

**VIEW LEARNING: A STATISTICAL RELATIONAL APPROACH  
TO MINING BIOMEDICAL DATABASES**

by

Jesse Jon Davis

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

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2007

© Copyright by Jesse Jon Davis 2007

All Rights Reserved

To my parents, Jill and James Davis, and my grandparents, Jane and William Leventhal.

## ACKNOWLEDGMENTS

Many people played instrumental roles in helping me complete this dissertation. My advisor, David Page, created a mentoring environment that fostered my growth as a researcher. He mastered the difficult task of giving me advice and direction, while allowing me ample freedom and time to pursue and develop my ideas. He provided perspective and insight into all of my projects. I am honored to count David among my friends. Elizabeth Burnside served as co-advisor. In the process, she was a patient teacher of all things medical related to mammography and as such, was instrumental in facilitating my ability to develop a computer driven method to analyze them.

I also want to thank the other members of thesis committee, Jude Shavlik, Mark Craven and Jerry Zhu as well as Raghu Ramakrishnan who served my preliminary exam committee. I had the privilege to collaborate with Jude on several projects. I have greatly benefited from his precise and exacting feedback on how to structure and organize many talks and papers. Furthermore, his probing questions during talks and meeting have greatly influenced my research, pushing me to clarify ambiguities and offering alternative paths to examine. Mark and Jerry have supplied valuable critiques of my dissertation as well as numerous practice talks. As a collaborator, Raghu taught me a great deal about databases and he provided invaluable advice on multiple papers and presentations.

I take particular pleasure in thanking Vitor Santos Costa, a brilliant researcher, with an astonishing breadth of knowledge. He provided steady input and guidance as well as assistance in formulating my ideas. He has contributed to every project that I have worked on. I am deeply indebted to him for all the support and assistance he has provided me with over the years. Inês Dutra also played a crucial role in my intellectual development and provided a significant amount of support during the beginning of my graduate career. Rich Maclin has been an invaluable asset

throughout the last five years. He has provided advice to me on almost every aspect of graduate school. I also want to thank Andrea Danyluk, whose undergraduate class in artificial intelligence sparked my interest in machine learning. She was my original inspiration to follow this career path.

I have also had the pleasure of working with several students and post-docs while in Madison: Mark Goadrich, Soumya Ray, Jan Strufy, Irene Ong and Eric Lantz. It was an energizing learning experience to investigate precision-recall space with Mark. He also provided me with an excellent sounding board for my ILP related ideas. Soumya taught me a great deal about how to compose and structure a research paper. Jan and Irene put in many long hours of work with me on a variety of different projects.

The other students in the machine learning group have provided valuable assistance during my graduate school career. I have had many excellent discussions with Lisa Torrey, Mike Molla, Trevor Walker, Keith Noto, Louis Oliphant, Sean McIlwain and Frank Dimaio. Lisa has also edited many of my papers, which greatly improved their presentation. Mike has been my main mlgroup tea, coffee and lunch buddy. I also want to thank Allison Holloway, my girlfriend, as well as my other friends from graduate school. Additionally, my college friends, Jason Chapman, Eli Groban, Michael Gross, Matt Haldeman, Travis Hobart, Chris Kelly, Nishant Nayyar, Adam Sischy, Louis Tabloda, John Thomison, Joe Urwitz and Chris Zerwas, have provided many welcome diversions throughout the process.

I also want to express my gratitude to following funding sources: U.S. Air Force (F30602-01-2-0571) and the National Library of Medicine (NLM 5T15LM007359).

Finally, I need to thank my parents. I have had the unique advantage of attending graduate school in the same city where they live and they have helped me through every step of this process. They have maintained an unshakeable belief in my intellectual curiosity and my ability to overcome hurdles in reaching this goal. I send a thank you to my sisters who flew to Madison to attend my graduation to celebrate with me.

**DISCARD THIS PAGE**

# TABLE OF CONTENTS

|  | Page |
|--|------|
| <b>LIST OF TABLES</b> . . . . .  | vii  |
| <b>LIST OF FIGURES</b> . . . . .   | viii |
| <b>LIST OF ALGORITHMS</b> . . . . .  | xi   |
| <b>NOMENCLATURE</b> . . . . .  | xii  |
| <b>ABSTRACT</b> . . . . .  | xiv  |
| <b>1 Introduction</b> . . . . .  | 1    |
| 1.1 Contributions . . . . .  | 3    |
| 1.2 Thesis Statement . . . . .   | 5    |
| <b>2 Background</b> . . . . .  | 6    |
| 2.1 Bayesian Networks . . . . .  | 6    |
| 2.2 Inductive Logic Programming . . . . .  | 10   |
| 2.3 Statistical Relational Learning . . . . .  | 11   |
| 2.4 Review of ROC and Precision-Recall Curves . . . . .  | 14   |
| <b>3 Inductive Logic Programming for Feature Construction:<br/>A Case Study in Alias Detection</b> . . . . . | 16   |
| 3.1 Alias Detection and the Challenges It Presents . . . . .   | 17   |
| 3.2 ILP For Alias Detection . . . . .  | 18   |
| 3.3 Rule Combination Techniques . . . . .  | 19   |
| 3.4 Empirical Evaluation . . . . .   | 20   |
| 3.5 Related Work . . . . .   | 31   |
| 3.6 Chapter Summary . . . . .  | 32   |

|   | Page |
|---|------|
| <b>4 Multi-Step View Learning Framework</b> . . . . .   | 34   |
| 4.1 Mammography . . . . .   | 35   |
| 4.2 Learning Hierarchy . . . . .  | 37   |
| 4.3 Multi-Step View Learning . . . . .  | 39   |
| 4.4 Data . . . . .  | 41   |
| 4.5 Experiments . . . . .   | 42   |
| 4.6 Approach for Each Level of Learning . . . . .   | 42   |
| 4.7 Methodology . . . . .   | 46   |
| 4.8 Results . . . . .   | 46   |
| 4.9 Interesting Views . . . . .   | 49   |
| 4.10 Related Work . . . . .   | 51   |
| 4.11 Chapter Summary . . . . .  | 52   |
| <b>5 SAYU: A Framework for Integrated View Invention and Model Construction</b> . . . . .     | 53   |
| 5.1 The SAYU Framework . . . . .  | 54   |
| 5.2 SAYU with Bayesian Network Classifiers . . . . .  | 54   |
| 5.3 Datasets . . . . .  | 57   |
| 5.4 Experimental Setup and Results . . . . .  | 58   |
| 5.5 SAYU-View: Incorporating Initial Feature Sets . . . . .                                   | 66   |
| 5.6 Further Experiments and Results . . . . .   | 67   |
| 5.7 Related Work . . . . .  | 70   |
| 5.8 Chapter Summary . . . . .   | 70   |
| <b>6 Combing SAYU and Multiple-Instance Regression for Drug Activity Prediction</b> . . . . . | 72   |
| 6.1 Background and Related Work . . . . .   | 72   |
| 6.2 The MIR-SAYU Algorithm . . . . .  | 78   |
| 6.2.1 Scoring Candidate Rules with SAYU . . . . .   | 78   |
| 6.2.2 Predicting Activity with MI Regression . . . . .  | 82   |
| 6.3 Empirical Evaluation . . . . .  | 85   |
| 6.4 Chapter Summary . . . . .   | 89   |
| <b>7 Learning New Tables and Higher-Arity Predicates</b> . . . . .                            | 91   |
| 7.1 Learning New Predicates . . . . .   | 92   |
| 7.1.1 Scoring Higher-Arity Predicates . . . . .   | 93   |
| 7.1.2 Linkages . . . . .  | 94   |
| 7.1.3 Predicate Learning Algorithm . . . . .  | 96   |

## Appendix

|  | Page       |
|--|------------|
| 7.2 Data and Methodology . . . . .                                 | 99         |
| 7.3 Experiments and Results . . . . .                              | 99         |
| 7.3.1 Discussion of Results . . . . .                              | 101        |
| 7.3.2 Further Investigation of SAYU-VISTA . . . . .                | 106        |
| 7.4 Related Work . . . . .   | 107        |
| 7.5 Chapter Summary . . . . .                                      | 110        |
| <b>8 The Relationship between PR Space and ROC Space . . . . .</b> | <b>113</b> |
| 8.1 Why We Use PR Curves . . . . .                                 | 113        |
| 8.2 Relationship between ROC Space and PR Space . . . . .          | 115        |
| 8.3 Interpolation and AUC . . . . .                                | 121        |
| 8.4 Optimizing Area Under the Curve . . . . .                      | 123        |
| 8.5 Chapter Summary . . . . .                                      | 124        |
| <b>9 Conclusions and Future Work . . . . .</b>                     | <b>126</b> |
| 9.1 Summary . . . . .  | 126        |
| 9.2 Future Work . . . . .  | 129        |

**DISCARD THIS PAGE**

## LIST OF TABLES

| Table | Page  |     |
|-------|---|-----|
| 3.1   | Parameter Settings for EAGLE Simulator Version 1 . . . . .                          | 25  |
| 3.2   | Average AUC-PR for Each Task from the First Version of the EAGLE Simulator . . .    | 25  |
| 3.3   | Parameter Settings for EAGLE Simulator Version 2 . . . . .                          | 26  |
| 3.4   | Average AUC-PR for Each Task from the Second Version of the EAGLE Simulator . .     | 30  |
| 4.1   | Summary of Results for Different Structure Learners on the Mammography Dataset .    | 49  |
| 4.2   | Density of Benign vs Malignant Masses . . . . .                                     | 51  |
| 5.1   | Summary of SAYU and Aleph Results . . . . .   | 60  |
| 5.2   | Summary of $p$ -values for SAYU-TAN vs. Other Algorithms . . . . .                  | 60  |
| 5.3   | Characteristics of Clauses for SAYU and Aleph on the Mammography Dataset . . . .    | 60  |
| 5.4   | Characteristics of Clauses for SAYU and Aleph on the Yeast Protein Dataset . . . .  | 60  |
| 5.5   | Characteristics of Clauses for SAYU and Aleph on the Carcinogenesis Dataset . . . . | 60  |
| 5.6   | Characteristics of Clauses for SAYU and Aleph on the UW-CSE Dataset . . . . .       | 63  |
| 6.1   | Sample Pharmacophore Learned by Aleph . . . . .                                     | 75  |
| 6.2   | Root Mean Squared Errors for Drug Activity Prediction Datasets . . . . .            | 86  |
| 7.1   | Average AUC-PR for Recall $\geq 0.5$ for Each SAYU-VISTA Task . . . . .             | 101 |
| 8.1   | Correct PR Space Interpolation . . . . .  | 122 |

**DISCARD THIS PAGE**

## LIST OF FIGURES

| Figure  | Page |
|---|------|
| 1.1 An Example Dataset for Diagnosing Breast Cancer from a Mammography Report . . . | 2    |
| 2.1 A Sample Bayesian Network . . . . .   | 6    |
| 2.2 A Sample Naïve Bayes Network . . . . .  | 8    |
| 2.3 A Sample TAN Bayes Network . . . . .  | 9    |
| 2.4 A Sample Relational Skeleton for a PRM . . . . .                                | 13   |
| 2.5 Common Machine Learning Evaluation Metrics . . . . .                            | 14   |
| 3.1 Illustration of a Decision List . . . . .                                       | 19   |
| 3.2 Illustration of Unweighted Voting . . . . .                                     | 20   |
| 3.3 Precision-Recall Curve for EAGLE Dataset 1 . . . . .                            | 22   |
| 3.4 Precision-Recall Curve for EAGLE Dataset 2 . . . . .                            | 23   |
| 3.5 Precision-Recall Curve for EAGLE Dataset 3 . . . . .                            | 23   |
| 3.6 Precision-Recall Curve for EAGLE Dataset 4 . . . . .                            | 24   |
| 3.7 Precision-Recall Curve for EAGLE Dataset 5 . . . . .                            | 24   |
| 3.8 Precision-Recall Curve for EAGLE Dataset 6 . . . . .                            | 25   |
| 3.9 Precision-Recall Curve for EAGLE Dataset I . . . . .                            | 27   |
| 3.10 Precision-Recall Curve for EAGLE Dataset II . . . . .                          | 27   |
| 3.11 Precision-Recall Curve for EAGLE Dataset III . . . . .                         | 28   |

| Figure   | Page |
|--|------|
| 3.12 Precision-Recall Curve for EAGLE Dataset IV . . . . .   | 29   |
| 3.13 Precision-Recall Curve for EAGLE Dataset V . . . . .  | 29   |
| 4.1 Expert Bayesian Network for Mammography . . . . .  | 35   |
| 4.2 Abbreviated National Mammography Database Schema . . . . .   | 36   |
| 4.3 Hierarchy of Types of Learning for Mammography Domain . . . . .  | 38   |
| 4.4 Multi-Step Approach to View Learning . . . . .   | 40   |
| 4.5 A Possible View: Size of a Mass Increases Over Time . . . . .  | 41   |
| 4.6 The National Mammography Database Schema Extended with an Additional Field . . . . .                   | 41   |
| 4.7 Aggregation Example for Mammography Domain . . . . .   | 43   |
| 4.8 ROC Curves for Parameter Learning vs. Structure Learning on Mammography Dataset . . . . .              | 47   |
| 4.9 Precision-Recall Curves for Parameter Learning vs. Structure Learning on Mammography Dataset . . . . . | 48   |
| 4.10 Precision-Recall Curves for All Algorithms on Mammography Dataset . . . . .                           | 50   |
| 4.11 Average AUC-PR for Recalls $\geq 0.5$ for Mammography Dataset . . . . .                               | 50   |
| 5.1 General SAYU Framework . . . . .   | 54   |
| 5.2 Several Moves in SAYU's Learning Procedure . . . . .   | 55   |
| 5.3 Precision-Recall Curves for SAYU and Aleph on the Mammography Dataset . . . . .                        | 59   |
| 5.4 Precision-Recall Curves for SAYU and Aleph on the Yeast Protein Dataset . . . . .                      | 61   |
| 5.5 Precision-Recall Curves for SAYU and Aleph on the Carcinogenesis Dataset . . . . .                     | 61   |
| 5.6 Precision-Recall Curves for SAYU and Aleph on the UW-CSE Dataset . . . . .                             | 62   |
| 5.7 ROC Curves for SAYU and Aleph on the Carcinogenesis Dataset . . . . .                                  | 62   |
| 5.8 Precision-Recall Curves for Each Level of Learning . . . . .   | 68   |

| Appendix<br>Figure   | Page |
|--|------|
| 5.9 Average Area Under the Curve For Recalls $\geq 0.5$ on Mammography Dataset . . . . .             | 69   |
| 6.1 Sample Pharmacophore for ACE Inhibitor . . . . .   | 73   |
| 6.2 Contrasting Supervised and Multiple-Instance Learning . . . . .                                  | 74   |
| 7.1 Scoring a Predicate with Count Aggregation . . . . .   | 94   |
| 7.2 Result of Accepting a Predicate Scored with Count Aggregation . . . . .                          | 95   |
| 7.3 Scoring a Predicate Through Linkage . . . . .  | 96   |
| 7.4 Results of Accepting a Predicate Scored with a Linkage . . . . .                                 | 97   |
| 7.5 Precision-Recall Curves Comparing SAYU-VISTA, MLNs and SAYU on the Cora Dataset . . . . .        | 101  |
| 7.6 Precision-Recall Curves Comparing SAYU-VISTA, MLNs, and SAYU on the UW-CSE Dataset . . . . .     | 102  |
| 7.7 Precision-Recall Curves Comparing SAYU-VISTA, MLNs and SAYU on the Mammography Dataset . . . . . | 102  |
| 8.1 The Difference between Comparing Algorithms in ROC vs PR Space . . . . .                         | 114  |
| 8.2 Two Cases for Claim 1 of Theorem 8.2.2 . . . . .   | 116  |
| 8.3 Two Cases of Claim 2 of Theorem 8.2.2 . . . . .  | 118  |
| 8.4 Sample Convex Hull in ROC Space and Achievable Curve in PR Space . . . . .                       | 119  |
| 8.5 The Effect of Incorrect Interpolation in PR Space . . . . .                                      | 123  |
| 8.6 Difference in Optimizing Area Under the Curve in Each Space . . . . .                            | 123  |

**DISCARD THIS PAGE**

## LIST OF ALGORITHMS

| Algorithm                        | Page |
|----------------------------------|------|
| 1 SAYU Algorithm . . . . .       | 56   |
| 2 SAYU-View Algorithm . . . . .  | 67   |
| 3 MIR-SAYU Algorithm . . . . .   | 81   |
| 4 SAYU-VISTA Algorithm . . . . . | 98   |

**DISCARD THIS PAGE**

## NOMENCLATURE

|            |   |
|------------|---|
| AUC-PR     | area under the PR curve                                     |
| AUC-ROC    | area under the ROC curve                                    |
| FN         | false negative  |
| FP         | false positive  |
| FPR        | false positive rate   |
| ILP        | inductive logic programming                                 |
| MI         | multiple-instance   |
| MIR        | multiple-instance regression                                |
| MLN        | Markov logic network  |
| NMD        | National Mammography Database                               |
| PR         | precision-recall  |
| PRM        | probabilistic relational model                              |
| ROC        | receiver operating characteristic                           |
| SAYU       | score as you use  |
| SAYU-VISTA | score as you use with view invention through scoring tables |
| SRL        | statistical relational learning                             |

|     |                                    |
|-----|------------------------------------|
| TAN | tree-augmented naïve Bayes network |
| TN  | true negative                      |
| TP  | true positive                      |
| TPR | true positive rate                 |

## ABSTRACT

This dissertation develops and evaluates statistical relational learning (SRL) algorithms that can automatically alter the schema of a database by learning new field and table definitions. The algorithmic advances made in this dissertation are motivated by important problems such as providing decision support system for radiologists who read mammograms, predicting three-dimensional Quantitative Structure-Activity Relationships for drug design, and performing entity resolution—the task of recognizing when two molecules, patients, biological pathways, etc. are actually the same.

This dissertation introduces view learning, the ability to automatically alter the schema of a database through the addition of new fields or tables, for SRL, and presents two algorithms for augmenting the schema of a database by adding new fields. It then extends view learning by developing an algorithm for performing predicate invention. We demonstrate the utility of view learning for SRL in two ways. First, it learns significantly more accurate models on a wide variety of domains. Second, it uncovers important and useful knowledge in these domains. For example, it identified a novel feature from a mammography report that is indicative of malignancy.

Motivated by the preceding work, the last part of the dissertation investigates the relationship between receiver operator characteristic (ROC) space and precision-recall (PR) space. Among other contributions, this part proves that for a fixed number of positive and negative examples, one curve dominates another curve in ROC space if and only if the first curve dominates the second curve in PR space. This result implies the existence of an analog to the convex hull for PR space, which we call the achievable PR curve, and it provides an efficient algorithm for constructing the achievable curve.

# Chapter 1

## Introduction

Applications from other fields often drive advances in machine learning and data mining. Biology and medicine have served as a ready source of novel, interesting, challenging and important problems to drive innovation in machine learning and data mining. The advent and prevalence of high-throughput techniques, such as gene-expression microarrays, single-nucleotide polymorphism (SNP) chips and high-throughput screening of molecules for biological activity, have greatly increased the quantity of biological data available. Furthermore, these technologies have allowed people to collect data for the same phenomena from multiple sources. The most natural place to store this burgeoning collection of diverse data is in relational databases with multiple tables, one for each data type.

Biomedical data presents many challenges for data mining and machine learning research. Complex structured data, such as patient clinical histories or the structures of potential drug molecules, represent one obstacle. Such data are most naturally stored in a relational database with multiple relational tables, whereas most machine learning and data-analysis algorithms assume the data are stored in a single table. Additionally, even when data reside in a single table, data in different rows of the table may be related, rather than independent as assumed by most data analysis and machine learning algorithms.

For a concrete example, consider the problem of breast cancer diagnosis. Significant data about patients can be captured in a table, such as Figure 1.1, that contains information about abnormalities in patients from mammography reports. Each row in Figure 1.1 represents an abnormality on a mammogram. Bayesian networks are probabilistic graphical models that have been applied to the task of breast cancer diagnosis from mammography data. However, when attempting to

label a given abnormality as benign or malignant, a Bayesian network can only incorporate information about that specific abnormality. However, a radiologist might include data about other abnormalities on the same mammogram or prior abnormalities for the same patient (e.g. patient P1 in Figure 1.1) in the decision process.

Another complexity is entity-resolution—the need to recognize when two molecules, patients, biological pathways, etc. are actually the same. This research focuses on the challenges raised by concrete, significant tasks in the analysis of biological and medical data. Furthermore, these issues exist in many other data types; for example, entity resolution also appears as alias detection within intelligence analysis (Davis et al., 2005c), record de-duplication within databases and citation matching within document analysis (Davis et al., 2007c).

| Abnormality | Patient | Date | Mass Shape | ... | Mass Size | Location | Benign<br>Malignant |
|-------------|---------|------|------------|-----|-----------|----------|---------------------|
| 1           | P1      | 5/02 | Oval       |     | 2mm       | RU4      | Benign              |
| 2           | P2      | 5/04 | Round      |     | 4mm       | RU4      | Malignant           |
| 3           | P3      | 5/04 | Oval       |     | 1mm       | LL3      | Benign              |
| ...         | ...     | ...  | ...        |     | ...       | ...      | ...                 |

Figure 1.1 An Example Dataset for Diagnosing Breast Cancer from a Mammography Report

The goal of this dissertation is to upgrade standard learning algorithms to handle such interdependencies, which are common in biomedical domains. This work falls in the area of statistical relational learning (SRL). SRL advances beyond Bayesian network learning and related techniques by handling domains with multiple tables, by representing relationships between different rows of the same table, and by integrating data from several distinct databases. These algorithms are particularly adept at integrating multiple data sources into one cohesive model. Currently, SRL techniques can learn joint probability distributions over the fields of a relational database with multiple tables. Nevertheless, SRL techniques are constrained to use only the tables and fields already in the

database, without modification. In contrast, many human users of relational databases find it beneficial to define alternative *views* of a database—further fields or tables that can be computed from existing ones. For example, a human user might define a view stating that an increase in the size of the abnormality at a location since an earlier mammogram may be indicative of a malignancy.

This dissertation will present a series of algorithms, which augment SRL by adding the ability to learn new fields and tables. In database terminology, these new fields and tables constitute a learned *view* of the database. The end result is a state-of-the-art SRL framework that automatically defines new views of the data. The framework is broadly applicable to clinical and biological problems and this dissertation shows that it increases the accuracy of the learned models.

## 1.1 Contributions

Two common themes connect most of the work done for this dissertation. The first is an emphasis on biological and medical applications. The second is the use of inductive logic programming (ILP) techniques to learn view definitions. The contributions made in this thesis can be roughly divided into four parts. The first part evaluates different methods for converting hypotheses of rules into a classifier. The second part introduces view learning for SRL and presents two algorithms for augmenting the schema of a database by adding new fields. The third part extends view learning by developing an algorithm for performing predicate invention. Motivated by the preceding work, the fourth part investigates the relationship between receiver operator characteristic (ROC) space and precision-recall (PR) space.

The first section, consisting of Chapter 3, evaluates different approaches for converting an ILP learned hypothesis into a classifier.

- We propose using ILP algorithms for determining identity equivalence, otherwise known as alias detection, in intelligence analysis. A central advantage of using ILP for this task is that it produces understandable results.
- We demonstrate that the traditional ILP approach of constructing a decision list to convert the learned hypothesis into a classifier maximizes the number of false positives.

- We present a simple approach to decrease the number of false positives by representing each learned rule as an attribute in a Bayesian network. Empirically, this method yields significantly more accurate models than using unweighted voting, which is a simple extension of the decision list approach.

The second section, outlined in Chapters 4, 5 and 6, presents algorithms for learning views that augment a database schema through the introduction of new fields.

- We introduce the problem of classifying an abnormality on a mammogram as benign or malignant as an interesting problem for SRL. We use this application to illustrate a shortcoming of current SRL algorithms: their inability to alter the schema database. We propose view learning to address this weakness.
- We propose a multi-step framework for learning views. The first step involves using an ILP algorithm to construct a large set of relational features. The second step selects which relational features to include in the model. The final step builds a statistical model.
- We propose a method for generating views, based on the idea of *constructing the classifier as we learn the rules* (Davis et al., 2005a). This methodology, called *Score As You Use* or *SAYU*, interleaves feature invention, feature selection and model construction.
- We empirically demonstrate that SAYU leads to significantly more accurate models than the multi-step methodology for classification tasks.
- We present the MIR-SAYU system for drug activity prediction. We evaluate this algorithm on real-world datasets and demonstrate that MIR-SAYU results in more accurate predictions than a current state-of-the-art algorithm for this task. We also find that MIR-SAYU discovers biologically relevant features.

The third section, consisting of Chapter 7, extends view learning by adding the capability to invent *new relational tables*.

- We present the SAYU-VISTA algorithm which adds two important extensions. First, it learns predicates that capture many-to-many relations and require a new table to represent. Second, it constructs predicates that operate on different types than the target concept.
- We empirically demonstrate this leads to more accurate models than the SAYU algorithm. Furthermore, we demonstrate that it achieves better performance than Markov Logic Networks (MLNs), another leading SRL framework.

Finally, working on problems that contain a large skew in the class distribution led us to use precision-recall (PR) curves as an evaluation metric. Thinking about PR curves led to some theoretical insights about their relationship to receiver operator character (ROC) curves. This chapter makes four substantial contributions:

- A proof that for a fixed number of positive and negative examples, one curve dominates another curve in ROC space if and only if the first curve dominates the second curve in PR space.
- There exists an analog to the convex hull for PR space, which we call the achievable PR curve. We give an efficient algorithm for constructing the achievable curve.
- A justification for the fact that linearly interpolating between points in PR space is not achievable, in general. An algorithm is given for performing the correct interpolation between points in PR space.
- Optimizing the area under the curve (AUC) in one space does not optimize the AUC in the other space.

## 1.2 Thesis Statement

We propose and evaluate statistical relational learning algorithms that define new views of data. We hypothesize that statistical relational learning algorithms can benefit from the ability to define new views of a database. Learning algorithms that alter the schema of the database by adding new fields and tables will result in more accurate models.

## Chapter 2

### Background

This dissertation draws on three areas of machine learning: probabilistic models (namely Bayesian networks), inductive logic programming (ILP) and statistical relational learning. We will give background information about these areas that will be used in the rest of the dissertation. At the end of the section, we will give a brief overview of evaluation methods for machine learning.

#### 2.1 Bayesian Networks

Bayesian networks (Pearl, 1988) are probabilistic graphical models that encode a joint probability distribution over a set of random variables. A Bayesian network compactly represents the joint probability distribution over a set of random variables by exploiting conditional independencies between variables (attributes). Figure 2.1 shows a simple Bayesian network.

