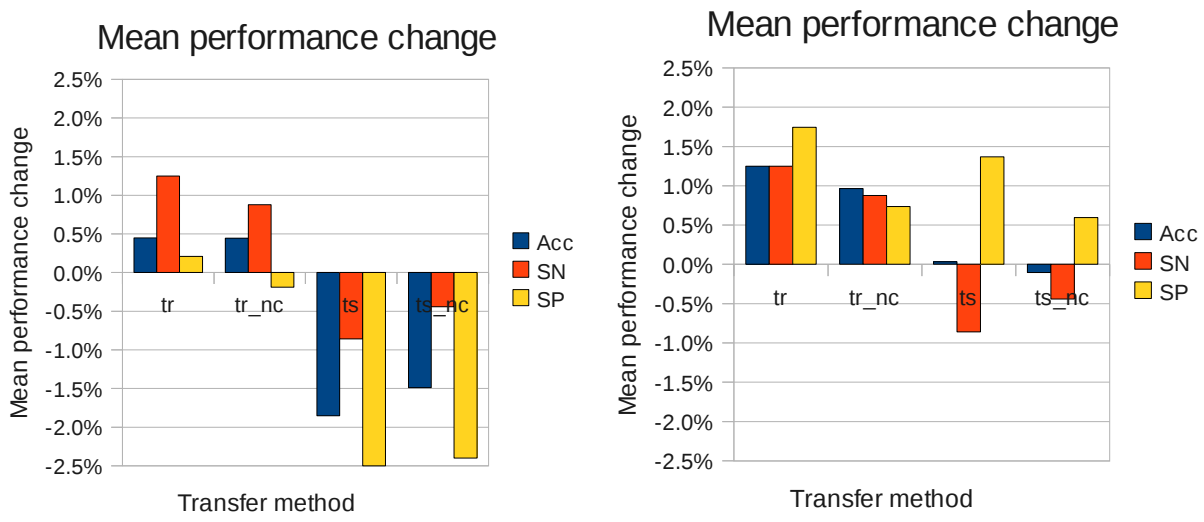


(a) (b)

Figure 16: The number of experiments, of 26, for which the performance increased, did not change, or decreased after transfer. (a) Abstentions treated as errors. (b) Abstentions treated as predictions for control

times performance increased, was not changed, or decreased after transfer, or the "win-tie-lose" ratio. The ratio is 0.50 : 0.23 : 0.27, which shows that transfer increased performance in most cases, but by a small amount. The ratios are shown visually in Figure 16.

When considering abstentions as negative data instances (controls), the improvement in accuracy and specificity were greater, 1% and 2% respectively, and the improvement in sensitivity was the same. The win-tie-lose ratio (Figure 16 (b)) is very similar to when considering abstentions as incorrect predictions. It is interesting that whenever transfer in one direction decreased performance, transfer in the other direction never decreased it and usually increased it. It is not always the case that transferring from the higher-performing data set to the lower-performing one improves performance. In fact in most cases, the baseline performance of the two data sets is comparable – for example, the lung cancer IMAC data from UPCI and Vanderbilt with peak selection have accuracies 63% and 61% respectively, while transferring from Vanderbilt to UPCI hurts accuracy the most among all



(a) (b)

Figure 17: Mean change in performance compared to training on the target data set alone, for each transfer method, mean over all 26 data sets. (a) Abstentions are considered errors. (b) Abstentions are considered predictions for control.

data sets, and transferring from UPCI to Vanderbilt improves accuracy the most among all data sets.

5.3 COMPARISON OF THE TRANSFER METHODS

This section examines the results with the four methods of transfer across the 26 data sets. The data sets to be used were selected in Section 5.1 and described in Section 4.1.

5.3.1 Effect on Performance

The changes in performance and standard deviations for each transfer method averaged over cross-validations of the data sets are shown in Figure 17. The mean performance changes were small relative to their standard deviations shown in Figure 18.

Figure 17 (a) counts abstentions as errors. In this case, there was an improvement in accuracy and sensitivity on average with simple rule transfer (the "tr" and "tr_nc" conditions). On the other hand, structure transfer ("ts" and "ts_nc" conditions) decreased performance. Specificity either decreased or was virtually unchanged. When we treat abstentions as predictions for control as shown in Figure 17 (b), we see some of the same trends. Specifically, rule transfer (the "tr" and "tr_nc" conditions in Figure 17 (b)) improves performance, while the structure transfer approaches (the "ts" and "ts_nc" conditions in Figure 17 (b)) have mixed results. However, evaluating in this way, we always see improvements in specificity from all transfer methods and smaller improvements in sensitivity for the rule transfer methods. Comparing the two panels in Figure 17, we see that the structure transfer models tend to abstain more than the baselines. Consequently they lead to lower accuracy compared to the baselines, when we score abstentions as errors. By contrast, when we count abstentions as predictions for control cases, they tend to have similar accuracy to the baseline models.

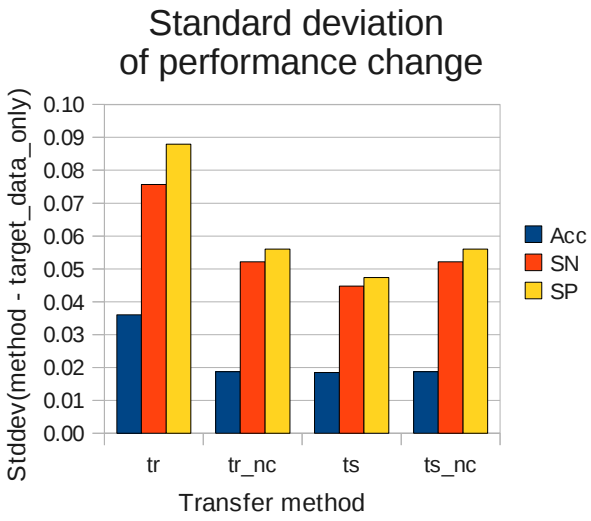
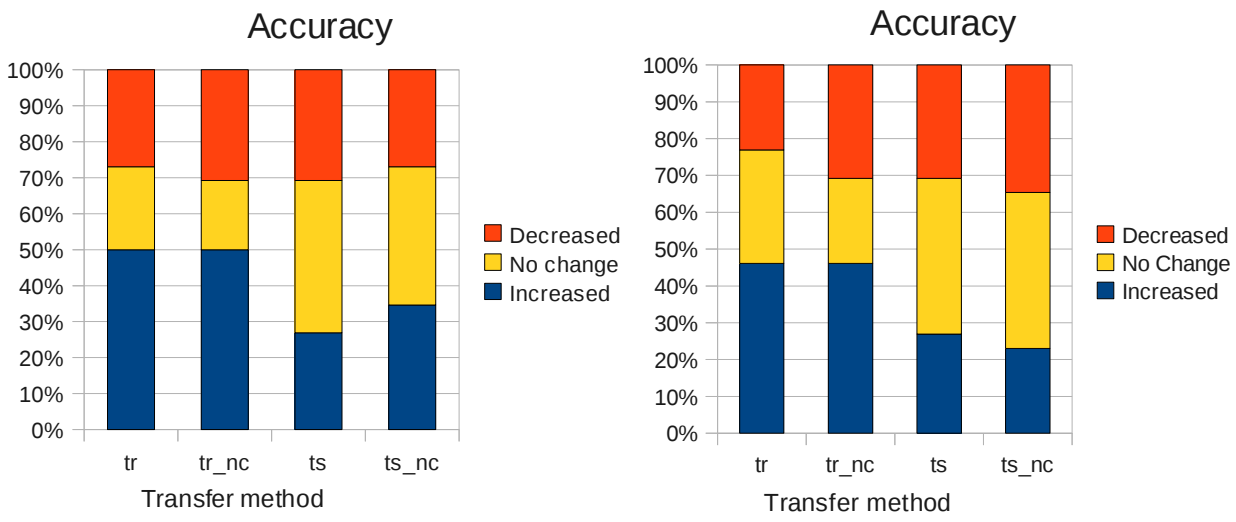


Figure 18: Standard deviation of the changes in performance for each method over the 26 data sets, compared to training on the target data set alone. Abstentions are considered predictions for Control; the standard deviation is taken over the cross-validation mean for each data set.



(a) (b)

Figure 19: Number of improvements, ties and decreases in accuracy after each method of transfer, compared to training on the target data set alone. (a) Considering abstentions as errors. (b) Considering abstentions as predictions for control. Counts are over the 26 data sets, and performance was on cross-validation of each data set. (See Section 4.1.)