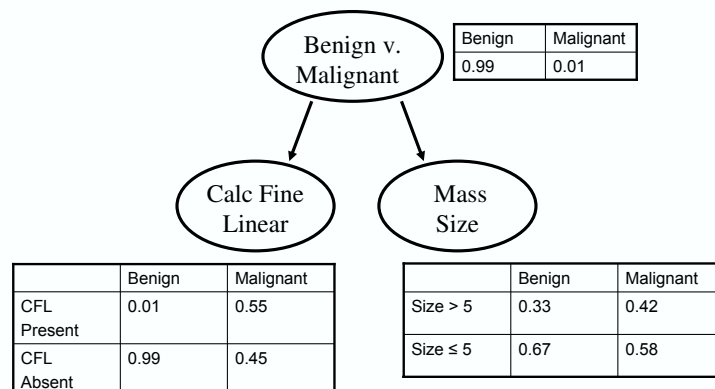


Figure 2.1 A Sample Bayesian Network

We will use uppercase letters (e.g.  $A$ ) to refer to a random variable and lower case letters (e.g.  $a$ ) to refer to a specific value for that random variable. Given a set of random variables  $X = \{X_1, \dots, X_n\}$ , a Bayesian network  $B = \langle G, \Theta \rangle$  is defined as follows.  $G$  is a directed, acyclic graph that contains a node for each variable  $X_i \in X$ . For each variable (node) in the graph, the Bayesian network has a conditional probability table  $\theta_{X_i|Parents(X_i)}$  giving the probability distribution over the values that variable can take for each possible setting of its parents, and  $\Theta = \{\theta_{X_1}, \dots, \theta_{X_n}\}$ . A Bayesian network  $B$  encodes the following probability distribution:

$$P_B(X_1, \dots, X_n) = \prod_{i=1}^{i=n} P(X_i|Parents(X_i)) \quad (2.1)$$

Two learning problems exist for Bayesian networks. The first learning task involves learning the parameters  $\Theta$ .

- Given: Dataset  $D$  (contains variables  $X_1, \dots, X_n$ ), Network structure  $G$
- Learn:  $\theta_{X_i|Parents(X_i)}$  for each node in the network

One common approach to learning parameters is computing maximum likelihood estimates (Friedman et al., 1997). The ELR algorithm (Greiner & Zhou, 2002) provides a mechanism for discriminative training of parameters. Another approach, used in this dissertation, is to use a prior probability in conjunction with the maximum likelihood estimate. This is also known as an  $m$ -estimate (Mitchell, 1997). Given a dataset  $D$ ,  $P(X = x)$  is given by the following formula:

$$P(X = x) = \frac{\hat{x} + m \times p_x}{n + m} \quad (2.2)$$

In this equation:

- $\hat{x}$  is the number of times that  $X = x$  in  $D$ .
- $p_x$  is the prior probability of  $X = x$ .
- $m$  is the term allows us to weight the relative importance of the prior distribution versus the empirical counts.

One common approach to setting  $p_x$  and  $m$  is known as the Laplace correction. This sets  $p_x = 1/k$  and  $m = k$ , where  $k$  equals the number of distinct settings for  $X$ .

The second learning task subsumes the first task, and involves learning the parameters  $\Theta$  as well as the network structure  $G$ .

- Given: Dataset  $D$  (contains variables  $X_1, \dots, X_n$ )
- Learn: Network structure  $G$  and  $\theta_{X_i|Parents(X_i)}$  for each node in the network

Popular structure learning algorithms include K2 (Cooper & Herskovits, 1992), BNC (Grossman & Domingos, 2004), tree augmented naïve Bayes (Friedman et al., 1997) and the Sparse Candidate algorithm (Friedman et al., 1999b).

This dissertation uses existing techniques for constructing Bayesian networks for classification. This work primarily makes use of two types of models: naïve Bayes and tree augmented naïve Bayes (TAN). For this discussion, assume we have a set of attributes  $A_1, \dots, A_n$ , a class variable  $C$  and a dataset  $D$ .

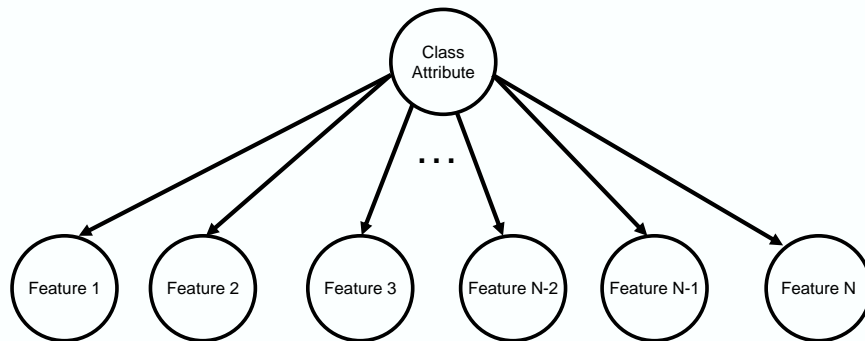


Figure 2.2 A Sample Naïve Bayes Network

The naïve Bayes model, illustrated in Figure 2.2, is a very simple model that involves no learning to determine the network structure. Each attribute has exactly one parent: the class node. For naïve Bayes, only the first learning task needs to be addressed. The drawback to using naïve Bayes is that it assumes that each attribute is independent of all other attributes given the value of the class variable.

A TAN model, illustrated in Figure 2.3, retains the basic structure of naïve Bayes, but also permits each attribute to have at most one other parent. This allows the model to capture a limited set of dependencies between attributes. To decide which arcs to include in the augmented network, the algorithm (Friedman et al., 1997) does the following:

1. Construct a complete graph  $G_A$ , between all non-class attributes  $A_i$ . Weight each edge between  $i$  and  $j$  with the conditional mutual information,  $CI(A_i, A_j|C)$ .
2. Find a maximum weight spanning tree  $T$  over  $G_A$ . Convert  $T$  into a directed graph  $B$ . This is done by picking a node and making all edges outgoing from it.
3. Add an arc in  $B$  connecting  $C$  to each attribute  $A_i$ .

In step 1,  $CI$  represents the conditional mutual information, which is given by in the following equation:

$$CI(A_i; A_j|C) = \sum_{a_i} \sum_{a_j} \sum_c P(a_i, a_j, c) \log \frac{P(a_i, a_j|c)}{P(a_i|c)P(a_j|c)} \quad (2.3)$$

This algorithm for constructing a TAN model has two nice theoretical properties (Friedman et al., 1997). First, it finds the TAN model that maximizes the log likelihood of the network structure given the data. Second, it finds this model in polynomial time.

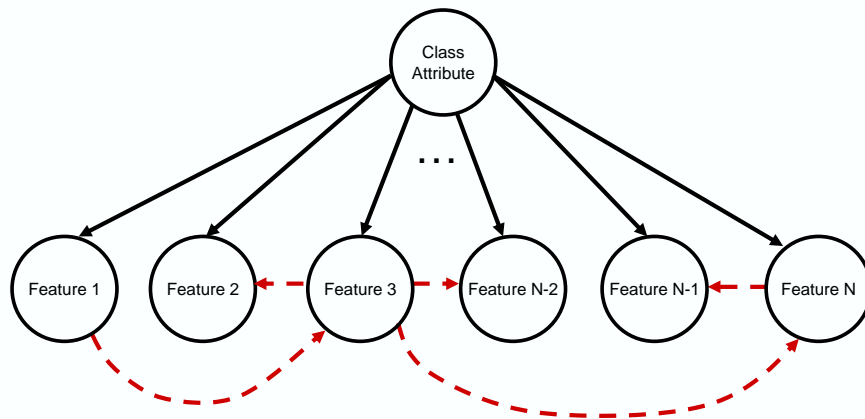


Figure 2.3 A Sample TAN Bayes Network

## 2.2 Inductive Logic Programming

Inductive logic programming (ILP) is a framework for learning relational descriptions (Lavrač & Džeroski, 2001). This paragraph provides a brief overview of first-order logic. For a more comprehensive overview of first-order logic and how it relates to logic programming, please see John Lloyd's (1984) book. First-order logic relies on an alphabet consisting of countable sets of:

- Predicate symbols  $p/n$ , where  $n$  refers to the arity of the predicate and  $n \geq 0$ .
- Function symbols  $f/n$ , where  $n$  refers to the arity of the function and  $n \geq 0$ .
- Variables.

A *term* is a variable or a function  $f(t_1, \dots, t_n)$ , where  $f$  has arity  $n$  and  $t_1, \dots, t_n$  are terms. If  $p/n$  is predicate with arity  $n$  and  $t_1, \dots, t_n$  are terms, then  $p(t_1, \dots, t_n)$  is an *atomic formula*. A *literal* is an atomic formula or its negation. A *clause* is a disjunction over a finite set of literals. A *definite clause* is a clause that contains exactly one positive literal. A *definite program* is a finite set of definite clauses. Definite programs form the basis of logic programming.

ILP is appropriate for learning in multi-relational domains because the learned rules are not restricted to contain fields or attributes for a single table in a database. ILP algorithms learn hypotheses expressed as definite clauses in first-order logic. Commonly-used ILP systems include FOIL (Quinlan, 1990), Progol (Muggleton, 1995) and Aleph (Srinivasan, 2001).

The ILP learning problem can be formulated as follows:

- Given: background knowledge  $B$ , set of positive examples  $E^+$ , set of negative examples  $E^-$  all expressed in first-order definite clause logic.
- Learn: A hypothesis  $H$ , which consists of definite clauses in first-order logic, such that  $B \wedge H \models E^+$  and  $B \wedge H \not\models E^-$ .

In practice, it is often not possible to find either a pure rule or rule set. Thus, ILP systems relax the conditions that  $B \wedge H \models E^+$  and  $B \wedge H \not\models E^-$ .

In our work, we use the Aleph ILP system, which implements the Progol algorithm (Muggleton, 1995) to learn rules. This algorithm induces rules in two steps. Initially, it selects a positive instance to serve as the “seed” example. It then identifies all the facts known to be true about the seed example. The combination of these facts forms the example’s most specific or saturated clause. The key insight of the Progol algorithm is that some of these facts explain this example’s classification. Thus, generalizations of those facts could apply to other examples. The Progol algorithm then performs a top-down refinement search (Muggleton, 1995) over the set of rules that generalize a seed example’s saturated clause.

### **2.3 Statistical Relational Learning**

Statistical Relational Learning (SRL) focuses on algorithms for learning statistical models from relational databases. SRL advances beyond Bayesian network learning and related techniques by handling domains with multiple tables, representing relationships between different rows of the same table, and integrating data from several distinct databases. SRL advances beyond ILP by adding the ability to reason about uncertainty.

Research in SRL has advanced along two main lines: methods that allow graphical models to represent relations and frameworks that extend logic to handle probabilities. Along the first line, Friedman, Getoor, Koller and Pfeffer (1999a) presented an algorithm to learn the structure of probabilistic relational models, or PRMs, which represented one of the first attempts to learn the structure of graphical models while incorporating relational information. Recently Heckerman, Meek and Koller (2004) have discussed extensions to PRMs and compared them to other graphical models. Other graphical approaches include relational dependency networks (Neville & Jensen, 2004) and relational Markov networks (Taskar et al., 2002).

Along the second line, a statistical learning algorithm for probabilistic logic representations was first given by Sato (1995) and Cussens (2001) later proposed a more general algorithm to handle log linear models. Additionally, Muggleton (2000) has provided learning algorithms for stochastic logic programs. The structure of the logic program is learned using ILP techniques,

while the parameters are learned using an algorithm scaled up from stochastic context-free grammar learning. Other logical approaches include Bayesian logic programs (Kersting & Raedt, 2002), constraint logic programming with Bayes net constraints (Santos Costa et al., 2003), logical Bayesian networks (Fierens et al., 2005) and Markov logic networks (MLNs) (Richardson & Domingos, 2006).

Another approach to combining relations and probability is through the design of programming languages. Two examples of this are IBAL (Pfeffer, 2001) and Bayesian logic or BLOG (Milch et al., 2005). A strength of the BLOG approach is that it provides a mechanism to reason about whether or not an object exists in a world.

For the purposes of this dissertation, it is worth going into more detail on PRMs and MLNs. PRMs upgrade Bayesian networks to handle relational data. A PRM relies on being provided with a relational skeleton: the database schema together with the objects present in the domain. It also specifies which attributes are associated with the objects, but it does not include the values for these attributes. In fact, a PRM models the joint distribution over possible settings that all the attributes of all the objects could take. Figure 2.4 shows a simple relational skeleton for a university domain, which has two types of objects, students and classes, and one relation that connects students with classes they have taken.

Like a Bayesian network, a PRM has a parameter  $\theta_{X_i}$  associated with each attribute  $X_i$ , where:  $\theta_{X_i} = P(X_i | Parents(X_i))$ . However, in a PRM a parent could be an attribute in another table. To illustrate the complexities involved, we use an example domain from Getoor et al. (2001). Consider the relational skeleton in Figure 2.4 and assume that *grade* is a parent of *rank*. Note that students take many courses and that students take variable numbers of courses. Consequently, *rank* depends on a multi-set of values for *grade*, and this multi-set varies in size by student. To model this type of dependency, PRMs use an aggregation function to collapse the multi-set into a single value. Algorithms for learning in PRMs can be found in Getoor (2001).

MLNs combine first-order logic with Markov networks. Markov networks are undirected graphical models. Formally, an MLN is a set of pairs,  $(F_i, w_i)$ , where  $F_i$  is a first-order formula and  $w_i \in \mathbb{R}$ . MLNs soften logic by associating a weight with each formula. Worlds that violate

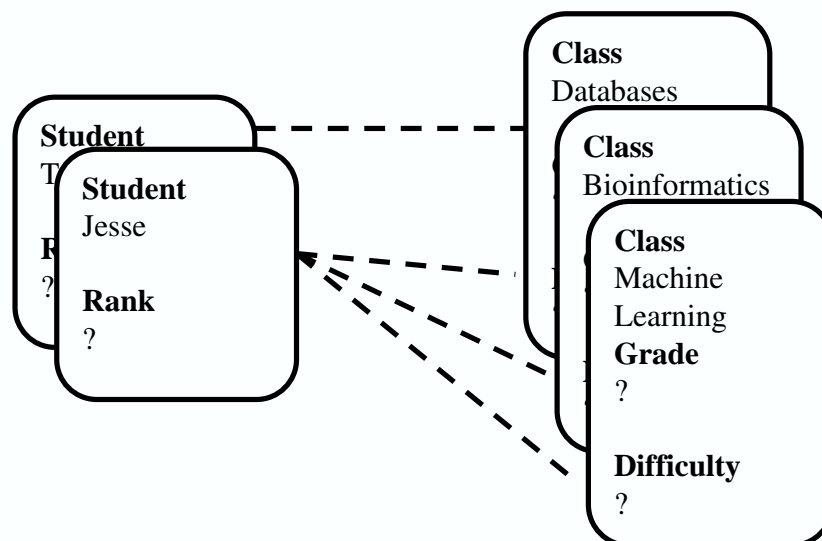


Figure 2.4 A relational skeleton for a university domain. The dashed lines indicate which classes a student has taken. The idea for this skeleton comes from Getoor et al. (2001).

formulas become less likely, but not impossible. Intuitively, as  $w_i$  increases, so does the strength of the constraint  $F_i$  imposes on the world. Formulas with infinite weights represent a pure logic formula.

MLNs provide a template for constructing Markov networks. When given a finite set of constants, the formulas from an MLN define a Markov network. Nodes in the network are the ground instances of the literals in the formulas. Arcs connect literals that appear in the same ground instance of a formula. This discussion ignores how MLNs use the weights to parameterize the underlying Markov network. Information on this process can be found in Richardson and Domingos (2006). Algorithms have been proposed for learning the weights associated (Richardson & Domingos, 2006; Singla & Domingos, 2005) with each formula, as well as for learning the formulas themselves (Kok & Domingos, 2005).

|                    | actual positive | actual negative |
|--------------------|-----------------|-----------------|
| predicted positive | $TP$            | $FP$            |
| predicted negative | $FN$            | $TN$            |

(a) Confusion Matrix

|                     |   |                    |
|---------------------|---|--------------------|
| Recall              | = | $\frac{TP}{TP+FN}$ |
| Precision           | = | $\frac{TP}{TP+FP}$ |
| True Positive Rate  | = | $\frac{TP}{TP+FN}$ |
| False Positive Rate | = | $\frac{FP}{FP+TN}$ |

(b) Definitions of metrics

Figure 2.5 Common Machine Learning Evaluation Metrics

## 2.4 Review of ROC and Precision-Recall Curves

In a binary decision problem, a classifier labels examples as either positive or negative. The decision made by the classifier can be represented in a structure known as a confusion matrix or contingency table. The confusion matrix has four categories: True positives (TP) are examples correctly labeled as positive. False positives (FP) refer to negative examples incorrectly labeled as positive. True negatives (TN) correspond to negatives correctly labeled as negative. Finally, false negatives (FN) refer to positive examples incorrectly labeled as negative.

Figure 2.5(a) shows a confusion matrix. The confusion matrix can be used to construct a point in either ROC space or PR space. Given the confusion matrix, we are able to define the metrics used in each space as in Figure 2.5(b). In ROC space, one plots the False Positive Rate (FPR) on the  $x$ -axis and the True Positive Rate (TPR) on the  $y$ -axis. The FPR measures the fraction of negative examples that are misclassified as positive. The TPR measures the fraction of positive examples that are correctly labeled. In PR space, one plots Recall on the  $x$ -axis and Precision on the  $y$ -axis. Recall is the same as TPR, whereas Precision measures that fraction of examples classified as positive that are truly positive. Figure 2.5(b) gives the definitions for each metric. We will treat the metrics as functions that act on the underlying confusion matrix which defines a point in either ROC space or PR space. Thus, given a confusion matrix  $A$ ,  $\text{RECALL}(A)$  returns the Recall associated with  $A$ .

Often, researchers use the area under the curve (AUC) as a simple metric to define how an algorithm performs over the whole space. The area under the ROC curve (AUC-ROC) is equivalent to the Wilcoxon-Mann-Whitney statistic (Cortes & Mohri, 2003).

## Chapter 3

### **Inductive Logic Programming for Feature Construction: A Case Study in Alias Detection**

One of the challenges raised by complex, multi-relational data, as mentioned in Chapter 1, is the presence of aliases. One gene or protein may have several names. One scientific paper may have multiple, slightly different, citations. One person may have multiple names. In the most difficult case, which this chapter addresses, the person may actually be working to hide the alias. This chapter proposes a two-step methodology for detecting aliases. In the first step, an ILP system learns a set of rules that can achieve high recall, that is, they should be able to recognize most of the true aliases. Unfortunately, some of these rules may have poor precision, meaning that they falsely classify identity pairs as aliases. The second step addresses this problem. Each rule becomes a feature in a new classifier instead of just being treated as an individual classifier. We use Bayesian Networks as our model, as they calculate the probability that a pair of identities are aliases.

We evaluate our approach on synthetic datasets developed by Information Extraction & Transport, Inc. within the EAGLE Project (Schrag, 2004). Each dataset models an artificial world, which is characterized by *individuals*, and relationships between these individuals. Our results show excellent performance for several of the datasets.

This work originally appeared in Davis et al. (2004) and Davis et al. (2005c), which received the Best Technical Paper award at the International Conference on Intelligence Analysis.

### 3.1 Alias Detection and the Challenges It Presents

Determining *identity equivalence*, or alias detection, is a common problem in intelligence analysis. Two different identifiers are equivalent or *aliases* if both refer to the same underlying object. One traditional example of aliasing centers around mistyped or variant author names in documents. For example, one might want to determine if a citation for V.S. Costa and one for Vítor S. Costa refer to the same author. In this situation one evaluates matches based on textual similarity. Furthermore, the central information comes from the surrounding text (Pasula et al., 2003). However, intelligence analysis involves more complex situations, and often syntactic similarity is either unavailable or inapplicable. Instead, aliases must be detected through object attributes and through interaction patterns.

The intelligence analysis domain offers two further challenges. First, information is commonly stored in relational database management systems and involves multiple relational tables. We address this obstacle through the use of ILP, a multi-relational data mining technique that learns *rules* which can contain fields from different tables. Modern ILP systems can handle significant databases, containing millions of tuples.

A second challenge in intelligence analysis arises from *false positives*, or false alarms. In our task, a false positive corresponds to incorrectly hypothesizing that two names refer to the same individual. A false positive might have serious consequences, such as incorrectly adding individuals to a no-fly list. False positives always waste valuable time. False positives reduce trust in the system: if an expert system frequently gives spurious predictions, analysts will ignore its output. For all of these reasons, it is essential to limit the false positive rate. Unfortunately, intelligence analysis is particularly susceptible to false positives, as one is often trying to detect anomalies that occur rarely in the general population. For example, in a city with 1 million individuals, there are 499 billion possible alias pairs. In this case, even a false positive rate of only 0.001% will result in about 5 million false positives, or about five bad aliases per person. To limit the number of false positives, each ILP learned rule becomes a feature in a Bayesian network. The Bayesian network can reason about the quality of each rule when predicting alias pairs.

### 3.2 ILP For Alias Detection

ILP naturally handles multi-relational domains as the learned rules are not restricted to contain fields or attributes for a single table in a database. We use Srinivasan's Aleph ILP System (2001) to learn to predict whether two identifiers refer to the same person. One major advantage of using ILP is that it produces understandable results. We show a sample rule generated by Aleph:

```
Identity Id1 and identity Id2 may be aliases if:
    Id2 is a suspect, and
    there exists another identity Id3, and
    Id3 is a suspect, and
    there is a phonecall between Id2 and Id3, and
    there is a phonecall between Id3 and Id1.
```

The rule says that two individuals, Id1 and Id2 may be aliases if **(i)** they both made phone calls to the same intermediate individual Id3; and **(ii)** individuals Id2 and Id3 have the same attribute (suspect). The rule reflects that in this world model suspects are more likely to have aliases. Moreover, an individual and its aliases tend to talk to the same people.

The next rule uses different information:

```
Identity Id1 and identity Id2 may be aliases if:
    Id1 has capability Cap, and
    Id2 has capability Cap, and
    Id1 belongs to group G, and
    Id2 belongs to group G, and
    G is a nonthreatgroup.
```

Two individuals may be aliases because they have a common capability, and because they both belong to the same non-threat group.

Clearly, these two rules have low precision as the patterns these rules represent could easily apply to ordinary individuals. Note that we are only using the original database schema, whereas an

analyst might define views, or inferred relations, that highlight interesting properties of individuals. For instance, the first rule indicates that an individual and its aliases tend to communicate with the same people. We thus might want to compare sets of people an individual and its aliases talk to. In the spirit of aggregate construction for multi-relational learning (Knobbe et al., 2001; Neville et al., 2003; Perlich & Provost, 2003), we have experimented with hand-crafting rules that use aggregates over properties commonly found in the ILP learned rules. Even after inventing new attributes, it is impossible to find a single rule that correctly identifies all aliases. The next section discusses our approach for combining rules to form a better classifier.

### 3.3 Rule Combination Techniques

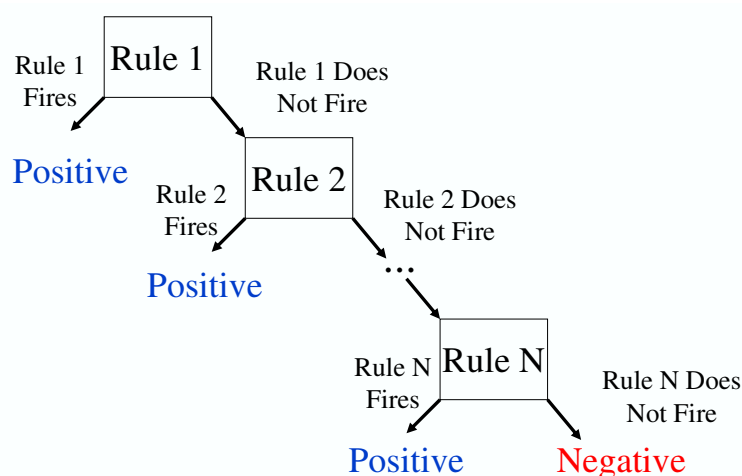


Figure 3.1 Illustration of a Decision List

One of the drawbacks of applying ILP to this problem is that each database for a world is extremely large. Consequently, it is intractable to use all the negative examples when learning the rules, which makes the final set of rules more susceptible to false positives. First, sampling the negative examples alters the proportion of the population that has aliases. Second, in ILP the final classifier traditionally consists of forming a disjunction over the learned clauses, resulting in a decision list as illustrated in Figure 3.1. An unseen example is applied to each clause in succession until it matches one of them. If the example does not match any rule, then it receives the negative

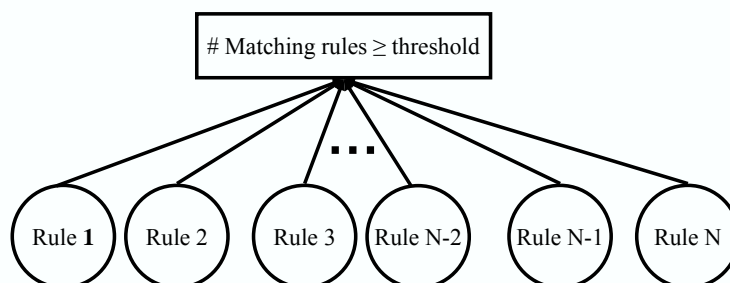


Figure 3.2 Illustration of Unweighted Voting

classification. Unfortunately, the disjunction of clauses maximizes the number of false positives. Unweighted ensemble voting, illustrated in Figure 3.2, generalizes the decision list approach.

These issues suggest a simple approach where we represent each learned rule as an attribute in a classifier. For a given example, the attribute receives a value of one if the example satisfies the rule and a zero otherwise. We use Bayesian networks to combine the rules for two reasons. First, they allow us to set prior probabilities to reflect the true proportion of the population that has aliases. Second, each prediction has a probability attached to it. We can view the probability as a measure of confidence in the prediction. We experiment with several different Bayes net models for combining the rules. Naïve Bayes (Pompe & Kononenko, 1995) is straightforward approach that is easy to understand and fast to train. However, naïve Bayes assumes that the clauses are independent of each other given the class value. Often, we expect the learned rules to be strongly related, so we also use TAN networks, which allow for a slightly more expressive model.

### 3.4 Empirical Evaluation

This section presents our results and analysis of the performance of our system on the U.S. Air Force EAGLE datasets (Schrag, 2004). These datasets simulate an artificial world with large numbers of relationships between agents. The data focuses on *individuals* which have a set of attributes, such as the capability to perform actions. Individuals may also obtain resources, which might be necessary to perform actions. Individuals belong to groups, and groups participate in a wide range of *events*. In our case, given that some individuals may be known through different

identifiers (e.g., through two different phone numbers), we are interested in recognizing whether two identifiers refer to the same individual.

The EAGLE datasets have evolved toward more complex and realistic worlds. Datasets vary along six dimensions:

1. **Population Size**, the number of individuals, rated 1 (smaller) to 4 (larger).
2. **Dataset Size**, the activity level of each individual, rated 1 (smaller) to 4 (larger).
3. **Observability**, the amount of information available as evidence, rated 1 (more information) to 6 (less information).
4. **Corruption**, the number of errors, rated 1 (less corruption) to 4 (more corruption).
5. **Clutter**, the amount of irrelevant information, rated 1 (less noise) to 4 (more noise).
6. **Entity Equivalence**, the level of difficulty of determining aliases, rated easy, fair, hard.

Each experiment is performed in two rounds. In the first round, the *dry-run*, we receive a number of datasets plus their corresponding ground truth, allowing us to experiment with our system and validate our methodology. In the second round, the *wet-run*, we receive datasets without ground truth and are asked to present a hypothesis. We have a limited amount of time to do so. Later, we receive the ground truth so that we can perform our own evaluation.

We adopt the following methodology. Rule learning is quite expensive in these large datasets. Moreover, we have found that most rules are relevant across the datasets, as we believe they capture aspects common to each simulated world. Consequently, we only perform rule learning during the dry-run. We use Srinivasan’s Aleph ILP system (2001) running on the YAP Prolog system. Ground-truth is used for training examples (and not used otherwise). The best rules from all datasets are passed forward to the wet-run.

In our experiments, we test two hypotheses. First, we hypothesize that using a Bayesian network to combine rules will produce more accurate results than using unweighted voting. Second, we hypothesize that combining rules with TAN will lead to the best results.

We evaluate three different rule combination techniques:

1. **Voting** refers to unweighted voting ensemble, where each rule gets one vote.
2. **Naïve Bayes** refers to using the rules learned from the dry-run data to construct a propositional feature vector and using naïve Bayes to produce the final classification.
3. **TAN** refers to the same methodology as before, except it uses a TAN model to produce the final classification.

**Results.** We evaluate our system on datasets generated by two versions of the simulator. We present results on the wet-run data in this chapter. Numbers index results from the first version of the simulator while Roman numerals refer to results for the newer datasets.

We first report results on six datasets from the earlier version of the EAGLE simulator, where each entity can have at most one alias. For the first set of data we use the wet-run datasets plus group information derived by the Kojak system (Adibi et al., 2004) (we did not have access to this information for the second batch of data). Table 3.1 contains the characteristics of each dataset.

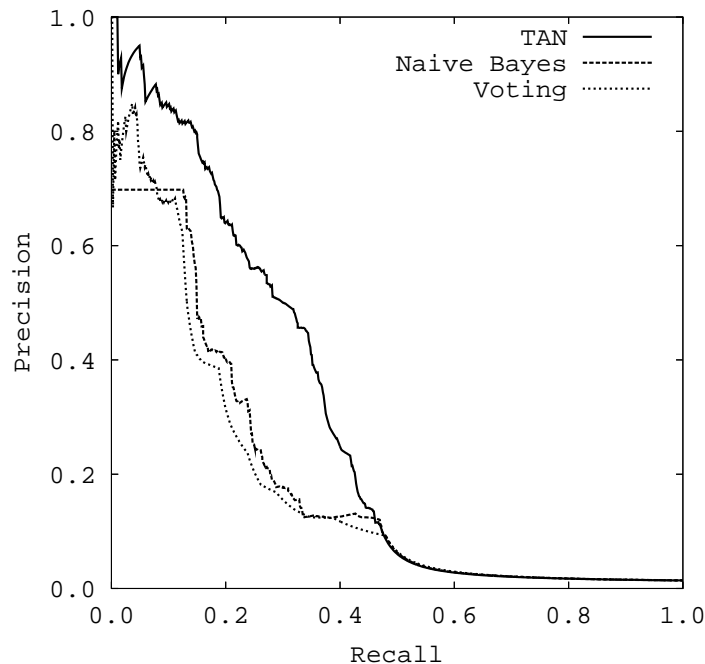


Figure 3.3 Precision-Recall Curve for Dataset 1

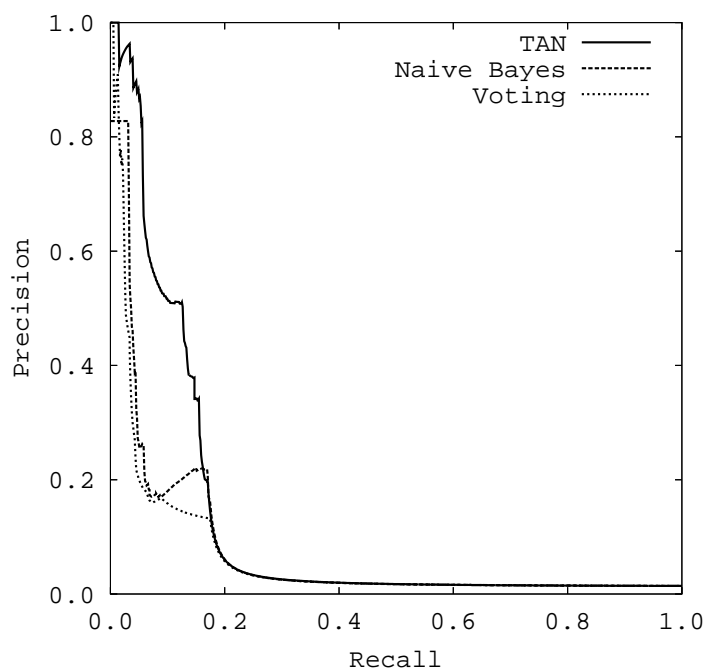


Figure 3.4 Precision-Recall Curve for Dataset 2

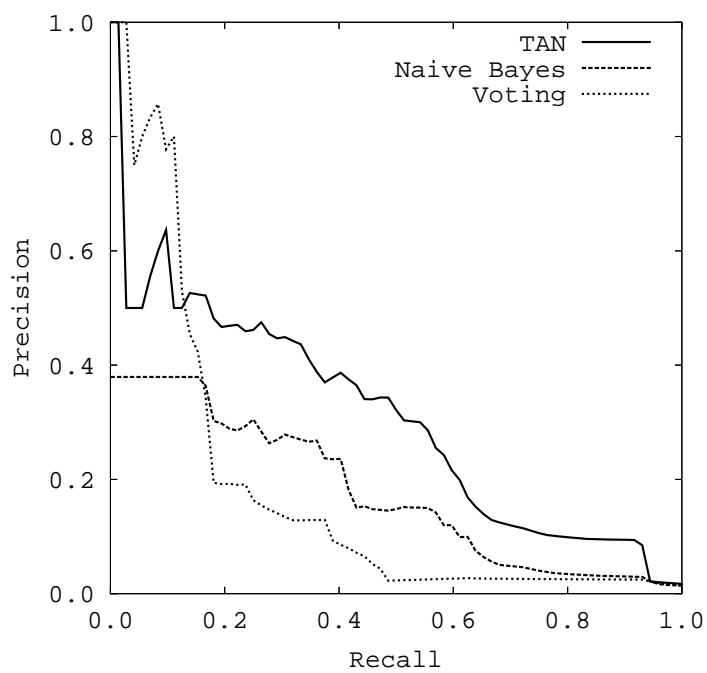


Figure 3.5 Precision-Recall Curve for Dataset 3

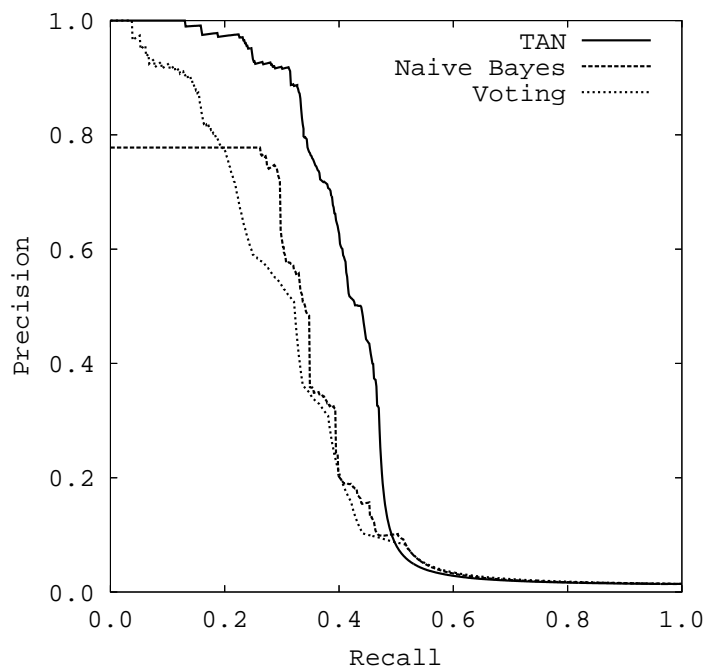


Figure 3.6 Precision-Recall Curve for Dataset 4

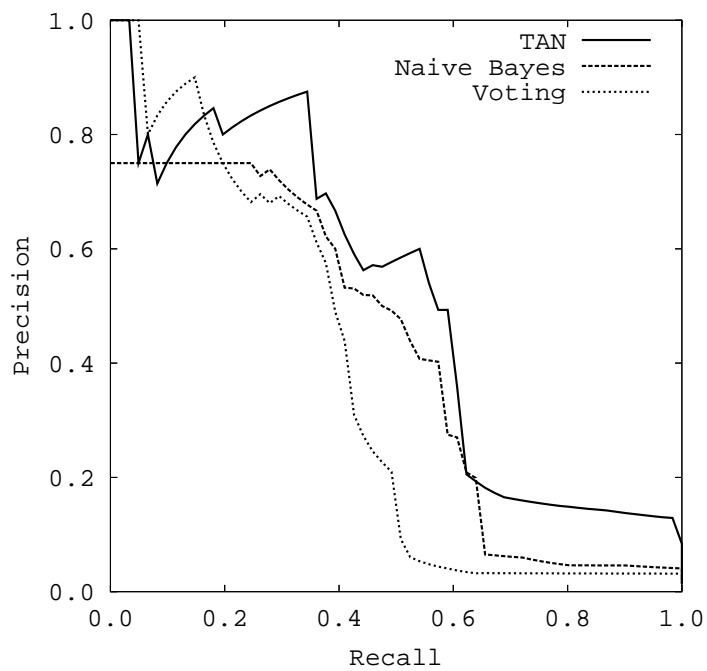


Figure 3.7 Precision-Recall Curve for Dataset 5

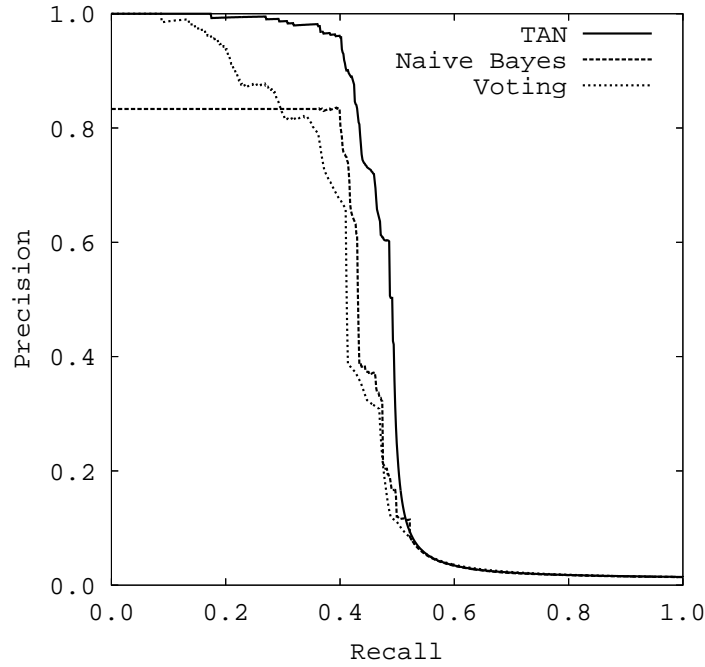


Figure 3.8 Precision-Recall Curve for Dataset 6

| Dataset | Population Size | Dataset Size | Observability | Corruption | Clutter | Entity Equivalence |
|---------|-----------------|--------------|---------------|------------|---------|--------------------|
| 1       | 2               | 2            | 4             | 3          | 3       | fair               |
| 2       | 2               | 2            | 6             | 1          | 3       | fair               |
| 3       | 1               | 2            | 4             | 3          | 3       | fair               |
| 4       | 2               | 2            | 2             | 3          | 3       | fair               |
| 5       | 1               | 2            | 4             | 1          | 3       | fair               |
| 6       | 2               | 2            | 2             | 1          | 3       | fair               |

Table 3.1 Parameter Settings for EAGLE Simulator Version 1

| Dataset | Voting | Naïve Bayes | TAN   | $p$ -value<br>TAN vs. Voting | $p$ -value<br>TAN vs. Naïve Bayes |
|---------|--------|-------------|-------|------------------------------|-----------------------------------|
| 1       | 0.185  | 0.194       | 0.302 | $4.88 \times 10^{-4}$        | $4.51 \times 10^{-4}$             |
| 2       | 0.0712 | 0.0825      | 0.130 | $9.67 \times 10^{-5}$        | $6.53 \times 10^{-4}$             |
| 3       | 0.209  | 0.219       | 0.369 | 0.0188                       | 0.0178                            |
| 4       | 0.311  | 0.313       | 0.430 | $4.35 \times 10^{-5}$        | $6.75 \times 10^{-4}$             |
| 5       | 0.434  | 0.478       | 0.581 | 0.00710                      | 0.0578                            |
| 6       | 0.413  | 0.399       | 0.488 | 0.00196                      | 0.00244                           |

Table 3.2 Average AUC-PR for Each Task from the First Version of the EAGLE Simulator

| Dataset | Population Size | Dataset Size | Observability | Corruption | Clutter | Entity Equivalence |
|---------|-----------------|--------------|---------------|------------|---------|--------------------|
| I       | 1               | 3            | 1             | 4          | 4       | fair               |
| II      | 3               | 3            | 1             | 1          | 3       | easy               |
| III     | 1               | 3            | 2             | 3          | 3       | hard               |
| IV      | 3               | 3            | 4             | 4          | 1       | fair               |
| V       | 3               | 3            | 1             | 3          | 1       | hard               |

Table 3.3 Parameter Settings for EAGLE Simulator Version 2

We use five fold cross validation in these experiments. We pool the results across all five folds to generate the precision-recall curves, which are shown in Figures 3.3 through 3.8. We achieve the best results for datasets 5 and 6, and the worst on dataset 2, where precision levels did not achieve 0.5 for levels of recall  $\geq 0.2$ . Table 3.2 shows the average area under the precision-recall curve (AUC-PR) for each algorithm on these datasets. We perform a paired  $t$ -test comparing the AUC-PR on each task. We found that TAN beats voting with  $p$ -value  $< 0.05$  on all six tasks. TAN beats naïve Bayes with  $p$ -value  $< 0.05$  on every task except for dataset 5, where the  $p$ -value is  $0.05 < p < 0.06$ . However, naïve Bayes only beats voting three times, on datasets 1, 2 and 6.

The second set of results comes from a more recent version of the simulator. Dataset sizes were at least as large, or bigger than before. The new simulator supports social network attributes, which could be used for the aliasing task. These worlds have higher error levels and each individual could have up to six aliases. Table 3.1 contains the dimension settings for each dataset. We use the same methodology as before, except that now we use ten fold cross validation. We pool the results across all ten folds to generate the precision-recall curves. Figures 3.9 through 3.13 show the precision-recall curves from these datasets.

Our results show much better performance for Datasets I and II. This is due to better rule quality. In this case, several rules have excellent recall, and the Bayes nets perform quite well at also achieving good precision. The results are particularly satisfactory using TAN on dataset I, as shown in Figure 3.9, where we can achieve precision over 60% for very high level recall. Dataset IV exhibits the worst overall performance. It shows a case where it is difficult to achieve both high

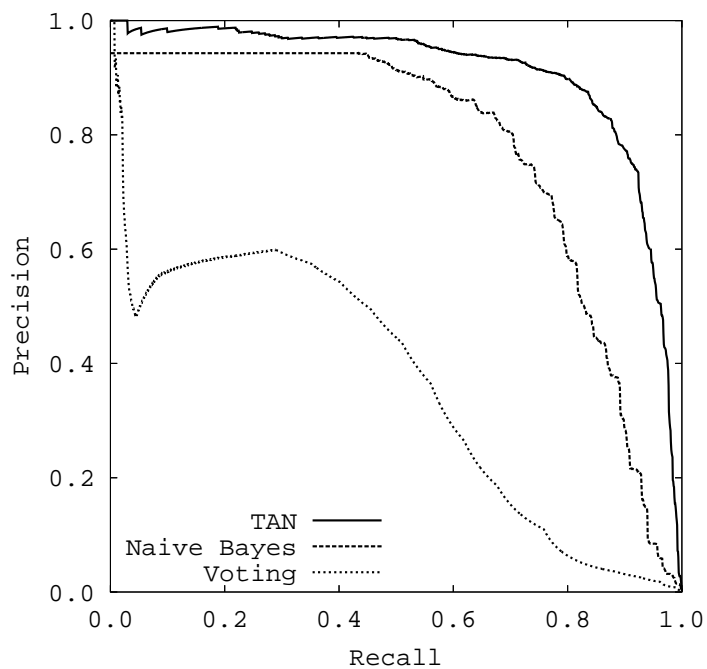


Figure 3.9 Precision-Recall Curve for Dataset I

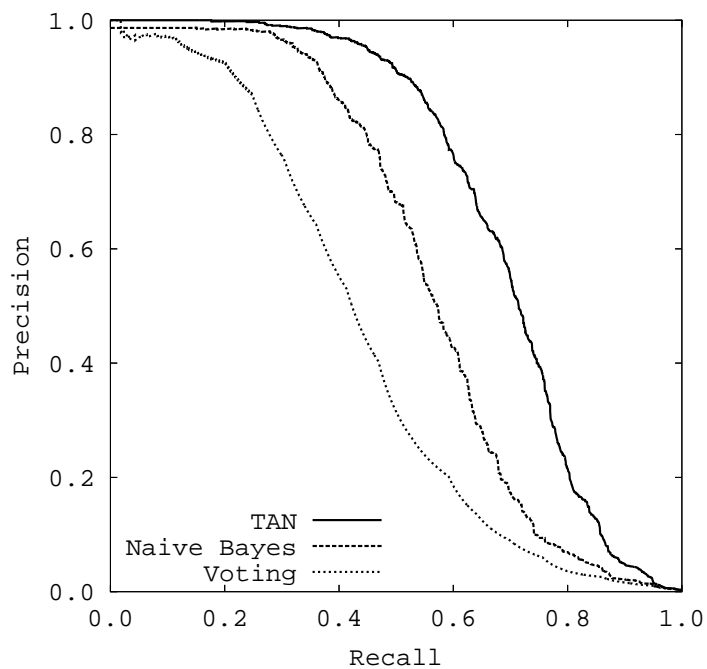


Figure 3.10 Precision-Recall Curve for Dataset II

precision and recall as there is low observability and hence little information about individuals. In this case, improving recall requires relying on only one or two rules, which results in low precision.

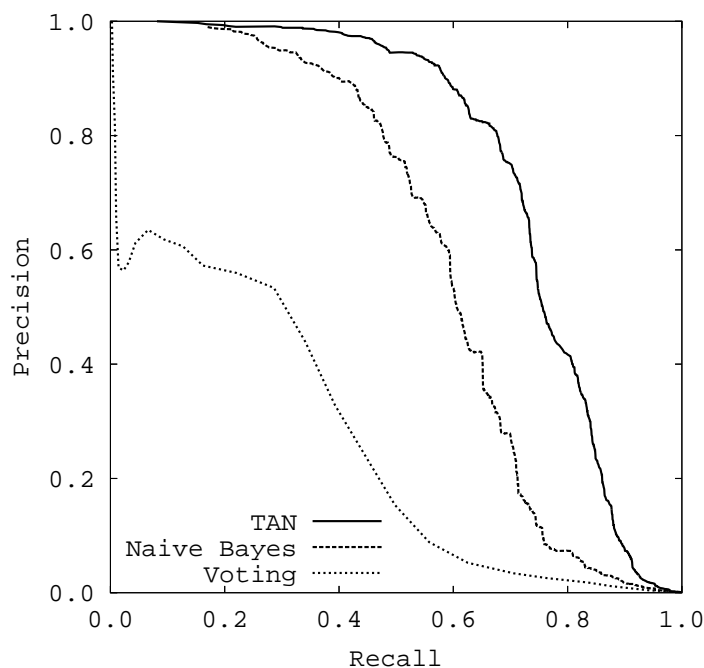


Figure 3.11 Precision-Recall Curve for Dataset III

The precision-recall curve for the TAN algorithm dominates the curves for naïve Bayes and ensemble voting on all five datasets. Table 3.4 shows the average AUC-PR for each algorithm on these datasets. We perform a paired  $t$ -test comparing the AUC-PR on each task and found that TAN beats both naïve Bayes and voting with  $p$ -value  $< 0.01$  on all five tasks. Furthermore, naïve Bayes beats voting with  $p$ -value  $< 0.01$  on all five tasks.

**Discussion.** The results from both versions of the simulator confirm our hypotheses. We see a huge payoff by treating each ILP learned rule as a feature in Bayesian network. The advantage is particularly pronounced when we use a TAN network. For many datasets, several levels of recall exist where TAN yields at least a 20 percentage point increase in precision, for the same level of recall over both naïve Bayes and voting. On eight of eleven of the datasets, naïve Bayes beats voting, while on the remaining three they have comparable performance. One reason for TAN's dominance compared to naïve Bayes is the presence of rules that are simply refinements of other

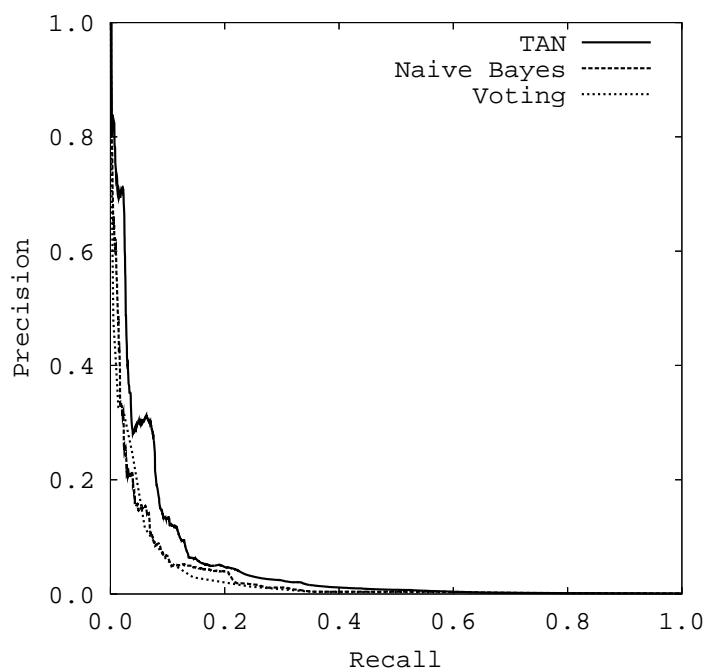


Figure 3.12 Precision-Recall Curve for Dataset IV

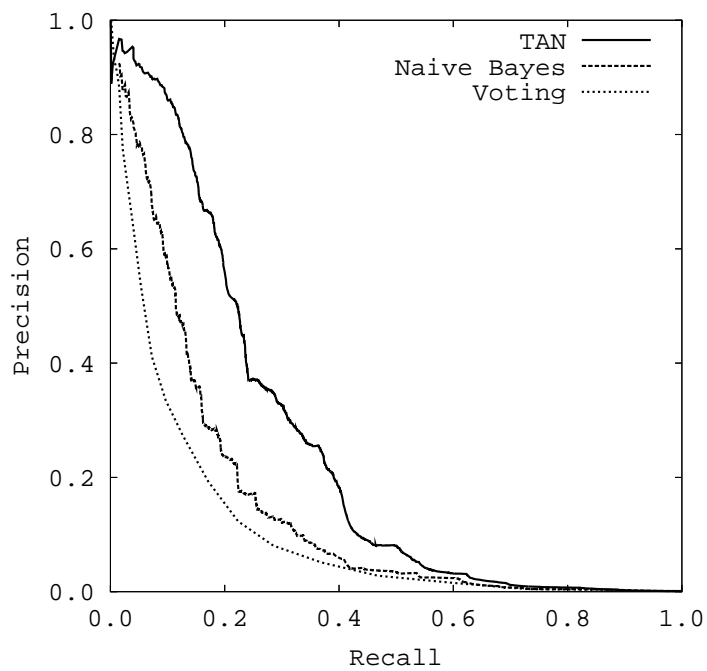


Figure 3.13 Precision-Recall Curve for Dataset V

| Dataset | Voting | Naïve Bayes | TAN    | <i>p</i> -value<br>TAN vs. Voting | <i>p</i> -value<br>TAN vs. Naïve Bayes |
|---------|--------|-------------|--------|-----------------------------------|--|
| I       | 0.372  | 0.784       | 0.916  | $3.01 \times 10^{-13}$            | $1.64 \times 10^{-8}$                  |
| II      | 0.436  | 0.573       | 0.701  | $1.31 \times 10^{-11}$            | $3.94 \times 10^{-10}$                 |
| III     | 0.265  | 0.602       | 0.757  | $1.75 \times 10^{-11}$            | $1.26 \times 10^{-10}$                 |
| IV      | 0.0272 | 0.0342      | 0.0567 | $1.60 \times 10^{-8}$             | $3.20 \times 10^{-7}$                  |
| V       | 0.105  | 0.153       | 0.254  | $4.85 \times 10^{-10}$            | $3.18 \times 10^{-9}$                  |

Table 3.4 Average AUC-PR for Each Task from the Second Version of the EAGLE Simulator

rules. The TAN model captures some of these interdependencies, whereas naïve Bayes explicitly assumes that these dependencies do not exist. The naïve Bayes independence assumption accounts for the similar performance compared to voting on several of the datasets.

In situations with imprecise rules and a preponderance of negative examples, such as link discovery domains, Bayesian models and especially TAN provide an advantage. Both TAN and naïve Bayes excel at handling imprecise rules. The Bayes nets effectively weight the precision of each rule either individually or based on the outcome of another rule in the case of TAN. The Bayesian nets further combine these probabilities to make a prediction of the final classification allowing them to discount the influence of spurious rules in the classification process. Ensemble voting does not have this flexibility and consequently lacks robustness to imprecise rules. Another area where TAN provides an advantage is when multiple imprecise rules provide significant overlapping coverage on positive examples and a low level of overlapping coverage on negative examples. The TAN network can model this scenario and weed out the false positives. One potential disadvantage to the Bayesian approach is that it could be overly cautious about classifying something as a positive. The high number of negative examples relative to the number of positive examples, and the corresponding concern of a high false positive rate, helps mitigate this potential problem. In fact, at similar levels of recall, TAN has a lower false positive rate than voting.