We can also examine for each transfer method, how often it increased, decreased or had no effect on performance compared to using the target data alone. Figure 19 (a) shows this ratio for accuracy when considering abstentions as errors. We see that simple rule transfer methods ("tr" and "tr-nc") improved accuracy about 50% of the time and decreased it about 25% of the time. Structure transfer had a much smaller proportion of improvements in accuracy and much larger proportion in no-changes. The trend also held when considering abstentions as predictions for control, as we can see in Figure 19 (b); simple rule transfer increased accuracy almost 50% of the time and decreased it about 25% of the time; while structure transfer had a much smaller proportion of increases which was similar to the proportion of decreases. This is consistent with the results we observed for the means in Figure 17.

Thus, overall simple rule transfer (the "tr" and "tr-nc" conditions) increased performance

(accuracy, sensitivity, specificity) significantly more often than decreased it, and on average made a small increase in performance. That simple rule transfer performed better than structure transfer is surprising because structure transfer was intended as a way to overcome the difference in discretization between the source data and target data, and thus make prior rules more generalizable on the target data. Recall that to overcome the differences in discretization, structure transfer instantiates every possible rule of the prior structure with the target variable-values, and adds all those rules to the beam for processing. This includes negation of the original prior rule. All the instantiated rules are then specialized during search, so the final model may also include those instantiations. Thus this result suggests that the benefit gained from more accurate discretization of the prior rules comes at the cost of adding less accurate rules to the final classifier, which more often than not mis-classify target examples.

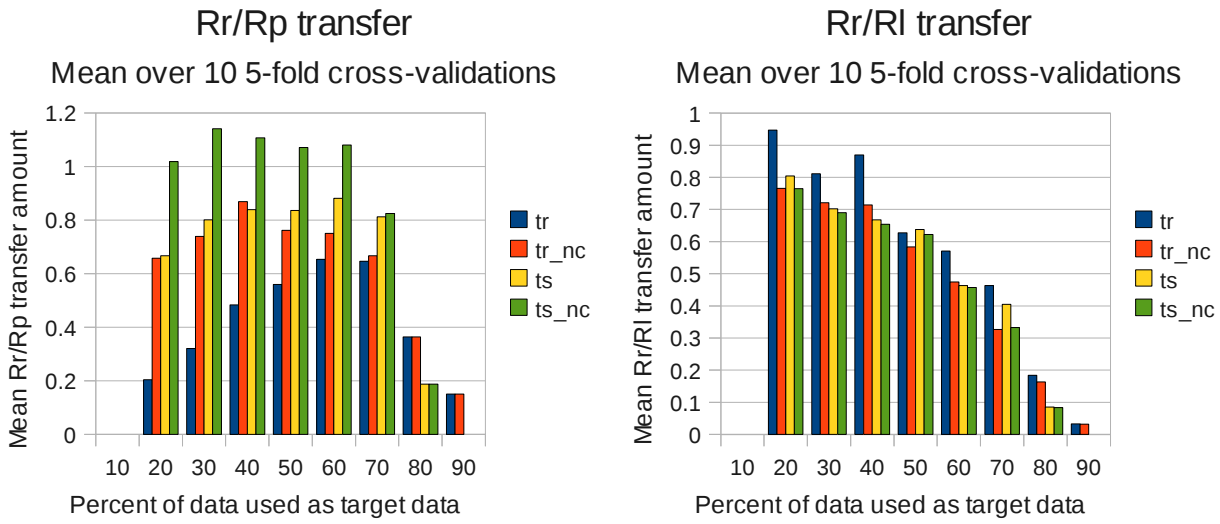
Another general trend in the results is that the no-coverage ("nc") transfer conditions show a less pronounced change than transfer where prior rules affect the coverage of examples. This is interesting because the no-coverage condition allows interaction of prior rules with new rules during learning only via the beam. However, rules do interact during inference. Thus the result suggests that it is beneficial to take account of these interactions between all rules via coverage of examples training rather than ignore them.

We note that for the rr/rl measure of amount of transfer, the highest amount of transfer is achieved with the whole-rule transfer method.

5.3.2 Amount of Transfer

In addition to performance, we are interested in the number or rules that are retained from the source to the target models. This measures to some extent the amount of knowledge that has been transferred from one model to the other. We will consider the number of retained rules as a fraction of the total prior rules and also as a fraction of the total learned rules.

First we consider the amount of transfer when we sub-sample the [Golub et al., 1999] gene expression data and transfer from one part of the data set to the other. The performance for these experiments were discussed in Section 5.2.1. In Figure 20 (a), we see the rules retained



(a) (b)

Figure 20: Transfer amount (intra-set) with varying sizes of source data for the gene expression data [Golub et al., 1999], as we vary the fraction of data used as a target set. (a) Rules retained as a fraction of prior rules. (b) Rules retained as a fraction of learned rules. Vertical axis: ratio of rules retained to prior rules. Horizontal axis: percentage of data set used as target. Averages of 10 random source/target splits, with 5-fold cross-validation.

as a fraction of the prior rules. Note that in the case of structure transfer with the no coverage condition, this ratio can be more than 1, since the model can include both a rule and its inverse. For example, if the prior rule was `IF` $\texttt{MZ}_{12.34}$ `= High THEN Class=Control`, then we could include in the transferred model both `IF` $\texttt{MZ}_{12.34}$ `= High THEN Class=Control` and also `IF` $\texttt{MZ}_{12.34}$ `= Low THEN Class=Disease`. Focusing first on the simple rule transfer, we see that we are not able to transfer a lot of rules from the source model when there is almost no target data. This is because even though the source rules are good, we do not have enough target data to verify this, and we are only able to use a few of them. As we increase the amount of target data, we see an increase in the fraction of prior rules that we are able to retain, peaking at retaining around 70% of the prior rules. As we increase the proportion of data used as target further, we start to decrease the amount of source data, and now the

prior rules are not retained because they are of poorer quality and the target data reject them. Note, however that the performance improvements also decrease as the amount of target data decreases, as we saw in Figure 13. This is because the baseline performance is improving, so even though we are able to retain a larger fraction of the rules when the source and target data are balanced, the performance *increase* is smaller than when we retain just a few rules, but the baseline is lower. The ratio of retained to learned rules, by contrast, is monotonically decreasing, since the quality of the prior rules is monotonically decreasing. When there is very little target data the model cannot verify all of the prior rules, but most of the rules it learns are retained rules. As the amount of target data increases, it is able to find new good rules, and reject bad prior rules.

We see in Figure 20 that for basic transfer with the no coverage condition, overall a larger fraction of prior rules are retained than under basic transfer. Recall that in order for rules to exist in the source model and be available for transfer, they need to be supported by different training examples. However, by allowing them to be supported by the same examples in the target data, we are able to retain a larger fraction of them than if we required them to be supported by different examples in the target data also. However, when we look at retained rules as a fraction of learned rules there is no such pattern, and in fact adding the no-coverage condition decreases the fraction of learned rules that are retained from the prior model. This is because the no coverage condition allows the model to learn more rules. Specifically it allows the model to learn more new rules, since prior rules are added to the beam in front of new rules.

Considering next the structure transfer setting, without a no coverage condition, we see the same pattern as with tr_nc. Overall, we retain more rules as a fraction of prior rules. However, the reasons are different than for the result with tr_nc. In the case of structure transfer, this is because we can consider both rules and variations of those rules, and search for support in the target data. This allows us to consider a larger set of rules to transfer, and it is more likely that the target data will have support for a larger number of them. Finally, the ts_nc line in Figure 20, represents both rule structure transfer and the no coverage condition. This has an even larger ratio of retained rules to prior rules, since we can not only consider multiple variations of a rule – such as its negation and contra-positive, but we

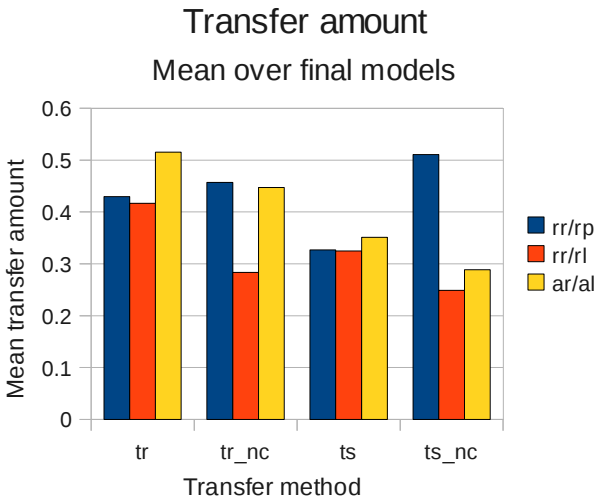


Figure 21: Amount of transfer with the four methods, means over the 26 data sets in their respective cross-validations. (The cross-validations are listed in Table 5). rr/rp: proportion of rules retained after transfer out of all the prior rules; rr/rl: proportion of prior rules retained out of all rules in the final model; ar/al: proportion of attributes in the retained prior rules out of of all the attributes in the final model.

can also use the same examples to validate different prior rules. In fact, we sometimes see a ratio greater than 1. Since more rules are learned under this setting, the ratio of retained rules to learned rules is similar to the structure transfer setting.