### 3.5 Related Work

Identity uncertainty is a difficult problem that arises in areas such as citation matching, record linkage and de-duplication in databases, natural language processing, in addition to intelligence analysis. A seminal work in this area is the theory of record linkage (Fellegi & Sunter, 1969), based on scoring the distances between two feature vectors (using naïve Bayes in the original work) and merging records below some threshold. Systems such as Citeseer (Lawrence et al., 1999) apply analogous ideas by using text similarity. The field of record matching has received significant contributions (Monge & Elkan, 1996; Cohen & Richman, 2002; Buechi et al., 2003; Bilenko & Mooney, 2003; Zelenko et al., 2003; Hsiung et al., 2004). On the other hand, it has been observed that interactions between identifiers can be crucial in identifying them (Morton, 2000). Pasula et al. (2003) use relational probabilistic models to establish a probabilistic network of individuals, and then use Markov Chain Monte Carlo to do inference on the citation domain. McCallum and Wellner (2003) use discriminative models, Conditional Random Fields, for the same task. These approaches rely on prior understanding of the features of interest, usually text based. Such knowledge may not be available for intelligence analysis tasks.

The present work builds upon previous work on using ILP for feature construction. Such work treats ILP-constructed rules as Boolean features, re-represents each example as a feature vector, and then uses a feature-vector learner to produce a final classifier. The first work on propositionalization is the LINUS system (Lavrač & Džeroski, 1992). LINUS transforms the examples from deductive database format into attribute-values tuples and pairs these tuples to a propositional learner. LINUS primary uses propositional algorithms that generate if-then rules. LINUS then converts the propositional rules back into the deductive database format.

Pompe and Kononenko (1995) were the first to apply naïve Bayes to combine clauses. Some other work, especially on propositionalization of first-order logic (Alphonse & Rouveirol, 2000), has been developed that converts the training set to propositions and then applies feature vector techniques to the converted data. This is similar to what we do, however we first perform learning

in the first-order setting to determine which features to construct. This results in significantly shorter feature vectors than in other work.

A popular alternative to decision lists is *voting*. Voting has been used in ensemble-based approaches, such as bagging (Breiman, 1996; Dutra et al., 2002) and boosting (Hoche & Wrobel, 2001; Quinlan, 1996). Boosting relies on the insight that one should focus on misclassified examples. Search is directed by having a sequence of steps, such that at each consecutive step misclassified examples become more and more valuable. We do not change example weights at each step. Instead, we rely on the classifier itself and trust the tuning data to give us approximate performance of the global system. On the other hand, we do try to focus the search on examples where we perform worse, by skewing seed selection.

ROCCER is a more recent example of a two-step algorithm that starts from a set of rules and tries to maximize classifier performance (Prati & Flach, 2005). ROCCER takes a set of rules, and returns a subset that corresponds to a convex hull in ROC space. ROCCER relies on the Apriori algorithm (Agrawal et al., 1993) to obtain the set of rules.

This chapter contributes two novel points to ILP based feature construction. First, it highlights the relationship between this category of work and ensembles in ILP, because when the feature-vector learner is naïve Bayes the learned model can be considered a weighted vote of the rules. Second, it shows that when the features are ILP-learned rules, the independence assumption in naïve Bayes may be violated badly enough that it lowers the precision of the system. However, a TAN model's ability to explicitly represent strong dependencies can improve the precision of the system.

### 3.6 Chapter Summary

Identity equivalence is an important problem in a variety of application areas involving complex multi-relational databases, including biology, medicine and intelligence analysis. In the most challenging case, individuals want to hide their identities, and therefore we cannot rely on textual information. Instead, we need to use attributes and contextual information to detect aliases. To

tackle this problem, we propose using ILP to construct features for a propositional learner, which then makes the final prediction about which identifiers refer to the same individuals.

We compare three different approaches for combining rules learned by an ILP system. We evaluate the performance of each approach on series of synthetic datasets, where information is subject to corruption and unobservability. We demonstrate that using Bayesian networks for rule combination can significantly improve the precision of the classifier. We obtain the best results through the use of a TAN network as it is more robust at handling dependencies between rules.

## Chapter 4

### Multi-Step View Learning Framework

The previous chapter described how to use ILP to define new features for a propositional classifier. This chapter discusses how ILP based feature construction can be used to address the weakness of many SRL frameworks: that they are constrained to use only the tables and fields already in the database, without modification. Many human users of relational databases find it beneficial to define alternative *views* of a database—further fields or tables that can be computed from existing ones. This chapter shows that SRL algorithms also can benefit from the ability to define new views, which can be used for more accurate prediction of important fields in the original database. This chapter demonstrates how to augment SRL algorithms by adding the ability to learn new fields, intentionally defined in terms of existing fields and intensional background knowledge. In database terminology, these new fields constitute a learned *view* of the database.

We present view learning in the specific application of creating an expert system in mammography. We chose this application for a number of reasons. First, it is an important practical application where there has been recent progress in collecting sizable amounts of data. Second, we have access to an expert-developed system. This provides a base reference against which we can evaluate this work (Burnside et al., 2000). Third, a large proportion of examples are negative. This distribution skew is often found in multi-relational applications. Last, the data consist of a single table. This allows for an easy comparison to standard propositional learning. In this case, it is sufficient for view learning to extend an existing table with new fields, achieved by using ILP to learn rules for unary predicates. For other applications, it may be desirable to learn predicates of higher-arity, which will correspond to learning a view with new tables rather than new fields

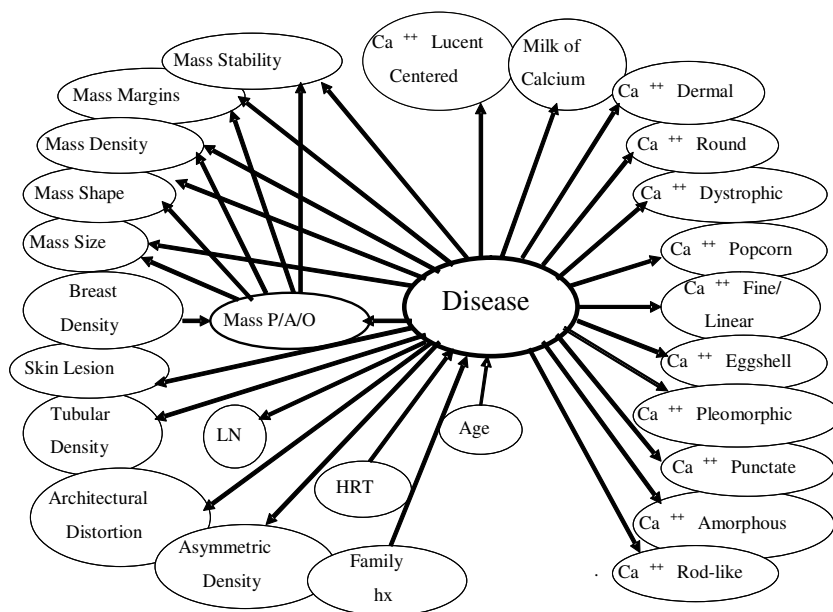


Figure 4.1 Expert Bayesian Network for Mammography

only. Chapter 7 will discuss how to extend view learning to learn new relations and higher-arity predicates.

This work originally appeared in Davis et al. (2005b) and Burnside et al. (2005).

## 4.1 Mammography

Offering breast cancer screening to the ever-increasing number of women over age 40 represents a great challenge. Cost-effective delivery of mammography screening depends on a consistent balance of high sensitivity and high specificity. It has been demonstrated that sub-specialist, expert mammographers achieve this balance and perform significantly better than general radiologists (Brown et al., 1995; Sickles et al., 2002). General radiologists have higher false positive rates and hence biopsy rates, diminishing the positive predictive value for mammography (Brown et al., 1995; Sickles et al., 2002). Unfortunately, despite the fact that specially trained mammographers detect breast cancer more accurately, there is a longstanding shortage of these individuals (Eklund, 2000).

| Id  | Patient | Date | Mass<br>Shape | ... | Mass<br>Size | Loc | Benign/<br>Malignant |
|-----|---------|------|---------------|-----|--------------|-----|----------------------|
| 1   | P1      | 5/02 | Oval          |     | 3mm          | RU4 | B                    |
| 2   | P1      | 5/04 | Round         |     | 8mm          | RU4 | M                    |
| 3   | P1      | 5/04 | Oval          |     | 4mm          | LL3 | B                    |
| 4   | P2      | 6/00 | Round         |     | 2mm          | RL2 | B                    |
| ... | ...     | ...  | ...           |     | ...          | ... | ...                  |

Figure 4.2 Abbreviated National Mammography Database Schema

An expert system in mammography has the potential to help the general radiologist approach the effectiveness of a sub-specialty expert, thereby minimizing both false negative and false positive results. Bayesian networks are probabilistic graphical models that have been applied to the task of breast cancer diagnosis from mammography data (Burnside et al., 2000; Burnside et al., 2004b; Kahn et al., 1997). Bayesian networks produce diagnoses with probabilities attached. Because of their graphical nature, they are comprehensible to humans and useful for training. Figure 4.1 shows the structure of a Bayesian network developed by a sub-specialist, expert mammographer. The Bayesian network in Figure 4.1 achieves accuracies higher than those of other systems and of general radiologists who perform mammograms, and commensurate with the performance of radiologists who specialize in mammography (Burnside et al., 2000).

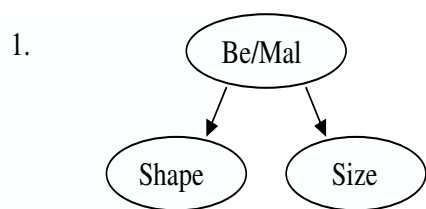
Table 4.2 shows the main table (with some fields omitted for brevity) in a large relational database of mammography abnormalities. The database schema is specified in the National Mammography Database (NMD) standard established by the American College of Radiology (2004). The NMD was designed to standardize data collection for mammography practices in the United States and is widely used for quality assurance. We omit, a second, much smaller *biopsy* table, simply because we want to predict—before the biopsy—whether an abnormality is benign or malignant. Note that the database contains one record per abnormality. By putting the database into one of the standard database “normal” forms, it would be possible to reduce some data duplication,

but only a very small amount: the patient’s age, status of hormone replacement therapy and family history could be recorded once per *patient and date* in cases where multiple abnormalities are found on a single mammogram date. Such normalization would have no effect on our approach or results, so we will operate directly on the database in its defined form.

## 4.2 Learning Hierarchy

Figure 4.3 presents a hierarchy of the four types of learning that might be used for this task. Level 1 and Level 2 are standard types of Bayesian network learning. Level 1 is simply learning the parameters for the expert-defined network structure. Level 2 involves learning the actual structure of the network in addition to its parameters. Notice that to predict the probability of malignancy of an abnormality, a Bayesian network uses only the record for that abnormality. However, data in other rows of the table may also be relevant: radiologists may consider other abnormalities on the same mammogram or previous mammograms. For example, it may be useful to know that the same mammogram also contains another abnormality, with a particular size and shape; or that the same person had a previous mammogram with certain characteristics. Incorporating data from other rows in the table is not possible with existing Bayesian network learning algorithms and requires SRL techniques, such as probabilistic relational models (Friedman et al., 1999a). Level 3 in Figure 4.3 shows the state-of-the-art in SRL techniques, illustrating how relevant fields from other rows (or other tables) can be incorporated into the network, using aggregation if necessary. Rather than using only the size of the abnormality under consideration, the new aggregate field allows the Bayes net to also consider the average size of all abnormalities found in the mammogram.

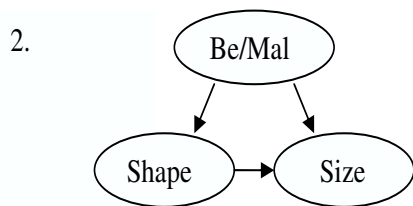
Presently, SRL is limited to using the original view of the database, that is, the original tables and fields, possibly with aggregation. Despite the utility of aggregation, simply considering only the existing fields may be insufficient for accurate prediction of malignancies. Level 4 in Figure 4.3 shows the key capability that will be introduced and evaluated in this dissertation: using techniques from rule learning to learn a new *view*. In this figure, the new view includes two new features utilized by the Bayes net that cannot be defined simply by aggregation of existing features. The new features are defined by two learned rules that capture “hidden” concepts potentially useful for



*Parameter Learning:*

**Given:** Features (node labels, or fields in database), Data, Bayes net structure

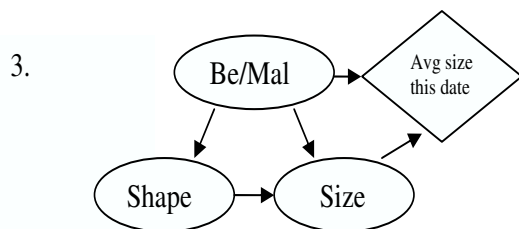
**Learn:** Probabilities. Note: probabilities needed are  $Pr(\text{Be/Mal})$ ,  $Pr(\text{Shape}|\text{Be/Mal})$ ,  $Pr(\text{Size}|\text{Be/Mal})$



*Structure Learning:*

**Given:** Features, Data

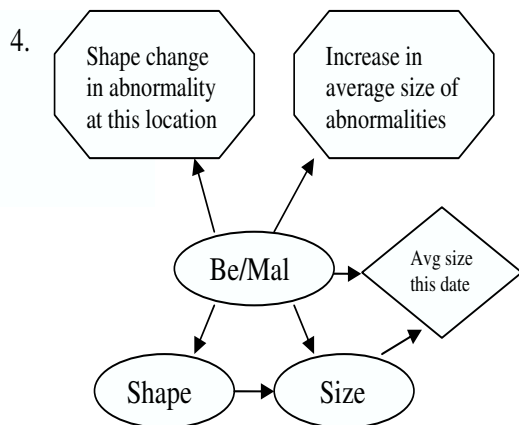
**Learn:** Bayes net structure and probabilities. Note: with this structure, now will need  $Pr(\text{Size}|\text{Shape}, \text{Be/Mal})$  instead of  $Pr(\text{Size}|\text{Be/Mal})$ .



*Aggregate Learning:*

**Given:** Features, Data, Background knowledge – aggregation functions such as average, mode, max, etc.

**Learn:** Useful aggregate features, Bayes net structure that uses these features, and probabilities. New features may use other rows/tables.



*View Learning:*

**Given:** Features, Data, Background knowledge – aggregation functions and *intensionally-defined relations* such as “increase” or “same location”

**Learn:** Useful *new features defined by views* (equivalent to *rules* or *SQL queries*), Bayes net structure, and probabilities.

Figure 4.3 Hierarchy of learning types. Levels 1 and 2 are available through ordinary Bayesian network learning algorithms, Level 3 is available only through state-of-the-art SRL techniques, and Level 4 is described in this chapter.

accurately predicting malignancy, but that are not explicit in the given database tables. One learned rule states that a change in the shape of an abnormality at a location since an earlier mammogram may be indicative of a malignancy. The other says that an *increase* in the average of the sizes of the abnormalities may be indicative of malignancy. Note that both rules require reference to other rows in the table for the given patient, as well as intensional background knowledge to define concepts such as “increases over time.” Neither rule can be captured by standard aggregation of existing fields.

Note that Level 3 and Level 4 learning would not be necessary if the database initially contained all the potentially useful fields capturing information from other relevant rows or tables. For example, the database might be initially constructed to contain fields such as “slope of change in abnormality size at this location over time”, “average abnormality size on this mammogram”, and so on. If humans can identify all such potentially useful fields beforehand and define views containing these, then Level 3 and Level 4 learning are unnecessary. Since the space of such *possibly* useful fields is quite large, it is more easily searched by a computer via Level 3 and Level 4 learning. Certainly in the case of the NMD standard, such fields are not available because they have not been defined and populated in the database by the domain experts, thus making Level 3 and Level 4 learning potentially useful.

### 4.3 Multi-Step View Learning

One can imagine a variety of approaches to perform view learning. As a first step, we apply existing technology to obtain a view learning capability. The initial view learning framework, illustrated in Figure 4.4, works in three steps. First, it learns rules to predict whether an abnormality is malignant. Second, it selects the relevant subset of the rules to include in the model and extends the original database by introducing the new rules as *additional features*. More precisely, each rule will correspond to a binary feature such that it takes the value *true* if the body, or condition, of the rule is satisfied, and *false* otherwise. Third, it runs a Bayesian network structure learning algorithm, allowing it to use these new features in addition to the original features, to construct

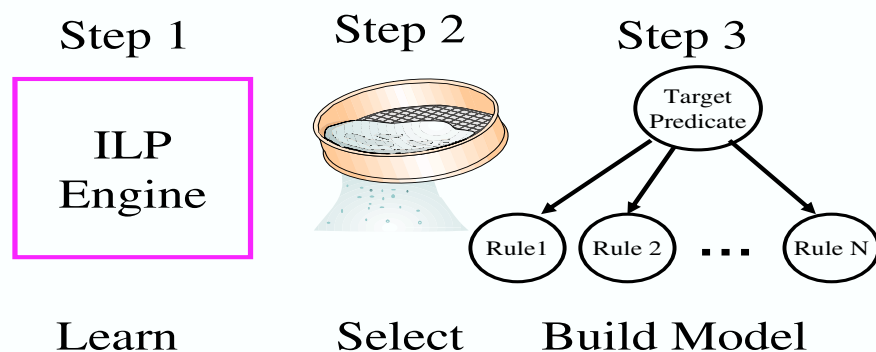


Figure 4.4 Multi-step approach to view learning. The first step takes as input a set of examples and background knowledge and produces a set of rules. The second step takes the set of rules and selects the relevant rules. The final step incorporates the selected rules as features in the final statistical model.

a model. Section 4.10 notes the relationship of the approach to earlier work on ILP for feature construction.

Figure 4.5 shows a potentially important piece of information a radiologist might use when classifying an abnormality. An ILP system could derive this concept by learning the following rule:

Abnormality A in mammogram M may be malignant if:

- A has mass size S1, and
- A has a prior abnormality A2, and
- A is same location as A2, and
- A2 has mass size S2, and
- $S1 > S2$ .

Note that the last three lines of the rule refer to other rows of the relational table for abnormalities in the database. Hence this rule encodes information not available to the initial version of the Bayesian network (Davis et al., 2005b). This rule can be used as a field in a new view of the database, as illustrated in Figure 4.6, and consequently as a new feature in the Bayesian network.

| Id  | Patient | Date | Mass Shape | ... | Mass Size | Loc | Benign/<br>Malignant |
|-----|---------|------|------------|-----|-----------|-----|----------------------|
| 1   | P1      | 5/02 | Oval       |     | 3mm       | RU4 | B                    |
| 2   | P1      | 5/04 | Round      |     | 8mm       | RU4 | M                    |
| 3   | P1      | 5/04 | Oval       |     | 4mm       | LL3 | B                    |
| 4   | P2      | 6/00 | Round      |     | 2mm       | RL2 | B                    |
| ... | ...     | ...  | ...        |     | ...       | ... | ...                  |

Figure 4.5 A Possible View: Size of a Mass Increases Over Time

| Id  | Patient | Date | Mass Shape | ... | Mass Size | Loc | Increase<br>Size<br>Same Loc | Benign/<br>Malignant |
|-----|---------|------|------------|-----|-----------|-----|------------------------------|----------------------|
| 1   | 1       | 5/02 | Oval       |     | 3mm       | RU4 | No                           | B                    |
| 2   | P1      | 5/04 | Round      |     | 8mm       | RU4 | Yes                          | M                    |
| 3   | P1      | 5/04 | Oval       |     | 4mm       | LL3 | No                           | B                    |
| 4   | P2      | 6/00 | Round      |     | 2mm       | RL2 | No                           | B                    |
| ... | ...     | ...  | ...        |     | ...       | ... | ...                          | ...                  |

Figure 4.6 The National Mammography Database Schema Extended with an Additional Field

## 4.4 Data

We collected data for all screening and diagnostic mammography examinations that were performed at the Froedtert and Medical College of Wisconsin Breast Imaging Center between April 5, 1999 and February 9, 2004. It is important to note that the data consist of a radiologist's interpretation of a mammogram and not the raw image data. The radiologist reports conformed to the NMD standard established by the American College of Radiology. From these reports, we followed the original network (Burnside et al., 2000) to cull the 36 features deemed to be relevant by an expert mammographer.

The data contain 435 malignant abnormalities and 65,365 benign abnormalities. Ground truth is determined by biopsy. This implies that the data may contain false negatives, as radiologists do not biopsy all abnormalities.

## 4.5 Experiments

In our experiments we want to test three hypotheses. First, we want to determine if using SRL yields an improvement compared to propositional learning. Second, we want to evaluate whether moving up a level in the hierarchy outlined in Figure 4.3 results in an improvement in performance. Third, we want to evaluate whether the rules learned during the view learning process contain useful information for radiologists.

To test the hypotheses, we employ four different algorithms. First, we try to learn a structure with just the original attributes (Level 2) and see if that performs better than using the expert structure with trained parameters (Level 1). Next, we add aggregate features to our network, representing summaries of abnormalities found either in a particular mammogram or for a particular patient. This corresponds to Level 3 and we test whether this improves over Levels 1 and 2. Finally, we investigate Level 4 learning through the multi-step algorithm and compare its performance to Levels 1 through 3. The following section will give detailed descriptions for each of these algorithms.

## 4.6 Approach for Each Level of Learning

**Level 1: Parameter Learning.** We estimate the parameters of the expert structure from the dataset using maximum likelihood estimates with Laplace correction. It has been previously noted that learning the parameters of the network improves performance over having expert defined probabilities in each node (Burnside et al., 2004a).

**Level 2: Structure Learning.** The relational database for the mammography data contains one row for each abnormality described on a mammogram. Fields in this relational table include all

| Id  | Patient | Date | Mass<br>Shape | ... | Mass<br>Size | Average<br>Size<br>Study | Average<br>Size<br>Patient | Loc | Benign/<br>Malignant |
|-----|---------|------|---------------|-----|--------------|--------------------------|----------------------------|-----|----------------------|
| 1   | P1      | 5/02 | Oval          |     | 3mm          | 3mm                      | 3mm                        | RU4 | B                    |
| 2   | P1      | 5/04 | Round         |     | 8mm          | 6mm                      | 5mm                        | RU4 | M                    |
| 3   | P1      | 5/04 | Oval          |     | 4mm          | 6mm                      | 5mm                        | LL3 | B                    |
| 4   | P2      | 6/00 | Round         |     | 2mm          | 2mm                      | 2mm                        | RL2 | B                    |
| ... | ...     | ...  | ...           |     | ...          | ...                      | ...                        | ... | ...                  |

Figure 4.7 Database after Aggregation on Mass Size Field. Note the addition of two new fields, Average Size Patient and Average Size Study, which represent aggregate features.

those shown in the Bayesian network of Figure 4.1. Therefore it is straightforward to use existing Bayesian network structure learning algorithms to learn a possibly improved structure for the Bayesian network.

**Level 3: Aggregate Learning.** We select the numeric (e.g. the size of mass) and ordered features (e.g. the density of a mass) in the database and compute aggregates for each of these features. In all, we determine that 27 of the 36 attributes were suitable for aggregation. We compute aggregates on both the patient and the mammogram level. On the patient level, we look at all of the abnormalities for a specific patient. On the mammogram level, we only consider the abnormalities present on that specific mammogram. To discretize the averages, we divide each range into three bins. For binary features we use predefined bin sizes, while for the other features we attempt to get equal numbers of abnormalities in each bin. For aggregation functions we use maximum and average. The aggregation introduces  $27 \times 4 = 108$  new features. The following paragraph presents further details on the aggregation process.

Constructing aggregate features involves a three-step process. First, choose a field to aggregate. Second, select an aggregation function. Third, decide which rows to include in the aggregate feature, that is, which keys or links to follow. This is known as a *slot chain* in Probabilistic Relational Model (PRM) terminology (Friedman et al., 1999a). In the mammography database, two

such links exist. The patient ID field allows access to all the abnormalities for a given patient, providing aggregation on the patient level. The second key is the combination of patient ID and mammogram date, which returns all abnormalities for a patient on a specific mammogram, providing aggregation on the mammogram level. To demonstrate this process, we will work through an example of computing an aggregate feature for patient 1 in the database given in Table 4.2. We will aggregate on the Mass Size field and use average as the aggregation function. Patient 1 has three abnormalities, one from a mammogram in May 2002 and two from a mammogram in May 2004. To calculate the aggregate on the patient level, we actually have to perform two computations, as we don't want the aggregate score for the abnormality in May 2002 to reflect the later findings. For the May 2002 mammogram, we only look at averaging size of abnormality 1, and the average size is 3mm. For the abnormalities 2 and 3, we average the size for all three abnormalities, which is 5mm. To find the aggregate on the mammogram level for patient 1, we have to perform two separate computations. First, follow the link P1 and 5/02, which yields abnormality 1. The average for this key mammogram is simply 3mm. Second, follow the link P1 and 5/04, which yields abnormalities 2 and 3. The average for these abnormalities is 6mm. Table 4.7 shows the database following construction of these aggregate features.

**Level 4: View Learning.** We use the ILP system Aleph (Srinivasan, 2001) to implement Level 4 learning. We introduce three new intensional tables into Aleph's background knowledge to take advantage of relational information.

1. The `prior_Mammogram` relation connects information about any prior abnormality that a given patient may have.
2. The `same_Location` relation is a specification of the previous predicate. It adds the restriction that the prior abnormality must be in the same location as the current abnormality. Radiology reports include information about the location of abnormalities.
3. The `in_Same_Mammogram` relation incorporates information about other abnormalities a patient may have on the current mammogram.

By default, Aleph generates rules that would fully explain the examples. In contrast, our goal is to extract rules that would be beneficial as new views. The major challenge in implementing Level 4 learning is *how to select rules that would best complement Level 3 information*. Clearly, Aleph’s standard coverage algorithm is not designed for this application. Instead, we first enumerate as many rules of interest as possible, and then pick interesting rules.

In order to obtain a varied set of rules, we run Aleph under the `induce_max` setting, which uses every positive example in each fold as a seed for the search. Also note that it does not discard previously covered examples when scoring a new clause. Aleph learns several thousand distinct rules for each fold, with each rule covering many more malignant cases than (incorrectly covering) benign cases. To avoid the rule overfitting found by other authors (Perlich & Provost, 2003), we use breadth-first search for rules and set a minimal limit on coverage.