The transfer amounts for transfer between data sets are shown in Figure 21. The numbers for the mean amounts of transfer over cross-validation folds are very similar. We see that the number of rules retained as a proportion of rules in the final model (rr/rl) was smaller with the no coverage condition , for both simple rule transfer and structure transfer (tr_nc and ts_nc, respectively in Figure 21). When we have a no-coverage condition, we can learn more rules overall since the prior rules that are retained cannot interfere with the new rules that we consider. Consequently the number of learned rules will be greater and the ratio of retained rules to learned rules will be smaller.

A similar trend can be seen with the ratio of attributes in the retained rules to attributes

in the learned rules (ar/al). This proportion was greater than rr/rl for every transfer method, which means that on average the retained prior rules were larger than the rest of the learned rules. This might be because we consider specializations of prior rules to be "retained" rules, and since prior rules are placed first on the beam, we would expect them to be specialized more frequently.

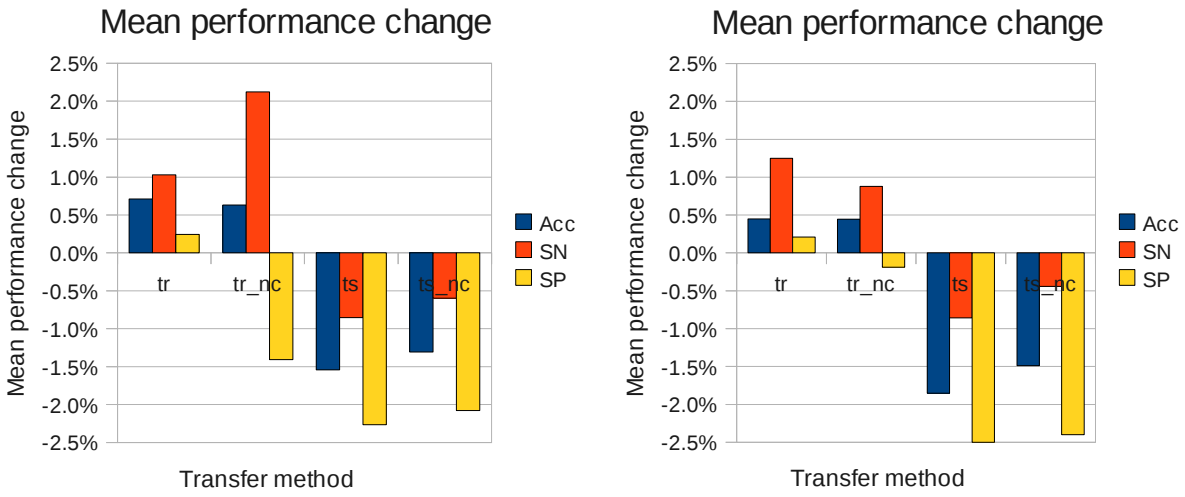
Finally, we can see in Figure 21 that the proportion of retained rules to prior rules was higher with the no coverage condition. The reason is that with the no coverage condition the prior rules do not conflict with each other, and the model can include a larger fraction of them.

5.4 SCALING OF THE DATA SETS

As explained in Section 4, it is beneficial to try to remove systematic measurement errors in the data before transfer learning. TRL provides a way to do that by scaling the data to equalize the means and variances of the source data to those of the target data.

We saw in Section 3.4, TRL is able to scale the source and target data in order to equalize their mean and standard deviation. If they are drawn from the same distribution of biological samples, and the data set sizes are large enough, this should counteract the systematic measurement errors.

We tested this hypothesis by normalizing the source and target data sets to have the same mean and standard deviation for each feature. Because Efficient Bayesian discretization (EBD, [Lustgarten, 2009]) only considers the order of feature values and not their absolute values, this does not change baseline results. The mean performance changes over all the data sets when we perform scaling are shown in Figure 22 (a). For comparison, Figure 22 (b) has the mean performance changes without scaling. We see that all the qualitative conclusions hold for the scaled data as for the unscaled data, except that the mean changes are smaller in magnitude. This is confirmed by a comparison of the number of data sets for which performance increases, does not change, or decreases in Figure 23. We see that the general trends are preserved, but that the fraction of data sets for which performance did



(a) (b)

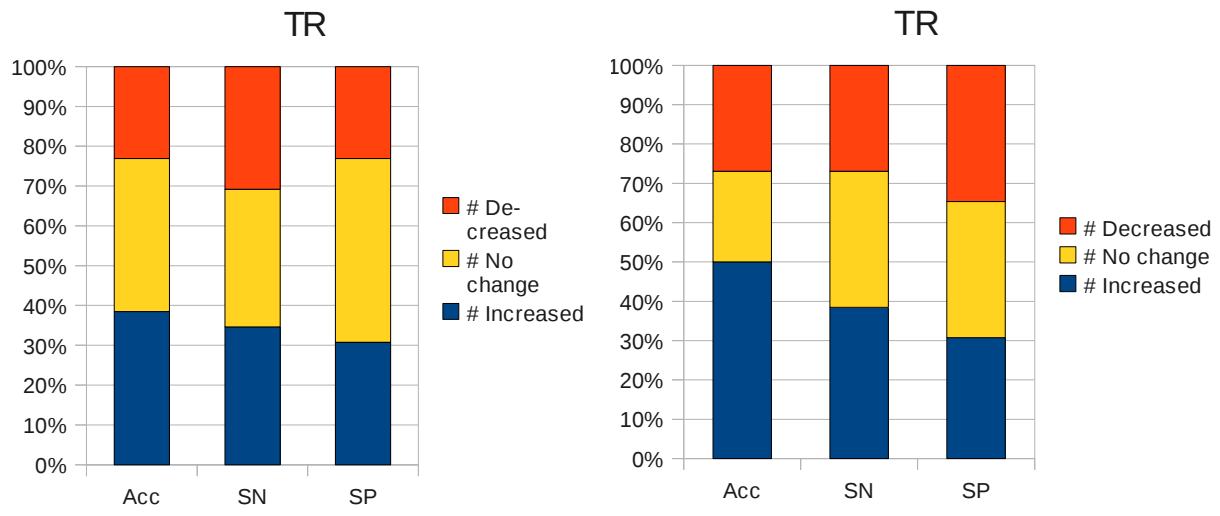
Figure 22: Comparison of mean performance change as compared to using only the target data set with and without scaling. For each evaluation metric, the mean is taken over all 26 data sets. Left: With scaling. Right: Without scaling. Abstentions are treated as errors.

not change is greater in Figure 23 (a) with scaling than in Figure 23 (b) without scaling. One possible explanation for this is that because most of the data sets are small, performing scaling actually distorts the source data, making it less useful for transfer learning. The rules learned on the source data are not as useful on the target data, and fewer of them are retained. This results in more data sets with no change in performance as compared with transfer without scaling.

5.5 SUMMARY OF RESULTS

To summarize the results, we can make the following observations:

1. Simple transfer increases accuracy 50% of the time and does not decrease performance 75% of the time.



(a) (b)

Figure 23: Comparison of the number of experiments, of 26, for which the performance increased, did not change, or decreased after transfer. Left: with scaling. Right: without scaling. Abstentions treated as errors.

2. Simple transfer increases accuracy by a small amount on average (0.5% with about 5% standard deviation).
3. Structure transfer decreases accuracy by about 1.5% on average.
4. The no-coverage transfer variations produce smaller changes in performance on average than the with-coverage variations.
5. When the source and target data are drawn from the same distribution, for all transfer methods improvement in model performance depends on the accuracy of the source model and baseline target model:
    a. The more accurate the source model compared to the model learned on the target data alone, the greater the improvement in performance after transfer.
    b. When the target data is too inaccurate (about 50% accuracy), there is very little or no improvement in performance after transfer.
6. When the source and target data are drawn from the same distribution, for all transfer

methods the amount of transfer depends on the accuracy of the source model and baseline target model:

   a. The more accurate the source model compared to the model learned on the whole data alone, the greater the amount of transfer.
   b. When the target data is too inaccurate (about 50% accuracy), there is very little or no transfer.

7. Scaling the source and target data to equalize their means and standard deviations before transfer results in smaller changes in performance than without scaling.

6.0 CONCLUSION

This thesis developed TRL, a novel framework for transfer learning with interpretable rule classifiers, using four transfer algorithm variations. It demonstrated how TRL can be applied to biomarker discovery and evaluated the methods on 26 real-world clinical data sets from biomarker discovery studies. Of the four methods, simple rule transfer was the most beneficial and result in the highest amount of transfer. Simple rule transfer increases performance much more often than decreases it. Unfortunately, with all the TRL methods, the effect on performance varies a lot from problem to problem.

One direction for future work is to confirm our observations about transfer learning in biomarker discovery in other data sets and with other transfer learning methods. For example, it might be the case that for some data, the most important information is the identity of the relevant variables, and the variable values change between domains and only serve to confuse the transfer learner.

This thesis concentrated on applying TRL to biomarker discovery in the setting where the source data and target data have some variables that are the same between the two data sets. However, TRL can be applied whenever a mapping exists between variables of one domain and variables of the other.

Our experiments with TRL on biomarker discovery data sets also allowed us to explore what kinds of information it is necessary to preserve when transferring rules from one study to another, as well as how tightly coupled learning needs to be to inference. This information will be important for designing future transfer learning algorithms.s

6.0.1 Future work

The work in this thesis opens new avenues for research into transfer rule learning. One direction of research is to define some more general mappings between different types of source and target domains, allowing transfer between different types of data, such as between proteomic and genomic data. For example, such a mapping could make use of the known mass of a protein when it is expressed from a gene studied in the source gene expression data set.

Another direction for research is to allow TRL to correct errors in variable naming, such as m/z shift and smearing in mass spectrometry. Recall that an m/z shift occurs when the m/z recorded for a protein in a mass spectrum is not the true m/z of the protein, and different mass spectra can have different shifts. Similarly, smearing of the peaks occurs because the spectrum records a tapered peak of m/z's instead of a single m/z. These phenomena could be addressed by constructing compound variables while specializing rules during the search. For example, a compound variable could be a disjunction of input variables in the data and would be used as usual variables, representing that when any of the m/z's in a particular range are above threshold, the protein is up-regulated in the sample. The same mechanism can be used to search for multiply-charged peaks from the same protein. Which input variables to consider for as candidates for constructing compound variables can be defined by by appropriate heuristics.

In Section 5, we observed that the variable values of prior rules are important information during transfer. This is reflected in the result that simple rule transfer performed better than rule structure transfer. Therefore, TRL can be refined to make it use simple rule transfer when possible and only use structure transfer to resolve a mismatch in discretization between source and target data. In particular, if a prior rule has a variable that has a different number of discrete values in the target data than the source data where the rule was learned, then TRL would use structure transfer for that variable, i.e., instantiate rules with all values of that variable.

A current limitation of TRL is that prior rules are never generalized. This is a problem because it is possible that a rule scores higher than its parent in the source data, but fails

a coverage threshold and is so is not retained on transfer into the target data, where the parent rule would have transferred some of the prior information. This can be addressed by importing all ancestors of prior rules during transfer.

Another extension of TRL is to use the conditional probability distributions of prior rules on the source data and update those on the target data. This would be a combination of the approach of DEFT and current TRL. The purpose of this is to achieve more accurate estimates of rule goodness.

Using the TRL framework, it would be interesting to know how transfer and performance are affected if prior rules do not at all affect the search of prior rules - neither by covering the training data nor by displacing rules from the beam. This can be accomplished by simply evaluating the prior rules without specializing them, and using them to make predictions alongside newly learned rules. Another way to accomplish this is to use two beams for search instead of just one: one beam for specializing prior rules and another beam for searching with new rules. This would mean that the same rules are learned on the target data regardless of whether prior rules are provided for transfer. This might result in more accurate posterior classifiers.

APPENDIX A

GLOSSARY

This section defines some concepts and terminology used in the rest of the document.

Given a set of input vectors (also called data vectors, data points, instances and learning examples) x in a vector space X and their corresponding classes $f(x)$ in a set of classes Y, the goal of classification learning is to find a classifier, that is a function

$$h : F \rightarrow X \rightarrow f(X) \tag{A.1}$$

that maps an object x to its class $f(x)$.

The dimensions of the data vectors $X \times Y$ are also called variables or features. The variables in X are called input variables, while Y is called the output variable or class variable. In more general learning settings, Y may contain more than one output variable.

A *learning bias* (also called *inductive bias*) of a learner is the set of assumptions that the learner uses to predict outputs given inputs it has not encountered [Utgoff, 1984, Mitchell, 1980].

The confusion matrix of a classifier evaluated on a test data set is a table where each row represents the examples of a predicted class, and each column represents the examples of an actual class. Thus, cell (i, j) in the matrix represents the number of examples of class j that the classifier predicted to be class i. Table 7 is an example of a confusion matrix for a binary classification problem. The matrix in Table 7 shows that of the predicted examples, 10 were correctly classified as cancer (true positives for cancer), 13 were correctly classified

	Actual	
	Cancer	Control
Predicted cancer	10	2
Predicted control	4	13

	Actual	
	Cancer	Control
Predicted cancer	TP	FP
Predicted control	FN	TN

Table 7: Left:An example confusion matrix. Right: abbreviations T for true, F for false, P for positives and N for negatives.

as controls (true negatives), 2 controls were classified as cancer (false positives), and 4 cancer examples were classified as controls (false negatives).

The accuracy of a classifier is the probability that the classifier will correctly predict an instance from the population

$$P(\text{correct prediction}) = P(h(x) = f(x)) \tag{A.2}$$

and is estimated from the classifier's predictions on an unseen test data set as:

$$\frac{\text{Number of correct predictions}}{\text{Number of predictions}} \tag{A.3}$$

The *sensitivity* of a classifier for class y is the probability of correctly predicting an instance of class y,

$$P(h(x) = y | f(x) = y) \tag{A.4}$$

and is estimated as:

$$\frac{\text{Number of correct predictions of class } y}{\text{Number of test instances of class } y} \tag{A.5}$$

The *specificity* of a classifier for a class y is the probability of correctly predicting an instance of class other than y:

$$P(h(x) \neq y | f(x) \neq y) \tag{A.6}$$

and is estimated as:

$$\frac{\text{Number of predictions of classes other than } y}{\text{Number of test instances of classes other than } y} \tag{A.7}$$

The *balanced accuracy* of a classifier for class y is the mean of the sensitivity and specificity:

$$\frac{\text{sensitivity} + \text{specificity}}{2} \tag{A.8}$$

Positive predictive value (PPV):

$$PPV = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}} \tag{A.9}$$

Signal-to-noise ratio (S/N):

$$S/N = \frac{\text{TruePositives}(\text{TrueNegatives} + \text{FalsePositives})}{\text{FalsePositives}(\text{TruePositives} + \text{False Negatives})} \tag{A.10}$$

Likelihood ratio (LR):

$$LR = \frac{\text{TruePositives}}{\text{FalsePositives}} \tag{A.11}$$

Chi-squared (χ^2) statistic for a variable with respect to the class variable:

$$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}}, \tag{A.12}$$

where: m is the number of discrete values (intervals) in the variable, k is the number of classes, A_{ij} is the number of examples in the i^{th} interval of the variable and the j^{th} class, $E_{ij} = E(A_{ij}) = R_i * C_j/N$, the expected frequency of A_{ij}, R_i is the number of examples in the ith interval of the variable, C_j is the number of examples in the jth class, N is the total number of examples.