Each seed generates anywhere from zero to tens of thousands of rules. Adding all rules would mean introducing thousands of often redundant features. We use the following algorithm to select which rules to include in the model:

1. Scan all rules removing duplicates and rules that perform worse than a more general rule. This step significantly reduces the number of rules to consider.
2. Sort the rules according to their  $m$ -estimate of precision. We use Aleph’s default value for  $m$ , which sets  $m = \sqrt{\text{positives covered} + \text{negatives covered}}$
3. Pick the rule with the highest  $m$ -estimate of precision such that it covers an unexplained training example. Furthermore, each rule needs to cover a significant number of malignant cases. This step is similar to the standard ILP greedy covering algorithm, except that it does not follow the original order of the seed examples.
4. Scan the remaining rules, selecting those that cover a significant number of examples, and that are different from all previous rules, *even if these rules do not cover any new examples*.

It is important to note that the rule selection is an automated process. It picks the top 50 clauses to include in the final learned model. We incorporate the resulting views as new features in the database.

## 4.7 Methodology

To evaluate and compare these approaches, we use stratified 10-fold cross-validation. We randomly divide the abnormalities into 10 roughly equal-sized sets, each with approximately one-tenth of the malignant abnormalities and one-tenth of the benign abnormalities. When evaluating just the structure learning and aggregation, we use nine folds for the training set. When performing aggregation, we use binning to discretize the created features and took care to only use the examples from the training set to determine the bin widths. When performing view learning, we had three steps in the learning process. The first part uses four folds of data to learn the ILP rules. The second part selects the rules to include in the model. The third part uses the remaining five folds to learn the Bayes net structure and parameters.

When using cross-validation on a relational database, there exists a potential methodological pitfall. Some of the cases may be related. For example, a single patient may have multiple abnormalities. Because these abnormalities are related (same patient), having some of these in the training set and others in the test set may improve performance better on those test cases relative to the performance on test cases for other patients. To avoid such “leakage” of information into a training set, we ensure that all abnormalities associated with a particular patient appear in the same fold for cross-validation. Another potential pitfall is learning a rule that predicts an abnormality to be malignant based on properties of abnormalities in *later* mammograms. We ensure that we will never predict the status of an abnormality at a given date based on findings recorded for later dates.

## 4.8 Results

We experimented with a number of structure learning algorithms for Bayesian Networks, including naïve Bayes, TAN, and the Sparse Candidate Algorithm (see Section 2.4 for descriptions). However, we obtained the best results with the TAN algorithm in all experiments, so the discussion will focus on TAN.

We present the results of the first experiment, comparing Levels 1 and 2, using both ROC and precision-recall curves. Figure 4.8 shows the ROC curve for these experiments, and Figure 4.9

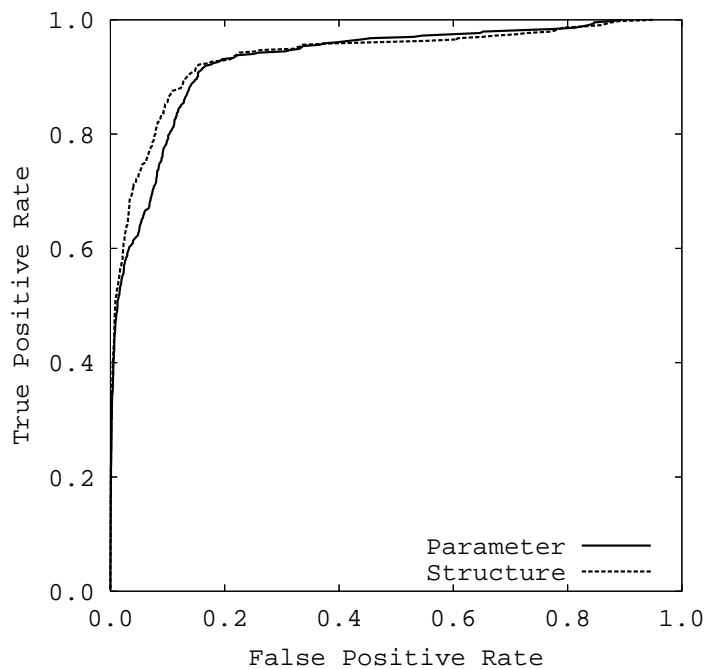


Figure 4.8 ROC Curves for Parameter Learning (Level 1) compared to Structure Learning (Level 2)

shows the precision-recall curves. Because of the skewed class distribution, due to the large number of benign cases, we prefer precision-recall curves over ROC curves because they better show the number of “false alarms,” or unnecessary biopsies. Therefore, we will use precision-recall curves for the remainder of the results. Here, precision is the percentage of abnormalities that we classified as malignant that are truly cancerous. Recall is the percentage of malignant abnormalities that were correctly classified. To generate the curves, we pool the results over all ten folds by treating each prediction as if it had been generated from the same model. We sort the estimates and use all possible split points to create the graphs.

Figure 4.10 compares performance for all levels of learning. Adding multi-relational features results in significant improvements. Aggregates provide the most benefit for higher recalls whereas rules help in the medium and low ranges of recall. We believe this is because ILP rules are more accurate than the other features, but have limited coverage.

Figure 4.11 shows the average area under the precision-recall (AUC-PR) curve, using TAN as the structure learner, for each level of learning defined in Figure 4.3. It only considers recalls

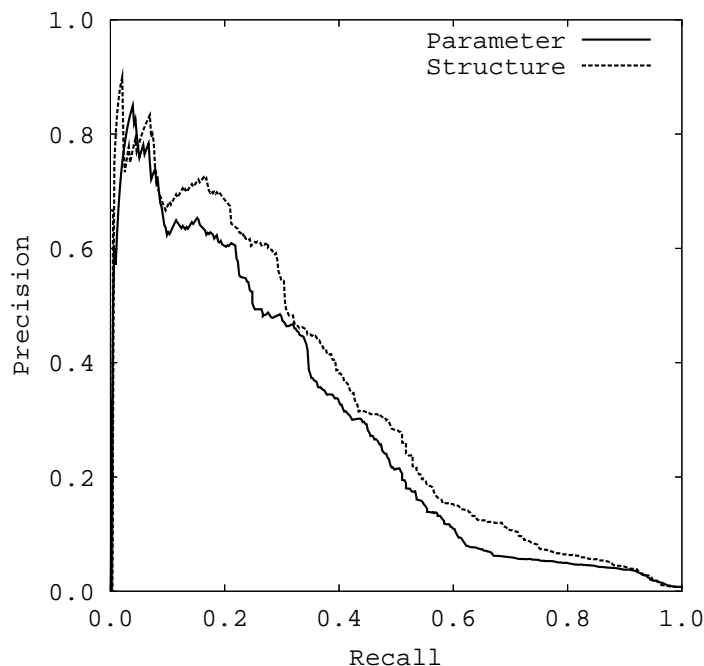


Figure 4.9 Precision-Recall Curves for Parameter Learning (Level 1) compared to Structure Learning (Level 2)

above 50%, as for this application radiologists would be required to perform at least at this level. We use a paired  $t$ -test to compare the areas under the curve (recall  $\geq 0.5$ ) for every fold. We found improvement of Level 2 over Level 1 to be statistically significant with  $p$ -value  $< 0.01$ . According to the paired  $t$ -test the improvement of Level 3 over Level 2 is also significant ( $p$ -value  $< 0.01$ ). Furthermore, Level 4 significantly improves over Level 2 with  $p$ -value  $< 0.01$ . However, there is no significant difference between Level 3 and Level 4.

Table 4.1 shows the results for all three structure learning algorithms evaluated. TAN clearly offers better performance than naïve Bayes and the sparse candidate algorithm. Naïve Bayes performs poorly for two reasons. First, the expert defined structure has a similar topology to naïve Bayes, thus making it difficult for naïve Bayes to offer improvement on the structure learning task. Second, adding aggregate and view features violates the naïve Bayes' independence assumption, which degrades performance.

We ran the sparse candidate algorithm with the following settings. We set the candidate parent set size to be five, we use Bayesian information criteria (BIC) as the scoring function and we start

| Structure Learning Algorithm | Parameter Learning | Structure Learning | Aggregate Learning | View Learning |
|------------------------------|--------------------|--------------------|--------------------|---------------|
| Naïve Bayes                  | 0.0428             | 0.0335             | 0.0257             | 0.0493        |
| TAN                          | 0.0428             | 0.0579             | 0.0743             | 0.0734        |
| Sparse Candidate             | 0.0428             | 0.0319             | 0.0324             | 0.0381        |

Table 4.1 Average AUC-PR for Recall  $\geq 0.5$  for Naïve Bayes, TAN and the Sparse Candidate Algorithm

with an empty network structure. The algorithms' poor performances likely indicates that BIC is not well-suited to finding a network topology that is good at predicting one variable, especially when starting from an empty network. When scoring the network structures with BIC, a more appropriate comparison is to start with a TAN network. Another alternative would be to use a discriminative scoring function such as conditional log likelihood.

Now we can evaluate the first two hypotheses. These results confirm the first conjecture, as in this task considering relational information is still crucial for improving performance. Both relational approaches outperform the propositional methods. The results partially support the second hypothesis. Moving up the learning hierarchy outlined in Figure 4.3 mostly results in significant improvement in performance.

## 4.9 Interesting Views

To evaluate the claim that the process of generating the views in Level 4 can be useful to the radiologist, we presented an expert mammographer (Elizabeth Burnside) with a set of 130 rules to review. She found several rules interesting, confirming the third hypothesis. The following rule is of particular interest:

Abnormality A in mammogram M may be malignant if:

A has BI-RADS category 5, and

A has a mass present, and

A has a mass with high density, and

P has a prior history of breast cancer, and

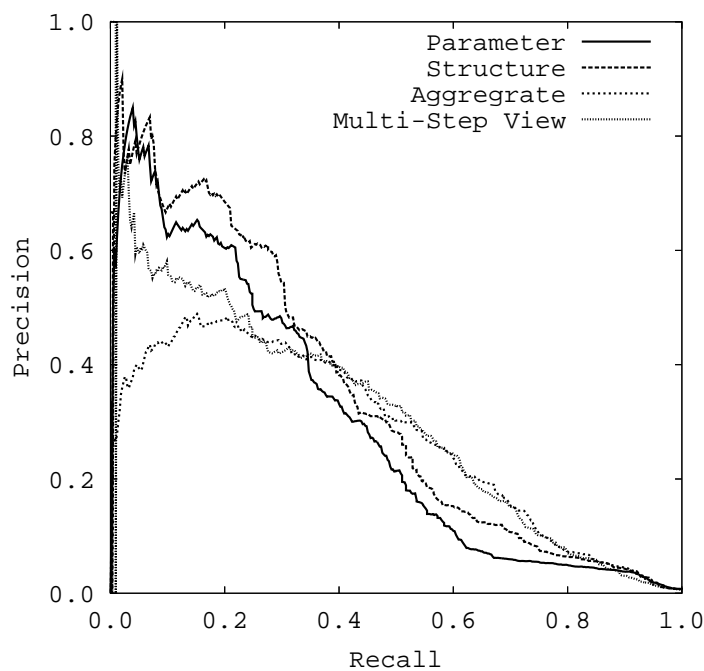


Figure 4.10 Precision-Recall Curves for Each Level of Learning

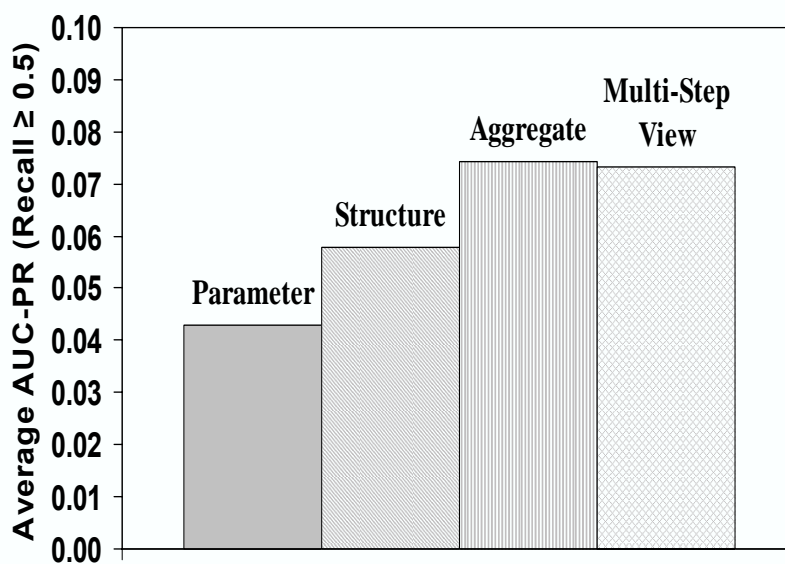


Figure 4.11 Average AUC-PR For Recalls  $\geq 0.5$

P has an extra finding on same mammogram (B), and  
 B has no pleomorphic microcalcifications, and

| Mass Density | Benign (%)  | Malignant (%) | Total |
|--------------|-------------|---------------|-------|
| Fat-density  | 493 (100.0) | 0 (0.0)       | 493   |
| Low          | 3406 (99.9) | 2 (0.1)       | 3408  |
| Equal        | 496 (96.7)  | 103 (31.7)    | 324   |
| High         | 221 (68.2)  | 103 (31.7)    | 324   |
| Total        | 4616 (97.4) | 122 (2.6)     | 4738  |

Table 4.2 Density of Benign vs Malignant Masses

B had no punctate calcifications.

This rule identified 42 malignant mammographic findings while only misclassifying 11 benign findings as cancer. The radiologist was intrigued by this rule because it suggests a hitherto unknown relationship between malignancy and high density masses. In order to analyze the second rule, the proportion of masses diagnosed as cancer that were high density was compared with the proportion of benign masses that were high density, as shown in Table 4.2. We found statistically significant differences in the rate of malignancy when comparing based on mass density. The fat density and low density descriptors were combined for this analysis (there was no statistically significant difference of malignancy in these two groups). Fisher's exact test (Fisher, 1922) demonstrated significant differences in the rate of malignancy for high density versus fat- or low-density masses, high density versus equal density masses, and equal density versus fat- or low-density masses ( $p < .001$  for all three comparisons). In general, mass density was not previously thought to be a highly predictive feature, so this rule is valuable in its own right (Burnside et al., 2005).

#### 4.10 Related Work

The work in this chapter is closely related to the propositionalization which is discussed in more depth in Section 3.5. Another important work in this category was by Srinivasan and King (1997). They combined rules learned by an ILP system with expert defined features for the task of predicting biological activities of molecules.

Much effort has been invested in investigating how to incorporate aggregate features within SRL. Relational Probability Trees (Neville et al., 2003) use aggregation to provide extra features

on a multi-relational setting, and are close to our Level 3 setting. Knobbe et al. (2001) proposed numeric aggregates in combination with logic-based feature construction into an algorithm for propositionalizing relational databases. Perlich and Provost (2003) discuss several approaches for attribute construction using aggregates over multi-relational features. The authors also propose a hierarchy of levels of learning: feature vectors, independent attributes on a table, multidimensional aggregation on a table, and aggregation across tables. Some of these techniques in their hierarchy could be applied to perform view learning in SRL. Vens et al. (2004) investigate how to introduce aggregates into first-order decision tree learning. They proposed using simple aggregates, which apply to a single literal within a clause. They investigated constructing complex aggregates, which allow aggregation over clauses that contain more than one literal.

## 4.11 Chapter Summary

This chapter introduces view learning for SRL. View learning helps overcome that fact that many current SRL algorithms cannot alter the schema of a database. We present a simple, multi-step algorithm for view learning that augments a database schema by introducing new fields. We use a standard ILP system to learn the field definitions. The resulting approach integrates learning from attributes, aggregates, and rules.

We introduce the problem of labeling abnormalities on mammogram as benign or malignant as an interesting and important task for SRL. We demonstrate that simple, propositional learning techniques can improve over an expert system for this task. We further show that SRL techniques significantly improve over propositional methods. Nevertheless, we found that the improvement obtained through the use of aggregation, as might be performed for example by a PRM, is roughly equivalent to view learning.

The process of generating the views can provide a benefit to users of a SRL system. The rules can help elucidate potentially interesting correlations between attributes. In our application, an expert mammographer discovered a relationship between malignancy and high density masses by looking at the learned views. In general, mass density was not previously thought to be a highly predictive feature, so this rule is valuable in its own right.

## Chapter 5

### SAYU: A Framework for Integrated View Invention and Model Construction

The previous chapter developed a multi-step process for learning new views. In the first step, an ILP algorithm learns a set of rules. The second step selects a relevant subset of rules for inclusion in the model. The third step constructs a statistical model, which includes the learned rules and the pre-existing features. This approach suffers from several weaknesses. First, the rule learning procedure is computationally expensive. Second, choosing how many rules to include in the final model is a difficult tradeoff between completeness and overfitting. Finally, the best rules according to coverage may not yield the most accurate classifier.

This chapter proposes an alternative approach, based on the idea of *constructing the classifier as we learn the rules*. The new approach scores rules by how much they improve the classifier, providing a tight coupling between rule generation and rule usage. We call this methodology *Score As You Use* or *SAYU*.

SAYU is closely related to Landwehr, Kersting and De Raedt's (2005) contemporary work nFOIL and is also related to Popescul et al.'s (2003) work on Structural Logistic Regression. The relationships to these important works are discussed in Section 5.7.

SAYU represents a general framework for dynamically constructing relational features for a propositional learner. In principle, SAYU could be implemented with any feature construction method and any propositional learning. This chapter will briefly develop the general SAYU framework and describe an implementation of SAYU for classification.

This work originally appeared in Davis et al. (2005a) and Davis et al. (2007a).

## 5.1 The SAYU Framework

The SAYU approach, starts from an empty model (or a prior model). Next, an ILP system generates rules, each of which represents a new feature to be added to the current model. SAYU evaluates each feature in the following manner. It extends the attributes available to propositional learner with the rule proposed by the ILP system. The propositional learner constructs a new model using the extended feature set. Next, it evaluates the generalization ability of the model extended with the new feature. SAYU retains the new model if the addition of the new feature improves the model's generalization ability; otherwise it keeps the original model. Figure 5.1 presents an overview of SAYU's process for inducing new features.

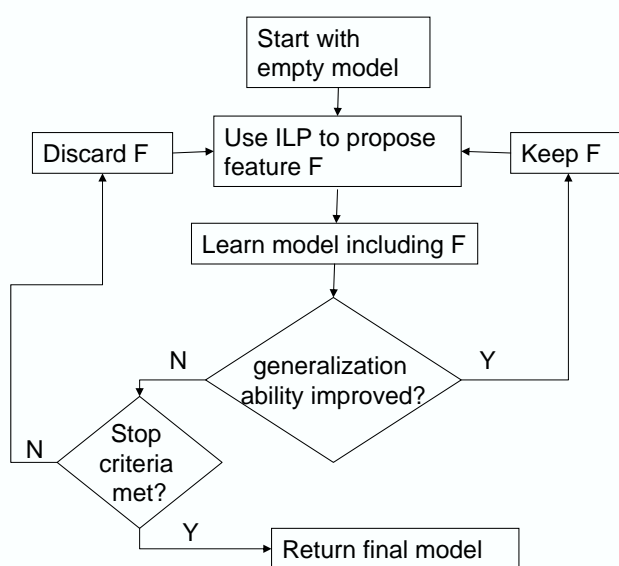


Figure 5.1 General SAYU Framework

## 5.2 SAYU with Bayesian Network Classifiers

The initial goal of the SAYU is to develop a classification systems. Our SAYU implementation uses the Aleph ILP system as a rule proposer and naïve Bayes or TAN as propositional learners. SAYU modifies the standard Aleph search as follows. Instead of using coverage to evaluate clauses, Aleph passes each clause it constructs to SAYU, which converts the clause to a binary

feature and adds it to the current training set. Next, SAYU learns a new model incorporating the new feature, and evaluates the model (described in the next paragraph). If the model does not improve, the rule is not accepted, and control returns to Aleph to construct the next clause. If a rule is accepted, or the search space is exhausted, SAYU randomly selects a new seed and re-initializes Aleph's search. Thus, it is not searching for the best rule, but the first rule that improves the model. However, SAYU allows the same seed to be selected multiple times during the search. Figure 5.2 shows several moves in SAYU's learning procedure.

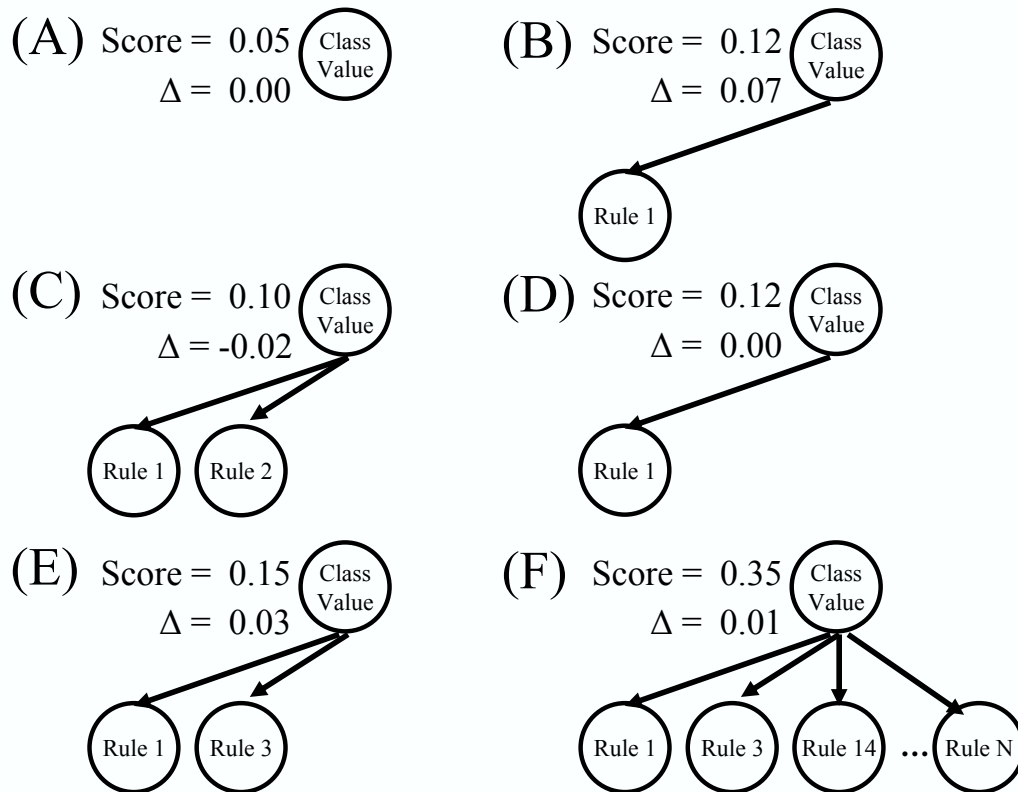


Figure 5.2 Network A shows the initial network structure. Network B shows the incorporation of the first rule, which is retained, as it improves the score. Network C shows the incorporation of the second rule, which is discarded, as it lowers the score. Network D shows the network after discarding the second rule. Network E shows the incorporation of the third rule, which is retained, as it improves the score. Network F shows the final network.

In order to decide whether to retain a candidate feature  $f$ , SAYU needs to estimate the generalization ability of the model with and without the new feature. SAYU does this by calculating

```

Input: Train Set Labels  $T$ , Tune Set Labels  $S$ , Background Knowledge  $B$ ,
Improvement Threshold  $p$ 
Output: Feature Set  $F$ , Statistical Model  $M$ 
/* Note the initial network will only have one feature: the
variable to predict */
 $F = \emptyset$ ;
 $M = \text{BuildBayesNet}(T, F)$ ;
 $BestScore = \text{AreaUnderPRCurve}(M, S, F)$ ;
while time remains do
   $done = \text{false}$ ; Choose a positive example as a seed and initialize Aleph
  search;
  repeat
     $f = \text{Generate new clause according to Aleph}$ ;
     $F_{new} = F \cup f$ ;
     $M_{new} = \text{BuildBayesNet}(T, F_{new})$ ;
     $NewScore = \text{AreaUnderPRCurve}(M, S, F_{new})$ ;
    /* Retain this feature */
    if ( $NewScore \geq (1 + p) * BestScore$ ) then
       $F = F_{new}$ ;
       $BestScore = NewScore$ ;
       $M = M_{new}$ ;
       $SelectedFeature = \text{true}$ ;
    end
    else
      /* revert back to old structure and continue
      searching */
    end
  until  $not(SelectedFeature)$ ;
end
return  $M, F$ 

```

**Algorithm 1:** SAYU Algorithm

the AUC-PR (Goadrich et al., 2004) on a tuning set. When calculating the AUC-PR, SAYU only includes recall levels of 0.2 or greater, as precision has high variation for low levels of recall. Algorithm 1 gives pseudo code for the SAYU algorithm.

### 5.3 Datasets

We evaluate our algorithm with four very different datasets. Two of the applications are relatively novel: the *Mammography* and *Yeast Protein* datasets. The third, *Carcinogenesis*, is well-known in the inductive logic programming community. The final domain is the University of Washington *UW-CSE* dataset that is a popular benchmark in Statistical Relational Learning (Richardson & Domingos, 2006; Singla & Domingos, 2005).

**Mammography.** The *Mammography* dataset is the original motivation of this work. See Section 4.4 for a more in depth discussion of the data and methodology for this dataset.

**Yeast Protein.** The second task consists of learning whether a yeast gene codes for a protein involved in the general function category of *metabolism*. The data for this task comes from the MIPS (Munich Information Center for Protein Sequence) Comprehensive Yeast Genome Database, as of February 2005 (Mewes et al., 2000). Positive and negative examples were obtained from the MIPS function category catalog. The positives are all proteins/genes that code for metabolism, according to the MIPS functional categorization. The negatives are all genes that have known functions in the MIPS function categorization and do not code for metabolism. Notice that the same gene may code for several different functions, or may code for different sub-functions. The dataset includes information on gene location, phenotype, protein class, and enzymes. It also uses gene-to-gene interaction and protein complex data. The dataset contains 1,299 positive examples and 5,456 negative examples. We randomly divide the data into ten folds. Each fold contains approximately the same number of positive and negative examples.

**Carcinogenesis.** The third dataset concerns the well-known problem of predicting carcinogenicity test outcomes on rodents (Srinivasan & King, 1997). This dataset has a number of attractive features: it is an important practical problem; the background knowledge consists of a number of non-determinate predicate definitions; experience suggests that a fairly large search space needs to be examined to obtain a good clause. The dataset contains 182 positive carcinogenicity tests and

148 negative tests. We randomly divide the data into ten folds. Each fold contains approximately the same number of positive and negative examples.

**UW-CSE.** The last dataset concerns learning whether one entity is advised by another entity, and comes from real data obtained by Richardson and Domingos (2006) from the University of Washington CS Department. The dataset contains 113 positive examples and 2,711 negative examples. Following the original authors, we divide the data in five folds, each one corresponding to a different group in the CS Department.