APPENDIX B

RL PARAMETERS

1. *Min. coverage*: the minimum overall coverage any rule may have if it is to be kept on the beam
2. *Min. pos.*: the minimum positive coverage any rule may have
3. *Max. neg.*: the maximum negative coverage any rule may have
4. *Max. conjuncts*: the maximum number of conjuncts any rule may have
5. *Min. conjuncts*: the minimum number of conjuncts any rule may have
6. *Min. CF*: the minimum certainty factor any rule may have
7. *Inductive strengthening*: number of examples any rule must cover that are not covered by previous rules
8. *CF function*: certainty factor function; one of: Frequency, Frequency normalized, Frequency with Yates correction, Likelihood ratio, P-value, P-value right, Signal-to- noise, Laplace estimate, Laplace depth, Laplace extended, F-measure, Jaccard
9. *CF value*: certainty factor the minimum certainty factor value that any rule may have
10. *Prune specialized*: whether to stop specializing rules that satisfy the constraints
11. *Beam width*: the number of rules stored during evaluation
12. *Evidence gathering*: the method for combining learned rules to classify a new example when the example matches more than one rule.
    - Best, Most specific, Best specific, Best p-value, Weighted voting, Minimum weighted voting, Maximum likelihood ratio, Nearest neighbor, Most specific single best.

APPENDIX C

RL MANUAL

```
NAME
        BRL - Program for rule learning, variable selection and
        discretization, and data pre-processing

SYNOPSYS
        java [JAVA_PARAMETERS] -jar BRL.jar
            -lp LEARNING_PARAMETERS
            [-ppp PREPROCESSING_PARAMETERS]
            -dp DATA_PARAMETERS

        java [JAVA_PARAMETERS] bayesianrl.BRL
            -ppp PREPROCESSING_PARAMETES]
            -dp DATA_PARAMETERS

        java [JAVA_PARAMETERS] -jar BRL.jar
            CONVERSION_PARAMETERS

JAVA_PARAMETERS
    For Java parameters, see the Java manual.  The Java classpath must
    include the Weka classes (jar file). The classpath can be set in
    the CLASSPATH environment variable, or using the "-classpath" Java
    parameter.

LEARNING_PARAMETERS:
    BRL parameters are not case sensitive; for example, "-rgm" means
    the same as
    "-Rgm", "-RGm", etc.

    -rgm
            Rule generation method
            0  Simple rule learning
            1  Bayesian global rule learning:
```

```
            1 0   Greedy search (GS)
            1 1   Beam search (BS)
        2  Bayesian local rule learning:
            2 0   Decision Tree (DT) GS
            2 1   DT BS
            2 2   DT parallel greedy search (PGS)
            2 3   Decision Graph (DG) GS
            2 4   DG BS
            2 5   DG PGS

-cftype INDEX_VAL
        Function to compute the certainty factor value for each rule.
            0   Likelihood ratio (default)
            1   Positive predictive value:
                    TP / (TP + FP)
            2   Positive predictive value with Yates correction:
                    (TP + 0.05) / (TP + FP)     if TP > FP,
                    (TP - 0.05) / (TP + FP)     if TP < FP,
                    TP / (TP + FP)              otherwise.
            3   Positive predictive value, normalized for asymmetric
                class distributions:
                    1   if FP = 0
                    0   if TP + FP = 0
                    TP / (TP + FP * Pos / Neg)
            4   Laplace estimate:
                    (TP + 1) / (TP + FP + number_of_target_values + 1)
            5   Laplace extended:
                    (TP + k*m) / (TP + FP + k),
                where
                    k = 1 + number of target values
                    m = Pos / (Pos + Neg)
            6   Laplace extended with bias for short rules:
                    (TP + c * q) / (TP + FP + k),
                where
                    c = 1 + number of conjuncts in the rule
                    q = TP / (TP + TN)

        (Six other functions are implemented but not compiled for use by
        default.)

-inftype INDEX_VAL
        0   Weighted voting (default). Predict the highest-weighted
            class, where the weight of each class is the sum of
            certainty factors of rules predicting that class. If there
            is a tie, predicts class 0.
        1   Maximum likelihood ratio
        2   "Combine CF"
```

```
        3   Lowest p-value: use the rule with the lowest p-value
        4   Single best: use the rule with the highest certainty factor
        5   Minimum weighted voting. Like weighted voting, but use only
            the highest k rules to calculate the weight of each class,
            where k is the minimum number of rules voting for any class.
        6   Single best specific: use the rule with the highest worth
            (certainty divided by cost) and the highest number of
            cojuncts.
        7   Most specific single best: use the rule with the most
            conjuncts among rules with the highest certainty factor.
-mincf NUMBER
        The minimum certainty factor value that any rule in the model
        will have.  The default is 0.85.

-minconj NUMBER
        The minimum number of conjuncts in any rule in the model. The
        default is 1.

-maxconj NUMBER
        The maximum number of conjuncts in any rule in the model. The
        default is 5.

-specialize
        If this option is specified, when a rule is added to the model,
        RL will also check if some specializations of this rule should
        also be added to the model. If the option is omitted, RL stops
        specialization of a rule once it is found to satisfy the search
        constraints.

-cover NUMBER
        The minimum number of training examples that any rule in the
        model will cover. The default is 4.

-minTP DECIMAL
        The minimum true positive rate that any rule in the model will
        have. The default is 0.05. Valid values are in the range [0, 1].

-maxFP DECIMAL
        The maximum false positive rate that any rule in the model will
        have. This option is not set by default. Valid values are in the
        range [0, 1].

-indStr NUMBER
        The minimum number of previously uncovered examples that each
        new rule must cover. The default is 1.

-beam WIDTH
```

```
        The number of rules kept at any time to be specialized in the
        next iteration iteration. The default is 1000.

    -bss [--d DEPTH] [--f NUM_FOLDS]
        Do bias space search
      --d 0    Shallow search (default)
      --d 1    Medium depth search
      --d 2    Most exhaustive search; consumes a lot of time and memory
      --f NUM_FOLDDS
             Use internal cross-validation with NUM_FOLDS for
             validation

    -cv NUM_FOLDS
        Cross-fold validation with NUM_FOLDS folds. A negative NUM_FOLDS
        denotes leave-1-out.

    -rrv NUM_SETS PERCENT_SIZE
        Random resample validation

    -d      Discretize. The parameters are as described in
        PREPROCESSING_PARAMETERS, but the discrete intervals for each
        variable are computed based on the training data only, then
        applied to the test data.  If cross-validation is specified, the
        discretization is computed on the training subset separately for
        each fold.

    -r      Remove trivial variables, as in PREPROCESSING PARAMETERS

    -rrv NUM_SETS PERCENT_SIZE
        Random resample validation, as in PREPROCESSING PARAMETERS

    -svCVR  Save rules learned in each cross-validation fold

    -svCvD  Save the fold data sets

PREPROCESSING_PARAMETERS:
    These parameters specify operations to be performed on the whole
    training data set at once, before any rule learning.

    -d DISC_METHOD PARAMETER
        Discretize using DISC_METHOD with specified parameter
        DISC_METHOD Meaning         Type           PARAMETER
                 0  Gaussian        Unsupervised   Number of bins
                 1  EqualWidth      Unsupervised   Number of bins
                 2  EqualFreq       Unsupervised   Number of bins
                 3  OneR            Supervised     Number of instances
                 4  ErrorBased      Supervised     Max number of bins
```

```
                    5  D2S              Supervised     (none - max number of bins is 8.)
                    6  FayyadIraniMDL   Supervised     Number of bins
                    7  EBD              Supervised     c structure prior (use value "1")
                    8  MODL             Supervised     none?

            Example: EBD (2008) discretization with default parameter 1:
                -d 7 1

    -r      Remove trivial variables after discretization

    -chi NUM_OF_VARS_TO_SELECT
            Chi-squared variable selection: select the top
            NUM_VARS_TO_SELECT variables.

    -s SCALING_METHOD ...
            Scale each variable by the specified SCALING_METHOD in turn
            0   0-1 scaling
            1   Subtract local minimum
            2   Subtract global minimum
            3   Log2
            4   Square root
            5   Exponent 2
            6   Square
            7   Normalize to mean 0 and standard deviation 1

    -ctr
            Combine technical replicates. The samples must have the same
            name, with '#' next to it

    -itst PERCENT_SIZE
            Create an independent test data set, by randomly sampling
            PERCENT_SIZE of the data

    -rrv NUM_SETS PERCENT_SIZE
            Random resample validation, selecting PERCENT_SIZE of the data
            without replacement as a test data set, repeated NUM_SETS times.

DATA_PARAMETERS:
    -itrncsv
            Training file is a csv
    -itstcsv
            Test file is a csv
    -t
            Files are transposed: rows are variables, columns are instances

    -dtr TRAINING_DIRECTORY
            Directory containing training files, one for each test case.
```

```
        Each file contains two columns: variable and value. Within
        the directory files grouped by class folder; e.g. inside the
        training directory, there are two folders: "disease" and
        "control".  There should be no trailing "/" in
        TRAINING_DIRECTORY name.

-tst TEST_FILE
        Specify a test data file

-dtst TEST_DIRECTORY
        Similar to -dtr.

-odri OUTPUT_DIRECTORY
        [Does not work]
        The output directory where to write the result files

-o OUTPUT_DATA_FORMAT
        (arff - not implemented)
            WEKA's attribute relation file format
        csv
            Comma-separated file format
        The default is tab-delimited format.

-rand SEED
        Specifies a seed for creating random folds when running multiple
        runs of cross-validation. SEED is an integer. On Unix-like
        systems and on Windows, a random integer is provided by the
        RANDOM environment variable.

-tpf FILE
        Transpose the input file

-cmbf DIRECTORY
        Combine the files in DIRECTORY. Each file represents one
        training example (such as a mass spectrum), and contains two
        comma-separated columns. The first column contains the names of
        the attributes (such as m/z values), and the second column
        contains the values (intensity values).

-cmbAtts DIRECTORY


-src
        Source data file for learning prior rules for transfer.
        These rules are put on the beam when learning on the target
        training data. See the "-transfer" option.
```

```
    -transferType INDEX_VAL
    -transferMethod INDEX_VAL
    -transfer INDEX_VAL
            The type of transfer for prior rules. Prior rules are handled
            before any learning of new rules on the training data. The
            argument NUMBER can be:
            1   whole rules
                Each prior rule is evaluated on the training data and put on
                the beam. To make sure that the source and training data
                have the same values for each variable, the source data is
                first discretized using the training data discretization
                before learning prior rules on the source data.

            2   rule structure
                Each prior rule is converted to a generalized structure
                where the variable values are removed, leaving only the
                variables in the LHS and the RHS. Then this structure is
                instantiated to create a set of rules with with all
                possible combinations values for the variables.

    -priorRulesSearch
            Specifies that the prior rules should be used in the search. If
            this option is omitted, the rules do not cover any of the data
            for purposes of inductive strengthening.

    -priorRulesSpecialize
    -specializePrior
            Specifies that the prior rules should be specialized. If this
            option is omitted, the prior rules are not specialized on the
            beam.

    TRAINING_FILE The training file is specified as the last argument.

CONVERSION_PARAMETERS
    -c CSV_DATA_FILE
            Convert the csv-delimited file to tab-delimited or vice versa.

    -tpf DATA_FILE
            Transpose the file data file; that is, make the first row be the
            first column, the first column be the first row, second row be
            the second column, etc.

OUTPUT
    The program prints to standard output a log of its working that includes
    the the program parameters (and learning parameters), rule model
    learned, its performance, starting time, total running time.
```

```
    For each rule, the log includes the following statistics: CF, CF/cost,
    p-value, true positive (TP) count, false positive (FP) count, and test
    TP and test FP. The last two statistics represent the number of test
    examples for which the rule was applied correctly (TP) or incorrectly
    (FP) when using the whole model. When applying the model, a rule may not
    fire even if it matches a test example, because of interaction with
    other rules.

FILE FORMATS
    Input files can be comma-separated (CSV), tab-separated. (WEKA's ARFF
    format is not yet supported.) With comma-separated and tab-separated
    formats, the class variable is indicated by placing an "@" in front of
    the name. Example: "@Group".

EXAMPLES
    10-fold cross-validation, EBD (2008), and Simple RL with shallow
    (default) bias space search with 5-fold internal cross validation with a
    random seed (on Unix):
        java -Xmx1300m -jar BRL.jar -lp -rgm 0 -bss --d 0 --f 5 \
        -cfv 10 -d 7 1 -r -dp -rand $RANDOM data.txt

    Discretize the data using EBD, without learning:
        java -jar BRL.jar -ppp -dr 7 1 -dp data.txt

    Discretize the data using EBD without learning; the data are in CSV
    files in directory "datadir" (no trailing "/") that has two
    sub-directories:
        java bayesianrl.BRL -ppp -dr 7 1 -dp -itrncsv -dtr datadir

    Bias space search, and using a test file for testing
    (not cross-validation):
        java -Xmx1300m -jar BRL.jar -lp -rgm 0 -bss --d 0 --f 5 -ppp \
        -d 7 1 -r -dp -tst test.txt train.txt

AUTHORS
    Jonathan Lustgarten
    Philip Ganchev, March 2009 -- November 2010
```

APPENDIX D

TRANSFER LEARNING LITERATURE

An early use of transfer learning in machine learning was demonstrated by [Caruana, 1997], where he improved the performance of an artificial neural network (ANN) in the task of steering a vehicle, by training it on an additional task: recognizing the edge of the road.

Much of the existing experimental work in inductive transfer deals with scenarios where the data is partially shared between learning tasks. For example, [Caruana, 1997] and [Silver and Mercer, 2002] use ANNs that share inputs and hidden representations, but have different output; for example in [Caruana, 1997] (see Figure 11), every data vector contains inputs representing an image of a driver's view of a road, and outputs specify (a) which way the car should be steered and (b) the location of the edge of the road. Similarly, in Abu Mostafa's "learning from hints" [Abu-Mostafa, 1990] the input patterns are the same for all tasks.

Alternatively, tasks may share the same output value but have different input values [Baxter, 2000, Ben-David et al., 2003]. [Baxter, 1995] assumes that training examples are generated independently for each task, and in [De Sa, 1994] the output representations are shared but the inputs come from different modalities. The examples may all belong to both tasks (for example if all input vectors are labeled with both steering direction and location of the edge of the road), or some examples may belong to just one task [Silver and Poirier, 2005]. If all data belong to one task each, this amounts to learning a classifier for one class using a related class, and is called inter-class transfer [Fink and Ullman, 2008].

[Pratt and Jennings, 1996] describe an example of a transfer setting. In training an ANN

to recognize images under a particular set of lighting conditions, it may be possible to use images taken under different conditions to improve the speed of learning or the generalization performance: representational transfer: an ANN trained on the auxiliary images is used as a starting point for learning. functional transfer: the two networks share information during learning. This constraint can provide an important bias. See [Silver and Mercer, 2002].

One approach to inductive transfer is multi-task learning (MTL) – the learning of tasks simultaneously (also called parallel transfer) [Caruana, 1997]. MTL has been demonstrated using various kinds of learners: artificial neural networks (ANNs) trained using back-propagation, nearest neighbor [Caruana, 1997], support vector machine (SVM) [Evgeniou and Pontil, 2004] and logistic regression [Raina et al., 2006].

Figure 11 illustrates the design of MTL ANNs of [Caruana, 1997]. Caruana explores MTL with ANNs trained using back propagation, and also suggests how to do MTL with decision trees. He demonstrates MTL on two image recognition domains and a Pneumonia domain. The pneumonia data contained 14199 positive cases from 78 hospitals and, for most cases, 65 measurements including 35 lab test results which are usually available only after hospitalization. The main learning task was to identify a given fraction of the population least at risk of dying. In all these domains, MTL significantly improved classification accuracy, and [Caruana, 1997] shows that this is due to positive inductive transfer, by ruling out other explanations.

[Caruana and De Sa, 1997] use input variables discarded by variable selection as extra outputs in a ANN, on data from the DNA Splice Junction domain. The best 30 of 180 binary variables are used as inputs and the next 30 as outputs. This produced a statistically significant but marginal improvement in accuracy and cross-entropy, compared to using selected variables as inputs only. [Caruana and De Sa, 1997] showed an improvement with synthetic data.

How to select appropriate auxiliary tasks is largely an open problem, but in some scenarios there are clear candidates, based on domain knowledge. For example, variables available during learning but not when the learned model is applied for prediction, such as measurements done in the future [Caruana, 1997] may be useful auxiliary tasks. In some studies, alternative representations of the main output variable, such as different calibrations, were

useful. [Thrun and O'Sullivan, 1996] use a nearest neighbor (NN) approach selectively transfer learned knowledge from a large number of learning tasks, only a few of which are relevant. Their Task Clustering algorithm groups tasks into similarity clusters. When facing a new task, it uses the the most related task cluster to inform the distance metric for the target task.

Mechanisms of positive transfer in back-propagation ANN MTL include the following: (a) the extra information in the training signals effectively amplifies the data; (b) this improves variable selection in the ANN; (c) difficult tasks eavesdrop on hidden variables learned by easy tasks; and (d) tasks prefer the same representation bias [Caruana, 1997].