## 5.4 Experimental Setup and Results

**Methodology.** On the first three datasets we perform stratified, ten-fold cross validation in order to obtain significance results. On each round of cross validation, both approaches use five folds as a training set, four folds as a tuning set and one fold as a test set. Both approaches only saturate examples from the training set. Since the UW-CSE dataset only has five folds, the approaches use two folds for a training set and two folds as a tuning set.

**Experimental Parameters.** The communication between the Bayes net learner and the ILP algorithm is computationally expensive. The Bayes net algorithm might have to learn a new network topology and new parameters. Furthermore, it must perform inference to compute the score after incorporating a new feature. The SAYU algorithm is strictly more expensive than standard ILP as SAYU also has to prove whether a rule covers each example in order to create the new feature. To reflect the added cost, we use a time-based stop criteria for the new algorithm. In effect, we test whether, given an equal amount of CPU time, the multi-step approach or SAYU performs better.

To obtain a performance baseline we first ran a set of experiments that use the original multi-step process. In all experiments we use Aleph (Srinivasan, 2001) as the rule learning algorithm. First, we use Aleph running under `induce_cover` to learn a set of rules for each fold. `Induce_cover` implements a variant of Progol's MDIE greedy covering algorithm, where it does not discard previously covered examples when scoring a new clause. Second, we select rules using

a greedy algorithm, where we pick the rule with the highest  $m$ -estimate of precision such that it covers an unexplained training example. Third, we convert each rule into a binary feature for a naïve Bayes and TAN classifier. In the baseline experiments, we use both the training and tuning data to construct the classifier and learn its parameters. Furthermore, we record the CPU time that it took for each fold to run to completion. We use this time as the stop criteria for the corresponding fold when evaluating the integrated approach. To offset potential differences in computer speeds, we run all of the experiments for a given dataset on the same machine.

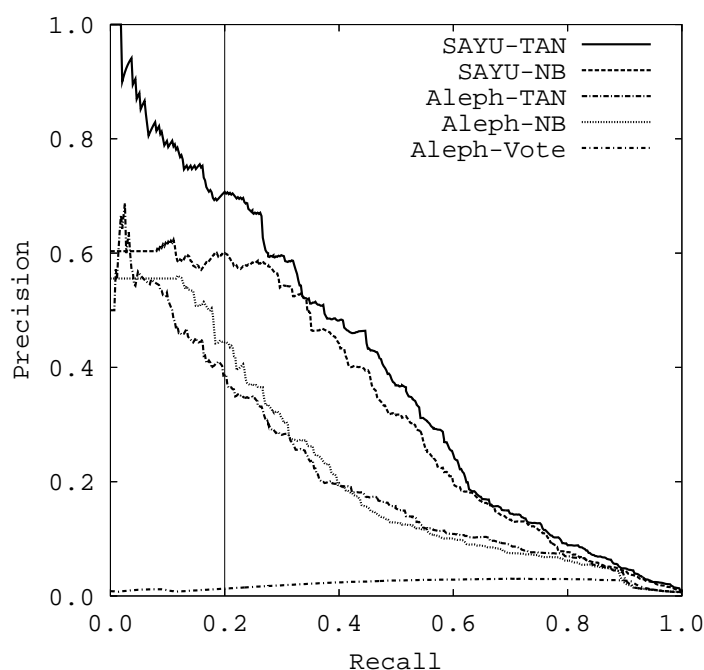


Figure 5.3 Mammography Precision-Recall Curves

SAYU only uses the training set to learn the rules. Additionally it uses the training set to learn the structure and parameters of the Bayes net, and it uses the tuning set to calculate the score of a network structure. Again, it uses Aleph to perform the clause saturation and propose candidate clauses to include in the Bayes Net. In order to retain a clause in the network, the AUC-PR of the Bayes net incorporating the rule must achieve at least a two percent improvement over the AUC-PR of prior Bayes net.

| Dataset        | Aleph-Vote | Aleph-NB | Aleph-TAN | SAYU-NB | SAYU-TAN |
|----------------|------------|----------|-----------|---------|----------|
| Mammography    | 0.0862     | 0.133    | 0.126     | 0.247   | 0.269    |
| Yeast Protein  | 0.269      | 0.333    | 0.357     | 0.397   | 0.400    |
| Carcinogenesis | 0.555      | 0.566    | 0.568     | 0.567   | 0.558    |
| UW-CSE         | 0.137      | 0.219    | 0.295     | 0.258   | 0.262    |

Table 5.1 Average AUC-PR for Recall  $\geq 0.2$ 

| Dataset        | $p$ -value<br>SAYU-TAN vs.<br>Aleph-Vote | $p$ -value<br>SAYU-TAN vs.<br>Aleph-NB | $p$ -value<br>SAYU-TAN vs.<br>Aleph-TAN | $p$ -value<br>SAYU-TAN vs.<br>SAYU-NB |
|----------------|--|--|---|---------------------------------------|
| Mammography    | $7.91 \times 10^{-6}$                    | $1.82 \times 10^{-4}$                  | $9.52 \times 10^{-5}$                   | 0.172                                 |
| Yeast Protein  | $2.59 \times 10^{-6}$                    | 0.00224                                | 0.0434                                  | 0.789                                 |
| Carcinogenesis | 0.905                                    | 0.790                                  | 0.748                                   | 0.677                                 |
| UW-CSE         | 0.00205                                  | 0.113                                  | 0.558                                   | 0.655                                 |

Table 5.2 Summary of  $p$ -values for SAYU-TAN vs. Other Algorithms

| Algorithm | Clauses in Theory | Number Predicates per Clause | Clauses Scored |
|-----------|-------------------|------------------------------|----------------|
| Aleph     | 99.6              | 2.82                         | 620000         |
| SAYU-NB   | 39.1              | 1.47                         | 85300          |
| SAYU-TAN  | 32.8              | 1.42                         | 20900          |

Table 5.3 Mammography. All metrics given are averages over all ten folds.

| Algorithm | Clauses in Theory | Number Predicates per Clause | Clauses Scored |
|-----------|-------------------|------------------------------|----------------|
| Aleph     | 170               | 2.93                         | 916000         |
| SAYU-NB   | 13.9              | 1.14                         | 190000         |
| SAYU-TAN  | 12.5              | 1.15                         | 132000         |

Table 5.4 Yeast Protein. All metrics given are averages over all ten folds.

| Algorithm | Clauses in Theory | Number Predicates per Clause | Clauses Scored |
|-----------|-------------------|------------------------------|----------------|
| Aleph     | 186               | 3.59                         | 3530000        |
| SAYU-NB   | 8.7               | 1.67                         | 875000         |
| SAYU-TAN  | 12.1              | 1.95                         | 679000         |

Table 5.5 Carcinogenesis. All metrics given are averages over all ten folds.

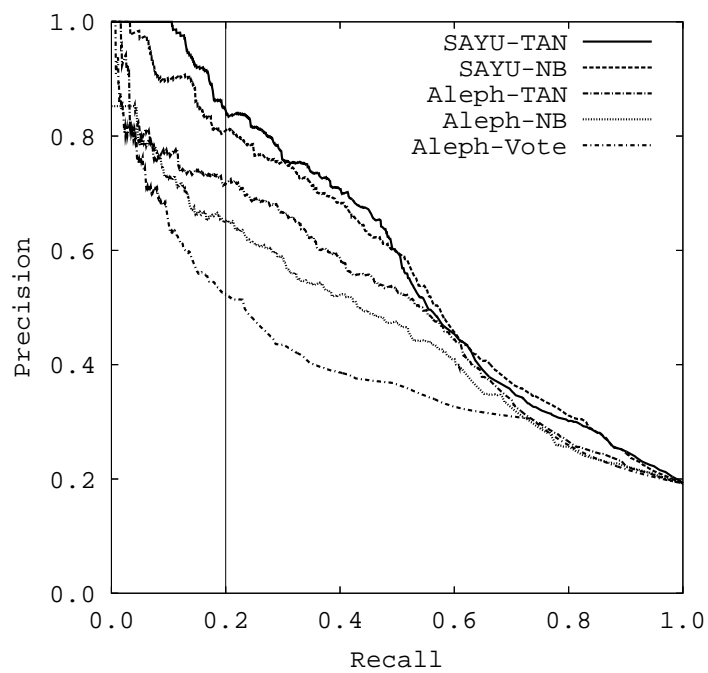


Figure 5.4 Yeast Protein Precision-Recall Curves

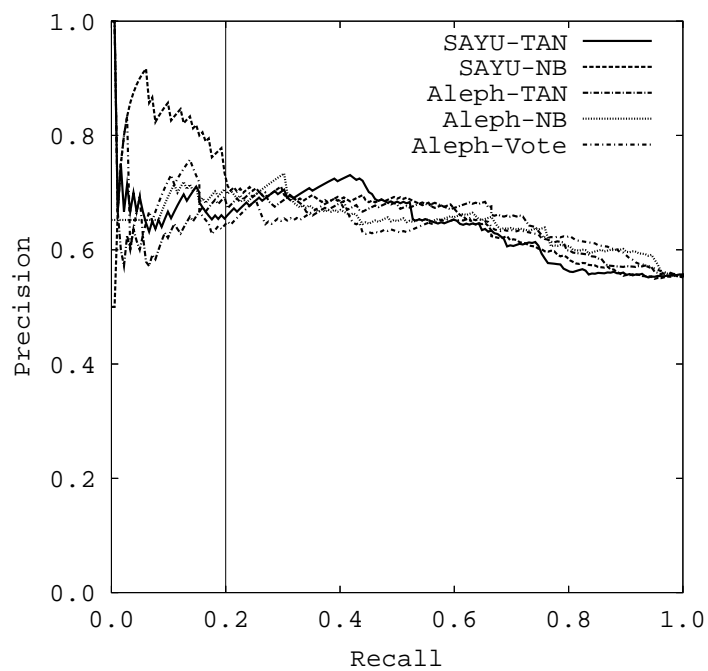


Figure 5.5 Carcinogenesis Precision-Recall Curves

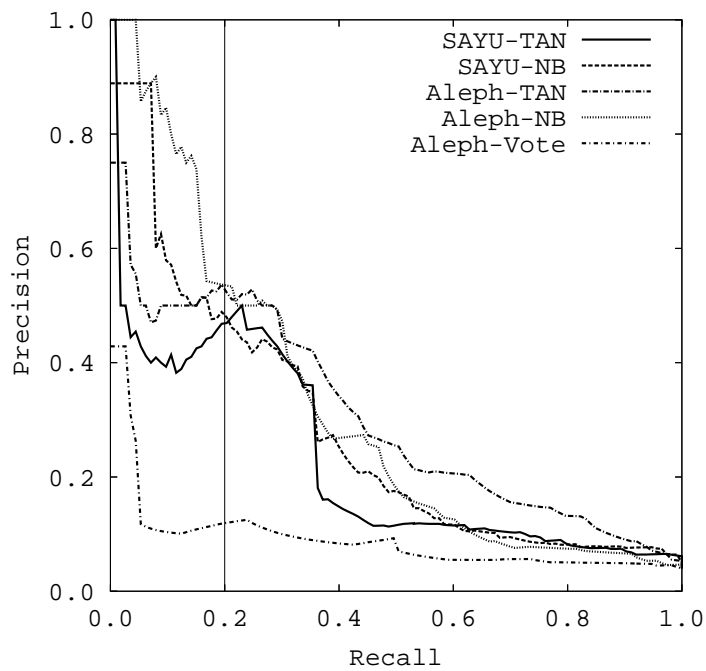


Figure 5.6 UW-CSE Precision-Recall Curves

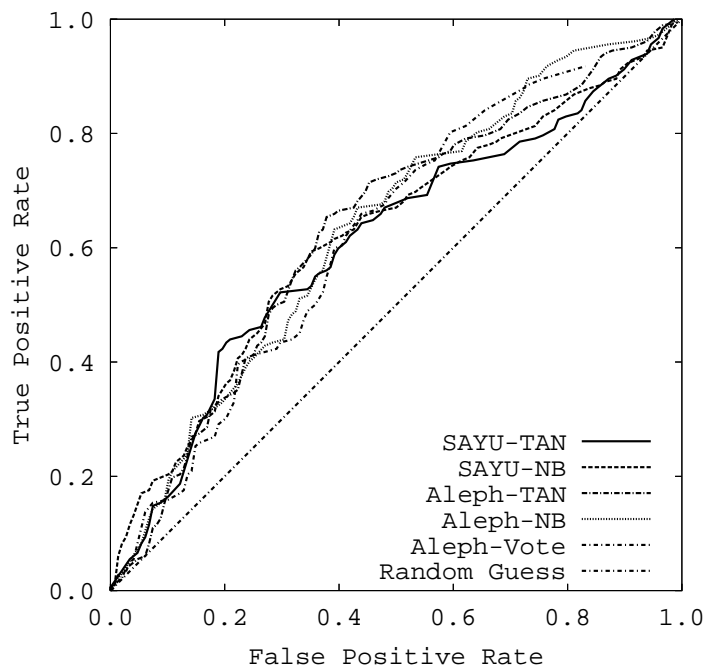


Figure 5.7 Carcinogenesis ROC Curves

| Algorithm | Clauses in Theory | Number Predicates per Clause | Clauses Scored |
|-----------|-------------------|------------------------------|----------------|
| Aleph     | 7.4               | 4.80                         | 180000         |
| SAYU-NB   | 13.0              | 2.96                         | 5320           |
| SAYU-TAN  | 13.6              | 2.68                         | 6050           |

Table 5.6 UW-CSE. All metrics given are averages over all ten folds.

In our experiments, we test three hypotheses. First, we hypothesize that using a Bayesian network to combine rules will produce more accurate results than using unweighted voting. Second, we hypothesize that the SAYU procedure results in features that are better suited than a feature construction criterion based on coverage, as standard Aleph uses. Third, we hypothesize that TAN will result in more accurate classification than naïve Bayes.

To test our hypotheses, we use five algorithms:

1. **Aleph-Vote**: This algorithm uses unweighed voting to combine rules and standard Aleph to construct and select features.
2. **Aleph-NB**: This algorithm uses naïve Bayes to combine rules and standard Aleph to construct and select features.
3. **Aleph-TAN**: This algorithm use TAN to combine rules and standard Aleph to construct and select features.
4. **SAYU-NB**: This algorithm uses naïve Bayes to combine rules and SAYU to select features.
5. **SAYU-TAN**: This algorithm uses TAN to combine rules and SAYU to select features.

**Results.** Figures 5.3 through 5.6 show precision-recall curves for all four datasets. For all the graphs, we generate the curves by pooling results over all folds. Table 5.1 shows the average AUC-PR for recall  $\geq 0.2$  for all algorithms. Table 5.2 gives the  $p$ -value, computed using a two-tailed, paired  $t$ -test comparing the AUC-PR (recall  $\geq 0.2$ ) for every fold, for SAYU-TAN versus the other four algorithms.

The Mammography dataset (Figure 5.3) shows a clear win for SAYU over the original multi-step methodology. We use the paired  $t$ -test to compare the areas under the curve for every fold, and found the difference to be statistically significant at the 99% level of confidence. The difference between using SAYU-TAN and SAYU-NB is not significant. The difference between using TAN and naïve Bayes to combine the Aleph learned rules is also not significant. Aleph-Vote is significantly outperformed,  $p$ -value  $< 0.05$ , by all the other approaches. Moreover, the results using SAYU match our best results on this dataset (Davis et al., 2005b), which had required more computational effort.

The Yeast Protein dataset (Figure 5.4) also shows a win for SAYU over the original multi-step methodology. The difference is not as striking as in the Mammography dataset, mostly because Aleph-TAN did very well on one of the folds. In this case Aleph-TAN is significantly better than Aleph-NB with 98% confidence. SAYU-TAN learning is significantly better than Aleph NB with 99% confidence, and Aleph-TAN with 95% confidence. SAYU-NB is better than Aleph-NB with 99% confidence. However, it is not significantly better than Aleph-TAN (only at 90% confidence), despite the fact that SAYU-NB beats it on nine out of ten folds. All other approaches significantly outperformed Aleph-Vote with  $p$ -value  $< 0.01$ .

The results for Carcinogenesis (Figure 5.5) are ambiguous: no method is significantly better than the other. One possible explanation is that precision-recall might not be an appropriate evaluation metric for this dataset. Unlike the other datasets, this one only has a small skew in the class distribution and there are more positive examples than negative examples. A more appropriate scoring function for this dataset might be the area under the ROC curve. We ran SAYU using this metric, which can be seen in Figure 5.7, and again found no difference between the integrated approach and the multi-step method. We believe an essential piece of future work is to run a simulation study to try to better discern the conditions under which the SAYU algorithm provides an advantage over the multi-step approach.

The results for the UW-CSE dataset (Figure 5.6) are also inconclusive. SAYU-TAN, SAYU-NB, Aleph-TAN and Aleph-NB all achieve roughly the same areas, and the differences among them are not significant. Aleph-Vote is significantly outperformed,  $p$ -value  $< 0.06$ , by all the

other approaches. However, it is interested to make a simple comparison to Markov Logic Networks (MLNs) on this dataset. Using the same folds for 5-fold cross-validation used in (Singla & Domingos, 2005), SAYU with either TAN or naïve Bayes achieves higher AUC-PR (recall  $\geq 0$ , than MLNs; specifically, SAYU-TAN achieves 0.414, SAYU-NB achieves 0.394, and MLN achieves 0.295 (taken from (Singla & Domingos, 2005)). We do not know whether the comparison with MLN is significant, because we do not have the per-fold numbers of MLN.

**Computational Cost of SAYU.** As discussed before, running SAYU is costly, as it needs to build a new propositional classifier when evaluating each rule. Moreover, the differences in scoring methods may lead to learning very different sets of clauses. Tables 5.3 through 5.6 display several statistics for the all four datasets. First, we look at the average number of clauses in a theory in the multi-step approach, and compare it with SAYU-NB and SAYU-TAN. Second, we compare average clause length, measured by the number of literals per clause body. Finally, we show the average number of clauses scored in each fold. Table 5.3 shows that SAYU’s theories contain far fewer clauses than the multi-step algorithm in Mammography. Moreover, it finds shorter clauses, some even with a single attribute. The two columns are very similar for SAYU-NB and SAYU-TAN. The last column shows that the cost of using SAYU is very high on this dataset: we only generate a tenth of the number of clauses when using naïve Bayes. Results for SAYU-TAN are even worse as it only generates 3% as many clauses as the original Aleph run. Even so, the SAYU-based algorithms perform better.

The Yeast dataset (Table 5.4) tells a similar story, as the SAYU-based approaches require fewer clauses to obtain a better result. Again, SAYU generates smaller clauses with the average clause length lower than for Mammography. The cost of implementing SAYU is less in this case. We believe this is because the cost of transporting the bitmaps (representing the new feature) through the Java-Prolog interface is smaller, since the dataset is not as large. Carcinogenesis (Table 5.5) again shows SAYU-based approaches learning smaller theories with shorter clauses, and paying a heavy price for interfacing with the propositional learning. Carcinogenesis is the smallest benchmark, so its cost is smaller than Mammography or Yeast Protein. Finally, on the UW-CSE dataset, SAYU

ends up with a model that includes more rules than Aleph. However, the rules are again shorter than those found by Aleph and it evaluates fewer rules than Aleph.

**Discussion.** The empirical results confirm our first hypothesis. We see a significant win through combining rules using a Bayesian network over unweighted-voting on all datasets except for carcinogenesis. In terms of our second hypothesis, using SAYU to select rules results in significant improvement on two domains. It is interesting to note that SAYU offers the improvement in performance on the two largest domains.

In three out of four datasets, the theory found by SAYU consists of significantly fewer and in all cases it results in shorter clauses. Even with the simpler classifier, SAYU does at least as well as the multi-step approach. Furthermore, SAYU achieves these benefits despite evaluating significantly fewer rules than Aleph.

All our results show no significant benefit from using SAYU-TAN over SAYU-NB, contradicting our third hypothesis. We believe there are two reasons for that. First, the SAYU algorithm itself might be searching for independent attributes for the classifier, especially when using SAYU-NB. Second, naïve Bayes is computationally more efficient, as the network topology is fixed. In fact, only the conditional probability table corresponding to the newly introduced rule must be built in order to evaluate the new rule. Thus, SAYU-NB benefits from considering more rules.

## 5.5 SAYU-View: Incorporating Initial Feature Sets

The previous section reported that SAYU performs on par with Level 3 and multi-step Level 4 from the previous chapter. However, in these experiments SAYU serves only as a rule combiner only, not as a tool for view learning that *adds* fields to the existing set of fields (features) in the database (Davis et al., 2005a). We modify SAYU to take advantage of the predefined features yielding a more integrated approach to View Learning. Here, we report on a more natural design, called SAYU-View, where SAYU starts from the Level 3 network. Algorithm 2 gives pseudo code for the SAYU-View algorithm.

```

Input: Train Set Labels  $T$ , Tune Set Labels  $S$ , Background Knowledge  $B$ ,
         Initial Feature Set  $F_{init}$ , Improvement Threshold  $p$ 
Output: Feature Set  $F$ , Statistical Model  $M$ 
 $F = F_{init}$ ;
/* This model will contain all the features in  $F_{init}$  in
   addition to the variable to predict */
 $M = \text{BuildBayesNet}(T, F)$ ;
 $BestScore = \text{AreaUnderPRCurve}(M, S, F)$ ;
while time remains do
   $done = \text{false}$ ; Choose a positive example as a seed and initialize Aleph
  search;
  repeat
     $f = \text{Generate new clause according to Aleph}$ ;
     $F_{new} = F \cup f$ ;
     $M_{new} = \text{BuildBayesNet}(T, F_{new})$ ;
     $NewScore = \text{AreaUnderPRCurve}(M, S, F_{new})$ ;
    /* Retain this feature */
    if ( $NewScore \geq (1 + p) * BestScore$ ) then
       $F = F_{new}$ ;
       $BestScore = NewScore$ ;
       $M = M_{new}$ ;
       $SelectedFeature = \text{true}$ ;
    end
    else
      /* revert back to old structure and continue
        searching */
    end
  until  $not(SelectedFeature)$ ;
end
return  $M, F$ 

```

**Algorithm 2:** SAYU-View Algorithm

## 5.6 Further Experiments and Results

We use essentially the same methodology as described previously for the initial approach to view learning. On each round of cross-validation, SAYU-View uses four folds as a training set, five folds as a tuning set and one fold as a test set. It only saturates examples from the training set. SAYU-View uses only the training set to learn the rules. The key difference between initial Level 4

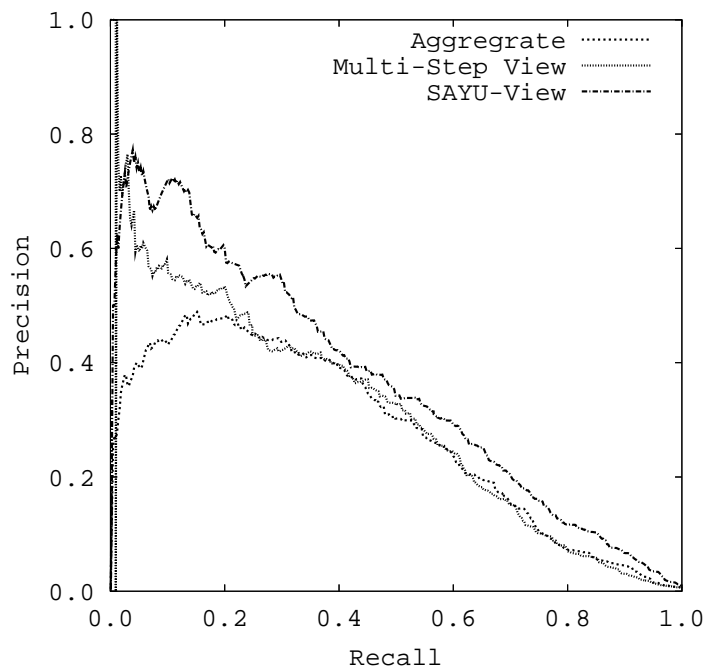


Figure 5.8 Precision-Recall Curves for Each Level of Learning

and SAYU-View is the following: for SAYU-View uses the training set to learn the structure and parameters of the Bayes net, and the tuning set to calculate the score of a network structure. The multi-step approach uses the tune set to learn the network structure and parameters.

In order to retain a clause in the network, the area under the precision-recall curve of the Bayes net incorporating the rule must achieve at least a two percent improvement over the area of the precision-recall curve of the best Bayes net. The main goal is to use the same scoring function for both learning and evaluation, so we use area under the precision-recall curve as the score metric. As in the previous chapter, the area under the precision-recall curve metric integrates over recall levels of 0.5 or greater.

As mentioned previously (see Section 5.4), the time required to score a rule using SAYU has increased. To reflect the added cost, SAYU-View uses a time-based stop criteria. For each fold, we use the times from the baseline experiments in (Davis et al., 2005a). In practice, these settings resulted in evaluating around 20000 clauses for each fold, requiring on average around 4 hours per fold on a Xeon 3MHz class machine.

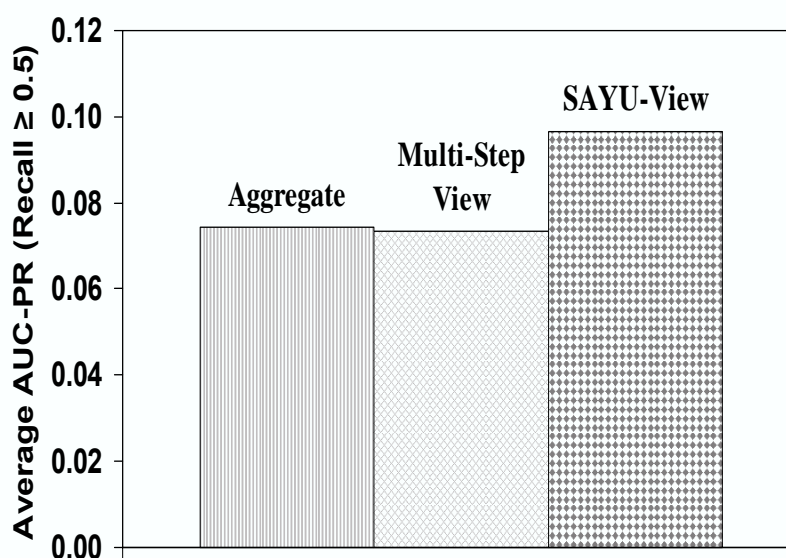


Figure 5.9 Average Area Under the Curve For Recalls  $\geq 0.5$  on Mammography Dataset

Figure 5.8 includes a comparison of SAYU-View to aggregate learning (Level 3) and the multi-step algorithm view learning (Level 4). Again, we perform a two tailed paired  $t$ -test on the AUC-PR for levels of recall  $\geq 0.5$ . SAYU-View performs significantly better than both these approaches with  $p$ -value  $< 0.01$ . Although not included in the graph, SAYU-View performs better than the SAYU-TAN (no initial features), though the difference is not significant ( $p$ -value  $< 0.18$ ). SAYU-View also performs better than Level 1 and Level 2 with a  $p$ -value  $< 0.01$ . The integrated framework for Level 4 now yields a significant improvement over all the lower levels of learning defined in the hierarchy in Figure 4.3.

Figure 5.9 shows the average AUC-PR for levels of recall  $\geq 0.5$  for Level 3, the initial approach to Level 4, and SAYU-View. The average AUC-PR for SAYU-View yields a 31% increase in the average AUC-PR over the initial approach to Level 4. Furthermore, it results in an increase in the average AUC-PR of 30% over Level 3.

## 5.7 Related Work

The most closely related to ours, (done in parallel with SAYU) is by Landwehr, Kersting and De Raedt (2005). That work presented a new system called nFOIL. The significant differences in the two pieces of work appear to be the following. First, nFOIL scores clauses by conditional log likelihood rather than improvement in classifier accuracy or classifier AUC (area under ROC or PR curve). Second, nFOIL can handle multiple-class classification tasks, which SAYU cannot. Third, the work with SAYU reports experiments on data sets with significant class skew, to which probabilistic classifiers are often sensitive. Finally, both papers cite earlier work showing that TAN outperformed naïve Bayes for rule combination (Davis et al., 2004). However, the empirical results with SAYU show that once clauses are scored as they are actually used, the advantage of TAN seems to disappear. More specifically, TAN no longer significantly outperforms naïve Bayes. Hence the present chapter may be seen as providing some justification for the decision of Landwehr et al. to focus on naïve Bayes. More recently, Landwehr et al. (2006) have proposed the kFOIL system. This system follows the same idea as nFOIL, except it uses a simple relational kernel as the statistical model. The use of a kernel permits the kFOIL system to handle regression tasks.

The work by Popescul and Ungar (2003) on structural logistic regression also integrates feature generation and model selection. Their work constructs features that can be used by a statistical learning algorithm. Feature definitions are in SQL queries and the statistical learner is logistic regression, but these are not especially important differences. This work also allows for aggregate functions to appear in the feature definitions. The drawback to this approach is that it is extremely computationally expensive. In fact, they report only searching over queries that contain at most two relations. In ILP, this would be equivalent to only evaluating clauses that contain at most two literals.

## 5.8 Chapter Summary

This chapter proposes an integrated framework, called SAYU, for constructing relational features and building statistical models. This framework alleviates a major drawback to the multi-step

approach described in Chapter 4: by only selecting rules that improve the performance of the statistical classifier that is being constructed. Finally, we present SAYU-View, a simple extension that allows SAYU to begin with an initial feature set.

We discuss an implementation of the SAYU framework that uses Bayesian networks as the underlying classifier. We empirically demonstrate that SAYU leads to more accurate models than the multi-step methodology. SAYU also constructs smaller theories than the multi-step approach. We found that SAYU-View results in significantly more accurate models on the mammography domain. Specifically, SAYU-View performs better than an SRL approach that only uses aggregation. SAYU's empirical success is quite surprising, as at first glance it would appear to be too expensive to build a new statistical classifier for each rule scored. SAYU overcomes the computation cost in two ways. First, it uses simple statistical models. Second, it is able to find small rule sets containing short rules that perform well. SAYU can find these theories with very little search.

## Chapter 6

### Combining SAYU and Multiple-Instance Regression for Drug Activity Prediction

The previous three chapters have shown the effectiveness of combining learned relational rules using a statistical classifier. SAYU, the most successful of these approaches, actually scores candidate relational rules by their ability to improve the statistical classifier. The present chapter extends the SAYU approach to multiple-instance, real-valued prediction. While such prediction is useful for many applications, our motivating application is drug design, specifically, predicting the real-valued binding affinities of small molecules for a fixed target protein. This chapter makes two contributions. First it demonstrates the feasibility and utility of extending SAYU to other types of tasks, such as multiple-instance, real-valued prediction, using a real-world application of significance for society and for the machine learning community in particular. Second, the chapter shows empirically that using multiple-instance regression in this context carries significant benefit over using ordinary regression.

This work originally appeared in Davis et al. (2007b).

#### 6.1 Background and Related Work

Drugs are small molecules that affect disease by binding to a target protein in the human body; the target may be a human protein or a protein belonging to some pathogen that has entered the body. Machine learning is potentially useful in a number of steps in the drug design process: for example, analysis of gene-expression microarray data or proteomics data using machine learning

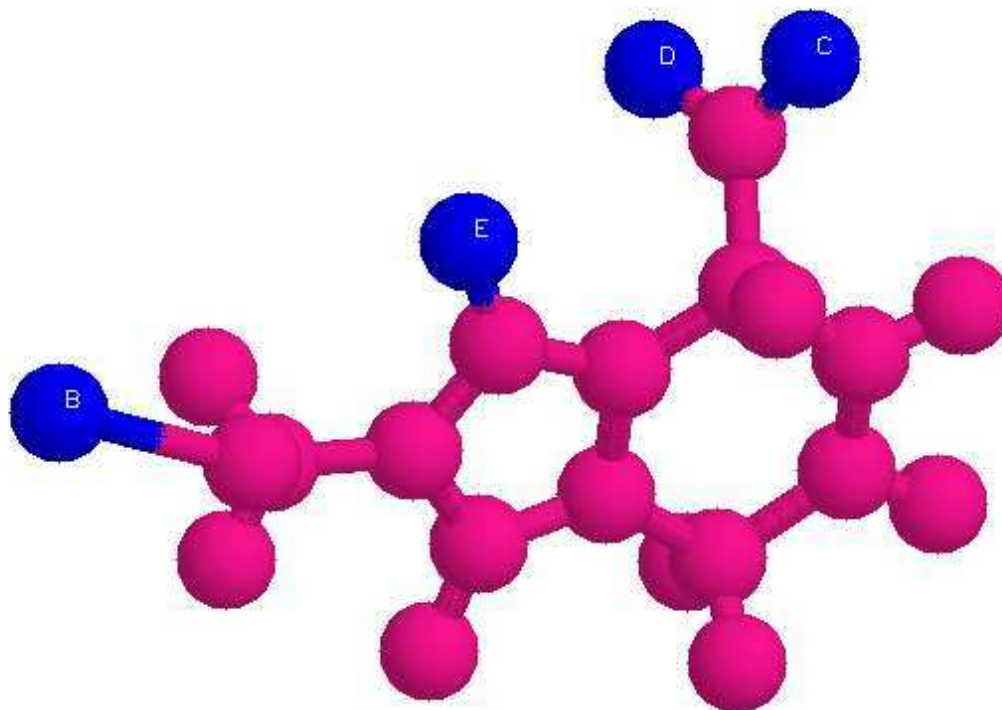


Figure 6.1 ACE inhibitor with highlighted 4-point pharmacophore. The point B is a zinc-binding site. Points C,D,E are hydrogen acceptors.

may help to identify target proteins; analysis of data from X-ray crystallography or nuclear magnetic resonance spectra may help determine the structure of a target; analysis of the structures of active molecules—those that bind to a target—may guide in the design of molecules with improved activity.

The focus of this chapter is on the last of these machine learning applications, which is known as predicting Three-dimensional Quantitative Structure-Activity Relationships (abbreviated as 3D-QSARs). Each 3D-QSAR task is defined by a target protein, typically whose 3D structure is not known. Given the structures of a set of molecules and their known binding affinities to the target, the task is to construct a model that accurately predicts the real-valued binding affinities of new small molecules to the target, based on their three-dimensional structures.

Prediction of structure-activity relationships is a particularly significant application for machine learning for at least two reasons. First, the potential benefit for society is large: the present

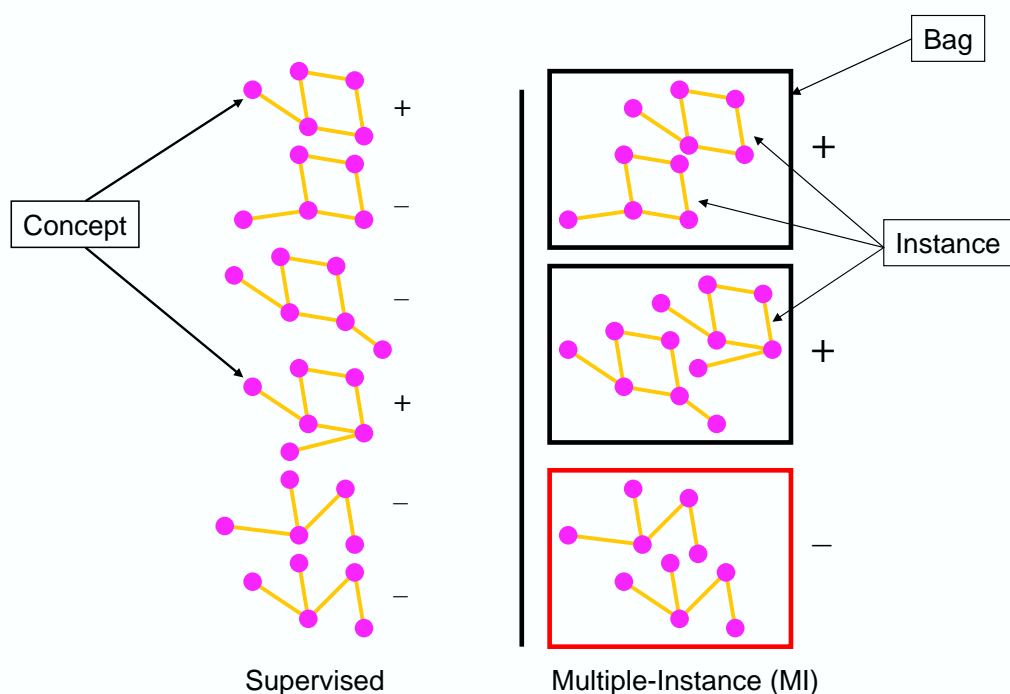


Figure 6.2 Contrasting Supervised and Multiple-Instance Learning

drug development cycle is time-consuming and expensive, estimated at close to one billion U.S. dollars per drug that makes it to market (DiMasi et al., 2003; Adams & Brantner, 2004). Second, the application has demonstrated potential to motivate important advances in machine learning; it has motivated the paradigm of multiple instance learning (Dietterich et al., 1997) and one of the first uses of relational learning algorithms as feature constructors (King, 1991). While those contributions came from the classification task, predicting real-valued binding affinities is perhaps an even more important task. Chemists prefer models that predict real-valued binding affinity, which can be used to identify molecules that are likely to have particularly high binding affinities.

A key obstacle in the drug design process is that the structure of a target protein cannot often be determined. In this scenario, researchers may use “high-throughput screening” to test a large number of small molecules to find some that bind to the target. The molecules that bind usually cannot be used as drugs for various reasons, typically related to ADMET (absorption, distribution, metabolism, elimination and toxicity). Nevertheless, some of these “hits” may be used as a “lead”

in the search for an appropriate drug; this lead then needs to be “optimized,” or modified in order to have an appropriate binding affinity and the other properties necessary in a drug, such as low side effects.

|  |   |
|--|---|
| <pre> active(M):-   conf(M, C),   hacc(M, C, P1),   hacc(M, C, P2),   hacc(M, C, P3),   pos_charge(M, C, P4),   dist(M, C, P1, P2, 4.60, 1.0),   dist(M, C, P1, P3, 7.75, 1.0),   dist(M, C, P2, P3, 8.77, 1.0),   dist(M, C, P1, P4, 6.85, 1.0),   dist(M, C, P2, P4, 7.56, 1.0),   dist(M, C, P3, P4, 1.24, 1.0). </pre> | <p>Molecule <math>M</math> is active if</p> <ul style="list-style-type: none"> <li><math>M</math> has a conformation <math>C</math></li> <li><math>C</math> has a hydrogen acceptor at location <math>P1</math></li> <li><math>C</math> has a hydrogen acceptor at location <math>P2</math></li> <li><math>C</math> has a hydrogen acceptor at location <math>P3</math></li> <li><math>C</math> has a positively charged group at location <math>P4</math></li> <li>the distance between <math>P1</math> and <math>P2</math> is <math>4.60 \pm 1.0 \text{ \AA}</math></li> <li>the distance between <math>P1</math> and <math>P3</math> is <math>7.75 \pm 1.0 \text{ \AA}</math></li> <li>the distance between <math>P2</math> and <math>P3</math> is <math>8.77 \pm 1.0 \text{ \AA}</math></li> <li>the distance between <math>P1</math> and <math>P4</math> is <math>6.85 \pm 1.0 \text{ \AA}</math></li> <li>the distance between <math>P2</math> and <math>P4</math> is <math>7.56 \pm 1.0 \text{ \AA}</math></li> <li>the distance between <math>P3</math> and <math>P4</math> is <math>1.24 \pm 1.0 \text{ \AA}</math></li> </ul> |
|--|---|

Table 6.1 An example of a 4-point pharmacophore learned by Aleph for the domain of thermolysin inhibitors. The left column shows the first-order logical clause in Prolog notation, while the right column shows the semantics of each literal.

To guide the lead optimization process, researchers try to find similarities between the most active molecules that are ideally not shared by any of the less active molecules. We briefly describe the types of similarities that are useful in predicting binding affinity. A small molecule binds to a protein primarily based on electrostatic and hydrophobic interactions. The most common electrostatic interaction is the hydrogen bond, where an atom carrying a slight negative charge, such as an oxygen (a “hydrogen acceptor”), on one molecule is attracted to a hydrogen atom carrying a slight positive charge (a “hydrogen donor”) on the other molecule. Hydrophobic interactions typically occur when hydrophobes from the two molecules shield each other from the surrounding aqueous environment. Because both electrostatic and hydrophobic interactions are weaker than the ordinary covalent bonds formed within a molecule, several such interactions—typically three to eight—are required in order for a small molecule to bind to a protein. Therefore, to bind to a given target protein at a particular site, a small molecule needs the right combination of charged atoms and/or hydrophobic groups *at the right locations*. In other words, the binding sites on the small molecule and protein need to be complementary, much as a key is to a lock—a common analogy

in drug design. Given a set of active molecules, a computational chemist may search for conformers of the active molecules that share some three-dimensional arrangement of charged atoms, such as potential hydrogen donors and acceptors, and hydrophobic groups, such as six-membered carbon rings. This three-dimensional substructure is sometimes called a *pharmacophore*. Figure 6.1 shows an example molecule with a highlighted pharmacophore that allows it to inhibit Angiotensin-Converting Enzyme (ACE).

One challenge arises from the fact that each molecule may have multiple low-energy 3D shapes, or *conformers*, and any one of these conformers may be the one that binds to the target. 3D-QSAR approaches directly address the multiple 3D conformers of molecules. These approaches include both special-purpose algorithms for 3D-QSAR and mechanisms for applying machine learning algorithms to the multiple 3D conformers of molecules. CoMFA and related approaches rely on careful feature construction based on structural properties at grid points defined on the molecule's surface (for example, see Cramer et al. (1988)). DISCO (Martin et al., 1993) uses a clique detection algorithm (Brint & Willett, 1987) to predict pharmacophores from the conformers of active molecules. The COMPASS algorithm (Jain et al., 1994a; Jain et al., 1994b) selects and aligns conformers—one per active molecule—and generates a feature vector for each molecule, where the features are the lengths of rays passing through the molecule at specified orientations. COMPASS then uses a neural network to learn either a classifier or real-valued predictor of affinities; once the model has been learned, COMPASS tries to improve the fit by revisiting the selection and alignment of conformers, and iterates until convergence.

Another approach to predicting binding affinities is based on relational learning (Finn et al., 1998; Marchand-Geneste et al., 2002), in particular inductive logic programming (ILP). This ILP-based framework starts with a first-order logical description of each molecule. This description details the locations of each atom and bond in the molecule. Additionally, the background knowledge contains relational descriptions of common groups of atoms. For example, the background knowledge can specify that a *methyl* group consists of a carbon atom bound to three hydrogen atoms with single bonds. A *k-point pharmacophore* in this representation is a first-order clause that has  $k$  literals, each describing a distinct chemical group (such as *methyl*), and  $\binom{k}{2}$  “distance”

literals. Each distance literal stores the Euclidean distance between two chemical groups. Since the distances in any two given molecules are unlikely to be exactly the same, the literal includes a tolerance that specifies how much each distance is allowed to vary. Given this representation, the approach uses an ILP system to hypothesize pharmacophores that cause the desired interaction between known active molecules and the target. The ILP system searches over the space of clauses (pharmacophores) using an objective function such as the following: any  $k$ -point pharmacophore that appears significantly more often in active molecules than in inactive ones is hypothesized to be an interaction-causing pharmacophore. Table 6.1 shows an example pharmacophore learned by the Aleph ILP system for the domain of thermolysin inhibitors.

In order to predict real-valued activities, the ILP-based approach treats learned clauses as binary-valued features and generates a binary (0/1) value depending on whether that molecule satisfies the given clause (i.e., has the specified pharmacophore in *any* conformation). Of course, using this representation, the inactive (or “less active”) molecules will have features that are mostly zero, which will likely lead to poor activity estimates. Thus, the ILP-based approach also learns a set of features that are more frequent in the inactive molecules than in the active molecules, and generates the corresponding feature vectors. This procedure generates a single feature vector for each molecule. This representation can then be used to learn a regression model using standard linear regression (Marchand-Geneste et al., 2002), which can predict activity levels for novel molecules.

Marchand-Geneste et al. (2002) found that this approach improves performance over CoMFA (Cramer et al., 1988). Despite its empirical success, this approach still has two shortcomings. First, even though the goal is to optimize the accuracy of the real-valued prediction, the relational learning procedure is not guided by this goal but rather by a different scoring function that is usually based on the coverage of the rules on the training set. Second, by operating on one feature vector per molecule, regression ignores the inherent multiple-instance nature of 3D molecular data (Dietterich et al., 1997). Information about the individual conformers is lost when we construct one feature vector per molecule.

## 6.2 The MIR-SAYU Algorithm

Our algorithm follows the ILP-based approach just described, but with two significant changes. First, it uses the “Score As You Use” method to learn rules directly judged as helpful to regression. For each potential pharmacophore, or rule, that must be scored, we recompute the regression model and check whether it generalizes better than the model that does not use the candidate rule. Thus, a pharmacophore is included in the regression model only if it helps to predict the observed activities. Second, we replace standard regression with multiple-instance regression (Ray & Page, 2001) when predicting activity.

In supervised learning, each example consists of a feature vector and a label. Multiple-instance learning generalizes supervised learning by representing examples with *multisets*, or bags, of feature vectors. It associates each bag with a class label in a classification setting or a real-valued response in a regression setting. Figure 6.2 contrasts supervised learning with multiple-instance learning. Therefore, when we evaluate a set of rules, we construct one feature vector per conformer rather than one feature vector per molecule. In multiple-instance terminology, a molecule is a bag, and each instance in the bag is a feature vector representing a conformer of that molecule. Multiple-instance regression operates on data of this form, with one real-valued response per bag, which is the activity level of the molecule. In the following sections, we describe each component of this approach in detail.

### 6.2.1 Scoring Candidate Rules with SAYU

In relational approaches to 3D-QSAR, as described above, an ILP system generates rules describing pharmacophores. In prior work (Finn et al., 1998; Marchand-Geneste et al., 2002), this system runs to completion, and a subset of the rules found are used to build the model. This approach relies on the ILP system’s score metric to evaluate rule quality. The most common metric is *coverage*, which is defined as the difference between the number of active and inactive molecules that satisfy a rule. The final model is built from the rules which have the highest coverages. This approach has several drawbacks. First, running to completion may take a long time. Second, the

rules may not be independent, leading to a model with dependent attributes. Third, choosing how many rules to include in the final model is a difficult tradeoff between completeness and overfitting. Finally, the best rules according to coverage may not give us the most accurate activity model.

Many of these drawbacks can be overcome by interleaving the rule learning and model building processes. In our work, we accomplish this interleaving by extending the SAYU approach (Davis et al., 2005a) to the MI regression setting. In the SAYU approach, we start from an empty model (or a prior model). Next, an ILP system generates rules, each of which represents a new feature to be added to the current model. We then evaluate the generalization ability of the model extended with the new feature. We retain the new model if the addition of the new feature improves the model's generalization ability; otherwise we remain with the original model. This results in a tight coupling between feature construction and model building.

To apply SAYU to our task, we need an ILP system to propose rules. In our work, we use Aleph, which implements the Progol algorithm (Muggleton, 1995) to learn rules. This algorithm induces rules in two steps. Initially, it selects a positive instance to serve as the “seed” example. It then identifies all the facts known to be true about the seed example. The combination of these facts forms the example's most specific or saturated clause. The key insight of the Progol algorithm is that some of these facts explain this example's classification. Thus, generalizations of those facts could apply to other examples. Aleph therefore performs a general-to-specific search over the set of rules that generalize a seed example's saturated clause. Thus, in our application, Aleph picks an “active” molecule, and generates potential  $k$ -point pharmacophores from it. It continues until either finding a potential pharmacophore that has high coverage or the search space is exhausted. In the latter case, the search restarts with a different seed.

SAYU modifies the standard Aleph search as follows. In contrast to Aleph, SAYU allows any example, positive or negative, to be selected as a seed, because it is possible for the generalization of any example to improve the final regression model. Instead of using coverage, Aleph passes each clause it constructs to SAYU, which converts the clause to a binary feature and adds it to the current training set. Next, SAYU learns a model incorporating the new feature, and evaluates the model (described below). If the model does not improve, the rule is not accepted, and control

returns to Aleph to construct the next clause. If a rule is accepted, or the search space is exhausted, SAYU randomly selects a new seed and re-initializes Aleph’s search. Thus, we are not searching for the best rule, but the first rule that improves the model. However, SAYU allows the same seed to be selected multiple times during the search. Since the search space is extremely large, it is impractical to search it exhaustively. Further, this may lead to overfitting. Therefore, as in prior work (Davis et al., 2005a), we terminate the search after a certain amount of time. The pseudo code for MIR-SAYU can be found in Algorithm 3.

In order to decide whether to retain a candidate feature  $f$ , we need to estimate the generalization ability of the model with and without the new feature. In our work, we do this by estimating the test-set  $r^2$  of each model, defined as:

$$\text{Test-set } r^2 = 1 - \frac{\sum_i (Y_i - p_i)^2}{\sum_i (Y_i - a_i)^2}, \quad (6.1)$$

where  $i$  ranges over test examples,  $Y_i$  denotes the true response of the  $i^{\text{th}}$  test example,  $p_i$  denotes the predicted response of the  $i^{\text{th}}$  test example using our model, and  $a_i$  denotes the average response on the training set. Thus,  $r^2$  measures the improvement in squared error obtained by using our model over a baseline constant prediction. Observe that if  $p_i = a_i$ ,  $r^2 = 0$ , and if  $p_i = Y_i$ ,  $r^2 = 1$ . Thus, a higher test-set  $r^2$  indicates a model with better generalization ability. Note though that unlike ordinary  $r^2$ , it is possible for test-set  $r^2$  to be negative, since predictions are made on novel data points. To estimate test-set  $r^2$  for our models, we use *internal n-fold cross validation* on our training set. In turn, we hold out one fold and learn a model using the remaining folds. We use the model to make predictions on the held-out data. At the end of this procedure, we have a set of predictions for each held-out fold. We then pool these predictions across all folds and calculate the test-set  $r^2$  metric for the model containing  $f$  over the full set of predictions. To decide whether to retain the candidate feature, we stipulate that the test-set  $r^2$  of the model with  $f$  must improve over the model without  $f$  by a certain fraction,  $p$ , which we call the improvement threshold.<sup>1</sup> While such cross-validation is computationally expensive, we have observed that it significantly

---

<sup>1</sup>A more principled solution might be to use a statistical hypothesis test between estimates of the test-set  $r^2$  measures of the two models. We have tried this; however, since we generally have very small samples in our experiments, we did not obtain consistent results with this approach.

```

Input: Train Set  $T$ , Number of Folds  $n$ , Improvement Threshold  $p$ , Stop
Criteria
Output: Multiple-Instance Regression Model  $M$ , Feature Set  $F$ 
Divide  $T$  into  $n$  folds ;
Let  $T_i$  = train data for fold  $i$  ;
Let  $S_i$  = score set for fold  $i$  ;
 $F = \emptyset$ ;
while stop criteria not met do
     $BestScore = 0$ ;
     $selectedFeature = false$  ;
    Choose an example as a seed and initialize Aleph search ;
    repeat
         $f =$  Generate new clause according to Aleph;
         $Predictions = \emptyset$ ;
        /* for each fold */
        for ( $i = 0; i < n; i++$ ) do
            /* build a new model incorporating  $f$  */
             $M = BuildMIRModel(T_i, F \cup f)$ ;
            /* collect the predictions the new model makes on
            the score set */
             $Predictions = Predictions \cup PredictOutput(M, S_i, F \cup f)$  ;
        end
         $NewScore = r^2(Predictions)$ ;
        /* Check whether we should retain this feature in the
        model */
        if ( $NewScore \geq (1 + p) * BestScore$ ) then
            /* update the final model, which is built on the
            whole train set */
             $F = F \cup f$ ;
             $M_{final} = BuildMIRModel(T, F)$ ;
             $BestScore = NewScore$ ;
            /* break out of the repeat-until loop to select a
            new seed example */
             $selectedFeature = true$ ;
        end
        else
            /* revert back to old structure and continue
            searching */
        end
    until not( $selectedFeature$ );
end
return  $M_{final}, F$ 

```

**Algorithm 3:** MIR-SAYU Algorithm

improves the quality of the features added to our models and reduces overfitting to the training set. Further, since we impose an external time constraint on SAYU as described in the previous paragraph, this procedure does not slow down our empirical evaluation. After a set of features have been selected using the cross-validation procedure, we learn the final model, which incorporates all selected features, using the entire training set. We use this model to make predictions on unseen examples. This procedure prevents features that do not help predict activity from being added to the regression model. However, note that it may add features that help explain low activity. Such features will be associated with negative coefficients in the regression model.

## 6.2.2 Predicting Activity with MI Regression

In this section, we present our multiple-instance regression model that we use to predict the activity of molecules. These are the models that the SAYU procedure constructs when evaluating candidate features.

The relational learning procedure described in Section 6.1 results in a single feature vector describing each molecule. In prior work (Marchand-Geneste et al., 2002), these features have been used as inputs to a linear regression procedure to predict activity. Linear regression on these features will be effective in predicting activity if the following assumption holds: *the activity of a molecule is a linear function of the pharmacophores it has in at least one of its conformations*. This assumption is somewhat unsatisfactory, as we treat molecules where all pharmacophores match the same conformation(s) and molecules where each pharmacophore matches a different conformation in exactly the same way. Chemically, activity is likely to be a function of specific conformation(s) of the molecule, and this information has been lost. To capture this knowledge, we use a *multiple-instance* representation (Dietterich et al., 1997). In MI learning, examples are represented using *multisets* of feature vectors instead of single feature vectors. In MI terminology, each example is a *bag* of *instances*. Each bag is associated with a class label in a classification setting or a real-valued response in a regression setting. Given this representation, MI algorithms can learn models that predict the class label or response of novel bags.