Context-sensitive MTL, csMTL [Silver et al., 2008], improves on the multi-output model using additional inputs to represent the task of the examples, as shown in Figure 12. For example, one additional input can take one value for each task. This allows representing task relatedness when two tasks share some but not all of the same examples. It also removes the redundant outputs and representations of the same task, from the original MTL NN model, allowing examples to be added to a task and prior knowledge to be indexed. csMTL was compared by accuracy to MTL and ηMTL (an MTL variant), on data from three domains Band, Logic and fMRI. ηMTL is an MTL variant that selects the most related task knowledge by correlation of the main outputs. csMTL outperformed MTL on the Band and fMRI domains, and equaled ηMTL on all domains. Single-task learning ANNs perform poorly on the main task for each domain due to limited training data, while on the auxiliary tasks, they achieve reasonable accuracies ($> .75$).

MTL can provide some advantages over sequential transfer [Caruana, 1997]: (a) In sequential transfer, the sequence must often be defined manually; (b) If tasks are difficult to learn in isolation, sequential transfer will lack the combined inductive bias; (c) In sequential transfer, earlier tasks cannot benefit from later ones, and therefore cannot help them as much; (d) Sequential transfer needs to balance the retention of learned bias with the learning of new bias; and (e) Sequential transfer is difficult to scale to many tasks.

[Baxter, 2000] casts inductive transfer as the search for a learning bias that is appropriate for all learning tasks in an environment. Each task is represented by a distribution P_θ on the same space of inputs and outputs, $X \times Y$, and a set of m examples is sampled from $X \times Y$

according to P_θ. Tasks are sampled from a set R of distributions according to a distribution Q which controls which tasks a learner is likely to encounter. Given the observed data sets, a bias learner searches for a learning bias appropriate for all the tasks; that is a hypothesis space H in a given family H of hypothesis spaces, which contains hypotheses with maximum average performance on the data sets. H is then likely to contain hypotheses to other tasks drawn from the same environment. Baxter bounds the number of tasks and examples per task required so that, for any H in H, the loss $\mathrm{er}_Q(H)$ is likely to be close to the empirical loss on the data, $\mathrm{er}_z(H)$. Then the sample complexity per task for learning new examples is smaller than when using one task, and the sample complexity for learning a new task is smaller after learning the bias of the environment. In particular, the number m of examples required to accurately estimate the error of a hypothesis depends inversely on the number of tasks. The learning bias being sought is a set of variables that are appropriate for all learning tasks in an environment. The variable set is a mapping from the input space $\mathbb{R}^d$ to a lower dimensional space $\mathbb{R}^k$. [Baxter, 2000] shows that variable sets can be learned using single hidden layer ANN architecture with a sigmoidal squashing function and a variation of gradient descent.

[Baxter, 1998] presents a hierarchical Bayes model [Gelman et al., 2004] of learning to learn, where Q is an "objective prior" for sampling the task distributions P_θ. The learner's bias is represented by the set of candidate distributions, $R = \{P_\theta | \theta \in \Theta\}$, and a "subjective prior" P_π parametrized by $\pi \in \Pi$. The learner's goal is to find the task-sampling distribution Q among the set of candidate prior distributions, $\{P_\pi | \pi \in \Pi\}$, given a hyper-prior P_Π on Π, and assuming Q is among the candidates. This model is used to derive bounds for the sample complexity of the ANN learning of Baxter (2000).

[Niculescu-Mizil and Caruana, 2005] consider learning the structure of a set of Bayesian belief networks for related problems. The learner is given a set of learning problems that have the same variables, but potentially different conditional independences among the variables. Hence these problems need to be encoded using similar but different belief networks. The paper presents an iterative greedy algorithm to optimize a combined objective of having (a) similar networks for all the problems and (b) high observed data likelihood, given a prior preference over network structures. The algorithm starts with the networks for all problems

having the same configuration, and changes an edge between the same two variables for all networks. The change that results in the greatest improvement in the combined objective is taken, and the procedure is repeated. The paper provides results on two synthetic data sets (Alarm and Insurance) and shows that this MTL approach improves on single task learning.

[Ben-David et al., 2002] bound the sample complexity for a number of data sets, each of which represents the same set of examples transformed in an unknown way. In other words, the data for different tasks are sampled from different probability distributions related by the transformations among examples. The learner searches for a hypothesis that minimizes the average error among the data sets. Using a generalized definition of VC dimension, the sample complexity is bounded by the size of hypothesis space used for each data set, and the size of the set of potential transformations among data sets. An important factor is the equivalence class induced on the hypothesis set by the transformation set. [Ben-David et al., 2003] prove a stronger bound for learning focused on one of the tasks, rather than maximizing the average performance.

[Fink et al., 2006] learn a representation for a multi-class problem, where predictive characteristics are shared among different classes. Their framework is essentially an alternative regularization for a multi-class linear classifier. They decompose the linear classifier into a linear change of representation and a linear classifier for the new representation. By regularizing the transformation and classifier separately, they encourage different classes to share input characteristics. For example, they apply their method to identifying the animal in a photograph. For some of the animals, they have a very small number of training examples, but these animals share characteristics such as the presence or absence of horns, length of legs, presence or absence of spots with other animals. By encouraging the new representation to have characteristics shared among problems, they allow the characteristics to be learned by pooling the training data of different classes, while the distinction between classes is based on the intermediate shared representation. They find that for the recognition of written characters and the identification of animals in photographs, their method significantly outperforms standard multi-class classification methods.

These are examples where inductive transfer was beneficial to learning. But inductive transfer may also be detrimental to learning. This may occur if the tasks are not appropri-

ately related, or the transfer mechanism is not appropriate. For example, [Caruana, 1997] cites experiments where MTL degraded performance. In addition, there may be practical problems preventing its use. For example, if tasks are averaged using a weighted average in ANN MTL or decision tree MTL, the base algorithm must be fast enough to run many times to learn appropriate task weights, so it may not be appropriate for large data sets or many variables. Also, if the optimization set is too small, it may be over-fitted.

BIBLIOGRAPHY

[Abu-Mostafa, 1990] Abu-Mostafa, Y. S. (1990). Learning from hints in neural networks. *Journal of Complexity*, 6(2):192–198.

[Anderson and Anderson, 2005] Anderson, N. L. and Anderson, N. G. (2005). Proteome and proteomics: New technologies, new concepts, and new words. *Electrophoresis*, 19(11):1853–1861.

[Baxter, 1995] Baxter, J. (1995). Learning internal representations. In *Proceedings of the eighth annual conference on Computational learning theory*, page 320. ACM.

[Baxter, 1998] Baxter, J. (1998). Theoretical models of learning to learn. In *Learning to learn*. Kluwer Academic Publishers.

[Baxter, 2000] Baxter, J. (2000). A model of inductive bias learning. *JAIR*, 12:149–198.

[Ben-David et al., 2003] Ben-David, S., , and Schuller, R. (2003). Exploiting Task Relatedness for Multiple Task Learning. In *Proceedings of Conference on Learning Theory (COLT)*.

[Ben-David et al., 2002] Ben-David, S., Gehrke, J., and Schuller, R. (2002). A theoretical framework for learning from a pool of disparate data sources. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 449.

[Bigbee et al., 2011] Bigbee, W. L., Gopalakrishnan, V., Weissfeld, J. L., Wilson, D. O., Dacic, S., Siegfried, J. M., and Lokshin, A. E. (2011). A Multiplexed Serum Biomarker Immunoassay Panel Discriminates Clinical Lung Cancer Patients from High-Risk Individuals Found to be Cancer-Free by CT Screening. *Submitted to: Cancer Epidemiology, Biomarkers & Prevention*.

[Blitzer, 2008] Blitzer, J. (2008). *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, University of Pennsylvania.

[Buchanan and Feigenbaum, 1978] Buchanan, B. and Feigenbaum, E. (1978). Dendral and Metadendral: Their Applications Dimensions. *Artificial Intelligence*, 11(1978):5–24.

[Caruana, 1997] Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41–75.

[Caruana and De Sa, 1997] Caruana, R. and De Sa, V. (1997). Promoting poor features to supervisors: Some inputs work better as outputs. In *Advances in Neural Information Processing Systems*. MIT Press.

[Clearwater and Provost, 1990] Clearwater, S. and Provost, F. (1990). RL4: A tool for knowledge-based induction. In *Proceedings of the 2nd International IEEE Conference on Tools for Artificial Intelligence*.

[Cohen, 1995] Cohen, W. (1995). Fast effective rule induction. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 115–123. MORGAN KAUFMANN PUBLISHERS, INC.

[Coombes et al., 2003] Coombes, K., Fritsche Jr, H., Clarke, C., Chen, J., Baggerly, K., Morris, J., Xiao, L., Hung, M., and Kuerer, H. (2003). Quality control and peak finding for proteomics data collected from nipple aspirate fluid by surface-enhanced laser desorption and ionization. *Clinical Chemistry*, 49(10):1615.

[Coombes et al., 2005] Coombes, K., Tsavachidis, S., Morris, J., Baggerly, K., Hung, M., and Kuerer, H. (2005). Improved peak detection and quantification of mass spectrometry data acquired from surface-enhanced laser desorption and ionization by denoising spectra with the undecimated discrete wavelet transform. *Proteomics*, 5(16):4107–4117.

[De Sa, 1994] De Sa, V. (1994). Learning classification with unlabeled data. *Advances in Neural Information Processing Systems*, 6:112–119.

[Ellis, 1965] Ellis, H. (1965). *The transfer of learning*. Macmillan New York.

[Evgeniou and Pontil, 2004] Evgeniou, T. and Pontil, M. (2004). Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117. ACM.

[Fayyad and Irani, 1993] Fayyad, U. M. and Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of Thirteenth Intenrational Joint Conference on Artificial Intelligence (IJCAI-93)*.

[Fink et al., 2006] Fink, M., Shalev-Shwartz, S., Singer, Y., and Ullman, S. (2006). Online multiclass learning by interclass hypothesis sharing. In *Proceedings of the 23rd international conference on Machine learning*, page 320. ACM.

[Fink and Ullman, 2008] Fink, M. and Ullman, S. (2008). From aardvark to zorro: A benchmark for mammal image classification. *International Journal of Computer Vision*, 77(1):143–156.

[Gelman et al., 2004] Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian data analysis*. Chapman & Hall.

[Golub et al., 1999] Golub, T., Slonim, D., Tamayo, P., et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531.

[Gopalakrishnan et al., 2006] Gopalakrishnan, V., Ganchev, P., Ranganathan, S., and Bowser, R. (2006). Rule learning for disease-specific biomarker discovery from clinical proteomic mass spectra. *Data Mining for Biomedical Applications*.

[Hall, 1999] Hall, M. (1999). *Correlation-based feature selection for machine learning*. PhD thesis, Universit of Waikato.

[Kolli et al., 2009] Kolli, V. S. K., Seth, B., Weaver, L., Lustgarten, J. L., Gopalakrishnan, V., Malehorn, D., Bigbee, B., Mural, R. J., and Shriver, C. D. (2009). Breast Cancer Sera for Pattern Analysis. In *Proceedings of the Humamn Protein Organisation Conference 2009*. Human Protein Organisation.

[Liu et al., 2002] Liu, H., Li, J., and Wong, L. (2002). A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *GENOME INFORMATICS SERIES*, 13:51–60.

[Liu and Setiono, 1995] Liu, H. and Setiono, R. (1995). Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of the Seventh International Conference on Tools with Artificial Intelligence*, page 88. Citeseer.

[Lustgarten, 2009] Lustgarten, J. (2009). *A Bayesian Rule Generation Framework for 'Omic' Biomedical Data Analysis*. PhD thesis, University of Pittsburgh.

[Mitchell, 1980] Mitchell, T. (1980). The need for biases in learning generalizations. *Readings in machine learning*, pages 184–191.

[Niculescu-Mizil and Caruana, 2005] Niculescu-Mizil, A. and Caruana, R. (2005). Learning the structure of related tasks. In *Proceedings of NIPS-2005 Workshop on Inductive Transfer: 10 Years Later*. Citeseer.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge And Data Engineering*.

[Pelikan et al., 2007] Pelikan, R., Bigbee, W., Malehorn, D., Lyons-Weiler, J., and Hauskrecht, M. (2007). Intersession reproducibility of mass spectrometry profiles and its effect on accuracy of multivariate classification models. *Bioinformatics*, 23(22):3065.

[Pratt and Jennings, 1996] Pratt, L. and Jennings, B. (1996). A survey of transfer between connectionist networks. *Connection Science*, 8(2):163–184.

[Provost et al., 1999] Provost, F., Aronis, J., and Buchanan, B. (1999). Rule-space search for knowledge-based discovery. Technical Report IS 99-012, Stern School of Business, New York University.

[Quinlan, 1993] Quinlan, J. (1993). *C4. 5: programs for machine learning*. Morgan Kaufmann.

[Raina et al., 2006] Raina, R., Ng, A., and Koller, D. (2006). Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on machine learning*, page 720.

[Ranganathan, 2004] Ranganathan, S. (2004). *Amyotrophic lateral sclerosis molecular mechanisms to diagnostics*. PhD thesis, University of Pittsburgh.

[Ranganathan et al., 2005] Ranganathan, S., Williams, E., Ganchev, P., et al. (2005). Proteomic profiling of cerebrospinal fluid identifies biomarkers for amyotrophic lateral sclerosis. *Journal of neurochemistry*, 95(5):1461–1471.

[Reid, 2007] Reid, M. (2007). *DEFT guessing: Using inductive transfer to improve rule evaluation from limited data*. PhD thesis, University of New South Wales.

[Rifai et al., 2006] Rifai, N., Gillette, M., and Carr, S. (2006). Protein biomarker discovery and validation: the long and uncertain path to clinical utility. *Nature biotechnology*, 24(8):971–983.

[Robnik-Šikonja and Kononenko, 2003] Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning*, 53(1):23–69.

[Ryberg et al., 2010] Ryberg, H., An, J., Darko, S., Lustgarten, J., Jaffa, M., Gopalakrishnan, V., Lacomis, D., Cudkowicz, M., and Bowser, R. (2010). Discovery and verification of amyotrophic lateral sclerosis biomarkers by proteomics. *Muscle and Nerve*, 42(1):104–111.

[Sem, 2007] Sem, D. (2007). *Spectral techniques in proteomics*. CRC.

[Semmes et al., 2005] Semmes, O., Feng, Z., Adam, B., Banez, L., Bigbee, W., Campos, D., Cazares, L., Chan, D., Grizzle, W., Izbicka, E., et al. (2005). Evaluation of serum protein profiling by surface-enhanced laser desorption/ionization time-of-flight mass spectrometry for the detection of prostate cancer: I. Assessment of platform reproducibility. *Clinical chemistry*, 51(1):102.

[Shortliffe Bruce and Edward, 1975] Shortliffe Bruce, G. and Edward, H. (1975). A model of inexact reasoning in medicine. *Mathematical Biosciences*, 23(3-4):351–379.

[Silver and Mercer, 2002] Silver, D. and Mercer, R. (2002). The task rehearsal method of life-long learning: Overcoming impoverished data. *Advances in Artificial Intelligence*, pages 90–101.

[Silver and Poirier, 2005] Silver, D. and Poirier, R. (2005). Requirements for machine lifelong learning. Technical Report TR-2005-009, Jodrey School of Computer Science.

[Silver et al., 2008] Silver, D., Poirier, R., and Currie, D. (2008). Inductive transfer with context-sensitive neural networks. *Machine learning*, 73(3):313–336.

[Thrun, 1996] Thrun, S. (1996). Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems*, pages 640–646.

[Thrun and O'Sullivan, 1996] Thrun, S. and O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The TC algorithm. In *Machine learning International workshop and conference*, pages 489–497. Citeseer.

[Utgoff, 1984] Utgoff, P. (1984). *Shift of bias for inductive concept learning*. PhD thesis, Rutgers University, New Brunswick, NJ.

[Vilalta and Drissi, 2002] Vilalta, R. and Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95.

[Wong et al., 2005] Wong, J., Cagney, G., and Cartwright, H. (2005). SpecAlign–processing and alignment of mass spectra datasets. *Bioinformatics*, 21(9):2088.

[Xue et al., 2007] Xue, Y., Liao, X., Carin, L., and Krishnapuram, B. (2007). Multi-task learning for classification with Dirichlet process priors. *The Journal of Machine Learning Research*, 8:63.

[Yildiz et al., 2007] Yildiz, P., Shyr, Y., Rahman, J., et al. (2007). Diagnostic accuracy of MALDI mass spectrometric analysis of unfractionated serum in lung cancer. *Journal of Thoracic Oncology*, 2(10):893.

[Yutaka et al., 1900] Yutaka, Y., Dale, M., Bao-Ling, A., Marcy, W., Mark, T., and Ziding, F. (1900). An automated peak identification/calibration procedure for high-dimensional protein measures from mass spectrometers. *Journal of Biomedicine and Biotechnology*, 2003(4):242–248.