To generate an MI representation for the drug activity prediction problem, we apply the proposed clauses (pharmacophores) *to each conformation* of each molecule separately. In this case, a 0/1 value represents whether a *specific conformation* has the given pharmacophore (clause). This creates an MI representation, where each molecule is represented by a bag of feature vectors, one per conformation, and the bag is labeled with the activity of the molecule. Given this representation, we use a multiple-instance regression algorithm to learn linear models. The task under consideration is defined as follows. We are given a set of  $n$  bags. The  $i^{\text{th}}$  bag consists of  $m_i$  instances and a real-valued response  $y_i$ . Instance  $j$  of bag  $i$  is described by a real-valued attribute vector  $\vec{X}_{ij}$  of dimension  $d$ . In the drug design example, each bag is a molecule, and each instance a conformation of the molecule represented by a feature vector. An iterative algorithm was presented in prior work (Ray & Page, 2001) to learn a linear model  $\hat{\mathbf{b}}$  under the assumption that there is some *primary* instance in each bag which is responsible for the real-valued label:

$$\hat{\mathbf{b}} = \arg \min_{\mathbf{b}} \sum_{i=1}^n (y_i - \vec{X}_{ip} \cdot \mathbf{b})^2, \quad (6.2)$$

where  $\vec{X}_{ip}$  is the feature vector describing the primary instance of bag  $i$ , and  $y_i$  is the response of bag  $i$ . The algorithm presented in prior work iterates between estimating the primary instance in each bag and solving the resulting linear regression problem until convergence. Recent work (Srinivasan et al., 2006) has used this approach to model drug activity, but has had limited empirical success.

Our approach extends this formulation to be more specific to activity prediction in the following way. Instead of assuming that a single, primary conformer is responsible for the activity of the molecule, we assume that the molecule’s activity is a *nonlinear weighted average* of the activities of its conformers. Biologically, each conformer can contribute to activity, but the contribution of a conformer dies off exponentially with goodness of fit between conformer and target. Thus, typically, the activity of a molecule will be dominated by its most active conformers. To model this scenario, we use a *softmax* function, denoted by  $S$  below:

$$S_{\alpha}(x_1, \dots, x_n) = \frac{\sum_{1 \leq i \leq n} x_i e^{\alpha x_i}}{\sum_{1 \leq i \leq n} e^{\alpha x_i}}. \quad (6.3)$$

The input to this function is the predicted activities of the conformation of any molecule. The output is a weighted average of the predicted activities, with the average being dominated by the most active conformation(s). As the parameter  $\alpha$  is increased, the output approximates the highest activity more closely. The softmax function has been used in prior work on MI classification as well (Maron, 1998). Thus, it is a suitable choice both from the biological and the MI perspectives. Further, note that the function is differentiable with respect to its inputs. This lets us use a gradient-based optimization procedure to solve for the best linear model  $\hat{\mathbf{b}}$  as follows:

$$\hat{\mathbf{b}} = \arg \min_{\mathbf{b}} \sum_{i=1}^n (y_i - S_{\alpha}(\vec{X}_{i1} \cdot \mathbf{b}, \dots, \vec{X}_{im_i} \cdot \mathbf{b}))^2 + \lambda \|\mathbf{b}\|^2. \quad (6.4)$$

Here,  $y_i$  represents the activity of the  $i^{th}$  molecule, and the predicted activity of conformation  $j$  of molecule  $i$  is defined by the linear function  $\vec{X}_{ij} \cdot \mathbf{b}$ . Thus, the first part of this objective specifies that we are searching for the linear model such that the total error between the weighted averages of the predicted conformation activities and the known molecular activities is minimized. The second part of the objective is a regularization factor proportional to  $\|\mathbf{b}\|^2$ . Incorporating such a factor is known to reduce overfitting to the training data, and thus improve generalization ability (Vapnik, 1999). We expect our approach to be more accurate than standard regression if (i) the activity of any *conformation* is a linear function of the pharmacophores it has, and (ii) the activity of any molecule is (approximately) an exponentially weighted average of the activities of its individual conformers.

The objective function in Equation 6.4 is nonlinear and nonconvex; hence, standard gradient-based optimization algorithms are susceptible to local minima. To reduce the possibility of being misled by local minima, we employ the technique of random restarts: when learning a model, the optimization algorithm is restarted several times from different, randomly chosen starting points and allowed to run to completion. The final solution returned is the one resulting in the lowest objective function value.

We call our approach, which combines the SAYU procedure with MI regression models, Multiple Instance Regression-SAYU, abbreviated as MIR-SAYU.

### 6.3 Empirical Evaluation

In this section, we evaluate our approach on three real-world activity prediction tasks: thermolysin inhibitors, dopamine agonists and thrombin inhibitors. We first describe the domains and characteristics of the datasets we use and then present and discuss our experimental results.

**Tasks.** The thermolysin inhibitors dataset we use is described in previous work (Marchand-Geneste et al., 2002). Thermolysin belongs to the family of metalloproteases and plays roles in physiological processes such as digestion and blood pressure regulation. The molecules in our dataset are known inhibitors of thermolysin. Activity for these molecules is measured in  $pK_i = -\log K_i$ , where  $K_i$  is a dissociation constant measuring the ratio of the concentrations of bound product to unbound constituents. A higher value indicates a stronger affinity for binding. The dataset we use has the 10 lowest energy conformations (as computed by the SYBYL software package ([www.tripos.com](http://www.tripos.com))) for each of 31 thermolysin inhibitors along with their activity levels. The relational background knowledge we have for this data was obtained from David Enot and Ross King and is similar (but not identical) to the background knowledge used in previous work (Marchand-Geneste et al., 2002). This background knowledge defines 26 chemical groups that can be used to define a pharmacophore.

The second dataset we use consists of dopamine agonists (Martin et al., 1993). Dopamine works as a neurotransmitter in the brain, where it plays a major role in movement control. Dopamine agonists are molecules that function like dopamine and produce dopamine-like effects and can potentially be used to treat diseases such as Parkinson’s disease. The dataset we use has 23 dopamine agonists along with their activity levels. For this dataset, the number of conformations for each molecule ranges from 5 to 50. The background knowledge we have for this dataset is more limited than in the previous dataset – we know about four groups: hydrogen donors, hydrogen acceptors, hydrophobes and basic nitrogen groups.

The final dataset we use consists of thrombin inhibitors (Cheng et al., 2002). Thrombin works as a blood coagulant and thus its inhibitors can be used as anti-coagulants. The dataset consists of

41 thrombin inhibitors and their activity levels. Each molecule has between 3 and 334 conformations. The background knowledge for this task includes information about six different types of chemical groups.

| Dataset                | Constant | LR-ALEPH | MIR-ALEPH | LR-SAYU | MIR-SAYU    |
|------------------------|----------|----------|-----------|---------|-------------|
| Dopamine Agonists      | 1.38     | 1.53     | 1.57      | 1.25    | <b>0.87</b> |
| Thermolysin Inhibitors | 1.93     | 1.47     | 1.31      | 1.37    | <b>1.27</b> |
| Thrombin Inhibitors    | 1.56     | 3.27     | 1.95      | 1.36    | <b>1.28</b> |

Table 6.2 Root mean squared errors for different methods on drug activity datasets. Values in bold indicate best results on each dataset. LR refers to linear regression, MIR to multiple-instance regression, and SAYU to the “Score As You Use” rule selection procedure.

**Experiments.** In our experiments, we test three hypotheses. First, we hypothesize that the SAYU procedure results in features that are better suited to regression than a feature construction criterion based on coverage, as standard Aleph uses. Second, we hypothesize that the MI regression procedure results in more accurate activity predictions than standard linear regression. Third, we hypothesize that the combined MIR-SAYU procedure will yield more accurate predictions than either extension by itself.

To test our hypotheses, we use four baselines along with our algorithm. These are as follows:

1. **Constant:** This algorithm simply predicts the average activity of all molecules in the training set as the activity for every novel molecule.
2. **LR-ALEPH:** This algorithm is the relational approach described in Section 6.1 and is similar to the framework of Marchand-Geneste et al. (2002). It uses Aleph to construct a set of clauses based on coverage. A single feature vector is generated for each molecule from these clauses. A model is learned using linear regression on these features. For novel molecules, feature vectors are generated using the same clauses and the learned linear model is used to predict activity.
3. **MIR-ALEPH:** This algorithm learns a MI regression model and uses it to predict activity, but uses the standard Aleph to construct features based on coverage.

4. **LR-SAYU**: This algorithm learns a linear regression model and uses it to predict activity, but uses the SAYU procedure to select features for the model.
5. **MIR-SAYU**: This is our proposed approach, as described in Section 6.2.

In prior work (Marchand-Geneste et al., 2002), relational approaches have been compared to other activity prediction methods, such as CoMFA (outlined in Section 6.1), and found to be competitive. Therefore, we restrict our current evaluation to the algorithms mentioned above.

In our experiments, Aleph searches over 4-point pharmacophores, as in prior work (Marchand-Geneste et al., 2002). For SAYU, we set the number of internal cross validation folds,  $n$ , to be 5 and the  $r^2$  improvement threshold,  $p$ , to be 0.2. As a stopping criterion for SAYU, we use a time threshold, allowing each fold one hour of runtime. For MI regression, the softmax parameter  $\alpha$  is set to 3 and the regularization factor  $\lambda$  to 1. These parameter values seemed reasonable after some initial exploration; they have not been tuned to these tasks. To optimize our objective functions, we use the L-BFGS algorithm (Fletcher, 1980). To evaluate the algorithms, we use leave-one-molecule-out cross validation. For each dataset, we hold out one molecule in turn as the test molecule and learn a model using the remaining molecules. We then predict the activity of the held-out molecule using the learned model. We report the root-mean-squared (RMS) errors averaged across the held-out molecules in Table 6.2.

From the table, we observe that for all three tasks, the methods using SAYU outperform the methods that do not use SAYU by a wide margin. In fact, we observe that for both the dopamine and thrombin datasets, LR-ALEPH and MIR-ALEPH both exhibit worse RMSE than the Constant model. This indicates that the coverage measure used by Aleph to induce features in these domains does not result in features that are able to generalize well to predicting the real-valued activity that we are ultimately interested in. From these results, we conclude that interleaving feature construction and model building using the SAYU procedure results in features that are better able to generalize to predicting activity than features generated by coverage-based measures, such as used by standard Aleph.

Comparing the two approaches using MI regression to the the ones using linear regression, we observe that in general, the MI approaches outperform their counterparts. The only exception is in the case of dopamine, where MIR-ALEPH is slightly worse than LR-ALEPH. However, we believe this is likely because the features generated by Aleph are not useful in predicting activity for this case, making any comparison between the linear regression and MI regression difficult. Apart from this case, we observe that MIR-ALEPH is more accurate than LR-ALEPH, and MIR-SAYU is more accurate than LR-SAYU. From these results, we conclude that incorporating knowledge about individual conformations using MI regression generally results in more accurate prediction models than using linear regression on a single feature vector for each molecule.

Finally, we observe that the combined approach we have presented in this work, MIR-SAYU, is the most accurate on all of our 3D-QSAR tasks. It is more accurate than either MIR-ALEPH, which uses MI regression models but does not use SAYU, or LR-SAYU, which uses SAYU, but not MI regression models. From these results, we conclude that combining the two extensions we have presented results in more accurate models than either extension by itself.

SAYU has been shown to consistently produce simpler models than ALEPH (Davis et al., 2005a). An interesting question to ask is whether MI regression yields more complex models than linear regression, that is, whether MIR-SAYU learns more complex models than LR-SAYU (the question makes sense only in the context of SAYU, because LR-ALEPH and MIR-ALEPH use the same set of features by design). While we did not enforce any constraint on the total number of features added to each model, we observed that these approaches used approximately the same numbers of features in our experiments. This indicates MIR-SAYU obtains its improvement over LR-SAYU by selecting more informative features (pharmacophores), rather than simply by using more features.

Another interesting question to ask is if the pharmacophores used by our MIR-SAYU models to predict activity have any biological interpretation. In fact, we observed that for dopamine, the rules used by MIR-SAYU on most of the folds agree with the general pharmacophore model in the literature (McGaughey & Mewshaw, 1999). They each have the key basic nitrogen, hydrogen

acceptor and hydrophobic group of the model. Since we specified that all rules must encode four-point pharmacophores, the rules all contained either an additional hydrophobic group or hydrogen acceptor; most contained the added hydrophobe. The one exception is a learned pharmacophore that had an extra hydrophobe in the position where the basic nitrogen should be; this feature had a substantial negative coefficient in the regression model. For thermolysin, the known pharmacophore model has seven interaction points, although all seven points are not required for binding. As a result, on every fold of cross-validation multiple rules were learned, capturing different four-point subsets of the seven-point pharmacophore. A combination of such four-point pharmacophores actually makes it possible to achieve better prediction of activity than would be done with a single seven-point pharmacophore, because different coefficients can be attached to each of the four-point pharmacophores. Finally, thrombin is a particularly interesting challenge because no pharmacophore model has been widely agreed upon or validated. Our models are less consistent across folds than for the other tasks, but again our approach shows improved predictive performance. Thus, we conclude that our approach is able to learn models that predict drug activity in terms of biologically meaningful pharmacophores. We expect that this property will prove helpful in analyzing the produced activity models.

## 6.4 Chapter Summary

We have presented MIR-SAYU, a novel machine learning approach for 3D-QSAR. Our approach extends prior work in two ways. First, we use SAYU to construct and select rules that define features, resulting in a tight coupling between feature construction and model building. This permits us to, at any time, learn the rule that most improves our prediction of real-valued activity. Second, we use MI regression for model building. This allows us to separate out the features that are true of each 3D conformer of a molecule. In our experiments on three real-world 3D-QSAR tasks, we observed that each extension by itself improved the accuracy of our predictions. Further, our proposed approach, which uses both extensions, resulted in the most accurate predictions. We also observed that our approach is able to discover biologically relevant pharmacophores when

predicting activity. In future work, we plan to explore more complex models of activity prediction, as well as feature construction procedures that search over more complex rule spaces.

## Chapter 7

### Learning New Tables and Higher-Arity Predicates

The initial approach to view learning, developed in Chapters 4, 5 and 6, suffers from two important drawbacks. First, it only creates new fields, not new tables. Second, the new fields are just learned approximations to the target concept. This chapter addresses both of these shortcomings. First, it provides a mechanism for learning a new view that includes full *new relational tables*, by constructing predicates that have a higher-arity than the target concept. Second, it learns predicates that apply to different *types* than the target concept, allowing us to invent predicates unrelated to the target concept. Third, it permits *a newly-invented relation, or predicate, to be used in the invention of other new relations*. Such re-use goes beyond simply introducing “short-cuts” in the search space for new relations; because the new approach also permits a relation to be from aggregates over existing relations, re-use actually extends the space of possible relations that can be learned by the approach. Because this new work extends SAYU by providing a mechanism for View Invention by Scoring TABLES, the resulting system is known as SAYU-VISTA.

In many domains, discovering intermediate hidden concepts can lead to improved performance. For instance, consider the well-known task of predicting whether two citations refer to the same underlying paper. The *CoAuthor* relation is potentially useful for disambiguating citations; for example, if S. Russell and S.J. Russell both have similar lists of coauthors, then perhaps they are interchangeable in citations. But the *CoAuthor* relation may not have been provided to the learning system. Furthermore, *CoAuthor* can be used as a building block to construct further explicit features for the system, such as a new predicate *SamePerson*. Ideally, the learning algorithm should be able to discover and incorporate relevant, intermediate concepts into the representation. SAYU-VISTA provides this capability.

This work originally appeared in Davis et al. (2006) and Davis et al. (2007c).

## 7.1 Learning New Predicates

The original motivation for view learning centers on learning a statistical expert system to provide decision support to radiologists (Davis et al., 2005b). There we use SRL because the learned statistical model sits on top of the NMD schema, a standard established by the American College of Radiology (2004). The goal of the data set is to predict which abnormalities on a mammogram are malignant. We will use mammography as a running example to help illustrate the key components of the algorithm.

SAYU-VISTA, nFOIL and SAYU all learn definite clauses and evaluate clauses by how much they improve the statistical classifier. The key difference in the algorithms rests in the form that the head of the learned clauses takes. In nFOIL and SAYU, the head of a clause has the same arity and type as the example, allowing us to precisely define whether a clause succeeds for a given example and hence whether the corresponding variable is true. In the Mammography domain, a positive example has the form `malignant(ab1)`, where `ab1` is a primary key for some abnormality. Every learned rule has the head `malignant(A)` such as in the following rule:

```
malignant(Ab1) if:
    ArchDistortion(Ab1,present),
    same_study(Ab1,Ab2),
    Calc_FineLinear(Ab2,present).
```

The Bayesian network variable corresponding to this rule will take value *true* for the example `malignant(ab1)` if the clause body succeeds when the logical variable `A` is bound to `ab1`.

SAYU-VISTA removes the restriction that all the learned clauses have the same head. First, SAYU-VISTA learns predicates that have a higher arity than the target predicate. For example, in the Mammography domain, predicates such as `p11(Abnormality1, Abnormality2)`, which relate pairs of abnormalities, are learned. Subsection 7.1.1 discusses scoring predicates that have higher-arithies than the target relation. Second, SAYU-VISTA learns predicates that have types

other than the example key in the predicate head. For example, a predicate  $p_{12}(\text{Visit})$ , which refers to attributes recorded once per patient visit, could be learned. In order to score predicates of this form, we introduce the concept of *Linkages*, which we explain in subsection 7.1.2. After discussing how to evaluate these types of predicates, we will present the full SAYU-VISTA algorithm.

### 7.1.1 Scoring Higher-Arity Predicates

SAYU-VISTA can learn a clause such as:

```
p11(Ab1,Ab2) if:
    density(Ab1,D1),
    prior-abnormality-same-loc(Ab1,Ab2),
    density(Ab2,D2),
    D1 > D2.
```

This rule says that  $p_{11}$ , some unnamed property, is true of a pair of abnormalities  $Ab_1$  and  $Ab_2$  if they are at the same location,  $Ab_1$  was observed first, and  $Ab_2$  has higher density than  $Ab_1$ . Thus  $p_{11}$  may be thought of as “density increase.” Unfortunately, it is not entirely clear how to match an example, such as  $\text{malignant}(ab_1)$ , to the head of this clause for  $p_{11}$ . SAYU-VISTA maps, or links, one argument to the example key and aggregates away any remaining arguments using existence or count aggregation. The next section describes the approach used for linkage; the remainder of this section discusses aggregation.

To illustrate the `exists` operator, consider predicate  $p_{11}$ , given above. In this clause variable  $Ab_1$  represents the more recent abnormality. Suppose we wish to create a feature for this clause, using existence aggregation. The feature is true for a given binding of  $Ab_1$  if there exists a binding for  $Ab_2$  that satisfies the body of the clause. Specifically, for an example  $\text{malignant}(ab_1)$ , this “density-increase” feature is true if there exists another abnormality  $ab_2$  such that “density-increase” is true of the tuple  $\langle ab_1, ab_2 \rangle$ .

Using the same clause and same example abnormality  $ab_1$ , we now turn to the count operator. In this case, we are interested in the number of solutions for  $B$  given that  $A$  is set to  $ab_1$ . This

means that the new feature we will propose is not binary. Currently, VISTA discretizes aggregated features using a binning strategy that creates three equal-cardinality bins, where three was chosen arbitrarily before the running of any experiments. Figures 7.1 and 7.2 illustrate how to score p11 with count aggregation.

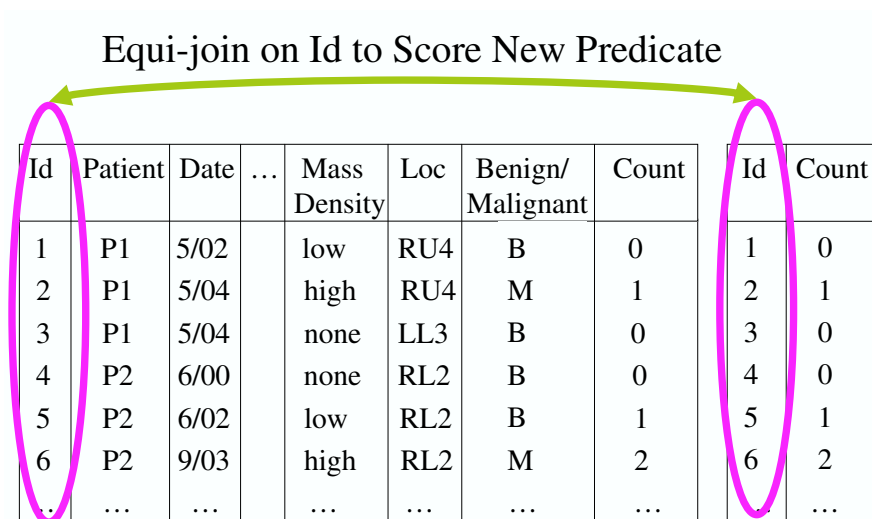


Figure 7.1 To score p11 using count aggregation, we join on Id to introduce the feature into the statistical model.

Aggregation queries are, in general, more expensive to compute than standard queries, as we may need to compute all solutions, instead of simply proving satisfiability. Thus, using aggregated views when inventing new views can be very computationally expensive (in fact, we can aggregate over aggregates). To address this problem, whenever VISTA learns an aggregated view, VISTA does not store the learned intensional definition of the view. Instead, VISTA materializes the view, that is, computes the model and stores the logical model as a set of facts. This solution consumes more storage, but it makes using aggregated views as efficient as using any other views.

### 7.1.2 Linkages

So far we have simplified matters by assuming that the first argument to the learned predicate has the same type as the example key. In our examples so far, this type has been *abnormality id*. There is no need to enforce this limitation. For example, in predicting whether an abnormality is

| Features in Statistical Model |         |      |     |              |     |                  |       | Added to Background Knowledge |       |
|-------------------------------|---------|------|-----|--------------|-----|------------------|-------|-------------------------------|-------|
| Id                            | Patient | Date | ... | Mass Density | Loc | Benign/Malignant | Count | Id                            | Count |
| 1                             | P1      | 5/02 |     | low          | RU4 | B                | 0     | 1                             | 0     |
| 2                             | P1      | 5/04 |     | high         | RU4 | M                | 1     | 2                             | 1     |
| 3                             | P1      | 5/04 |     | none         | LL3 | B                | 0     | 3                             | 0     |
| 4                             | P2      | 6/00 |     | none         | RL2 | B                | 0     | 4                             | 0     |
| 5                             | P2      | 6/02 |     | low          | RL2 | B                | 1     | 5                             | 1     |
| 6                             | P2      | 9/03 |     | high         | RL2 | M                | 2     | 6                             | 2     |
| ...                           | ...     | ...  |     | ...          | ... | ...              | ...   | ...                           | ...   |

Figure 7.2 If we accept p11, it will remain in the statistical model. Its definition will be added to the background knowledge, allowing for reuse in the future.

malignant, it might be useful to use the following clause, where Patient is a key that accesses patient level information:

```
p12(Patient) :-
    history_of_breast_cancer(Patient),
    prior_abnormality(Patient, Ab),
    biopsied(Ab, Date).
```

Predicate p12 is true of a patient, who has a family history of breast cancer and previously had a biopsy.

Linkage declarations are background knowledge that establish the connection between objects in the examples and objects in the newly invented predicates. When these objects are of the same type, the linkage is trivial; otherwise, it must be defined. For mammography, we use linkage definitions to connect an *abnormality* to its *patient* or to its *visit* (mammogram). Figures 7.3 and 7.4 illustrate how to score p12 by linking from a patient back to an abnormality. The linkages

| Features<br>in Statistical Model |         |      |     |                 |     |                      | New<br>Predicate |         |             |
|----------------------------------|---------|------|-----|-----------------|-----|----------------------|------------------|---------|-------------|
| Id                               | Patient | Date | ... | Mass<br>Density | Loc | Benign/<br>Malignant | p12<br>True      | Patient | p12<br>True |
| 1                                | P1      | 5/02 | ... | low             | RU4 | B                    | No               | P1      | No          |
| 2                                | P1      | 5/04 | ... | high            | RU4 | M                    | No               | P2      | Yes         |
| 3                                | P1      | 5/04 | ... | none            | LL3 | B                    | No               | P3      | No          |
| 4                                | P2      | 6/00 | ... | none            | RL2 | B                    | Yes              | P4      | No          |
| 5                                | P2      | 6/02 | ... | low             | RL2 | B                    | Yes              | P5      | Yes         |
| 6                                | P2      | 9/03 | ... | high            | RL2 | M                    | Yes              | P6      | No          |
| ...                              | ...     | ...  | ... | ...             | ... | ...                  | ...              | ...     | ...         |

Figure 7.3 Here we link from a patient back to an abnormality. The value of *Had Biopsy* for key P1 in the *New Predicate* relation gets applied to each row associated with P1 in the statistical model.

for the other datasets we use are equally straightforward and are presented when we describe those datasets.

### 7.1.3 Predicate Learning Algorithm

At a high level, SAYU-VISTA learns new predicates by performing a search over the bodies of definite clauses and selecting those bodies that improve the performance of the statistical model on a classification task. We use tree-augmented naive Bayes (TAN) (Friedman et al., 1997) as our statistical model.

The predicate invention algorithm takes several inputs from a user.

1. A training set, to learn the statistical model.
2. A tuning set, to evaluate the statistical model.
3. A pre-defined set of distinguished types, which can appear in the head of a clause.
4. Background knowledge, which must include linkage definitions for each distinguished type.

| Features in Statistical Model |         |      |     |              |     |                  |          | Added to Background Knowledge |          |
|-------------------------------|---------|------|-----|--------------|-----|------------------|----------|-------------------------------|----------|
| Id                            | Patient | Date | ... | Mass Density | Loc | Benign/Malignant | p12 True | Patient                       | p12 True |
| 1                             | P1      | 5/02 |     | low          | RU4 | B                | No       | P1                            | No       |
| 2                             | P1      | 5/04 |     | high         | RU4 | M                | No       | P2                            | Yes      |
| 3                             | P1      | 5/04 |     | none         | LL3 | B                | No       | P3                            | No       |
| 4                             | P2      | 6/00 |     | none         | RL2 | B                | Yes      | P4                            | No       |
| 5                             | P2      | 6/02 |     | low          | RL2 | B                | Yes      | P5                            | Yes      |
| 6                             | P2      | 9/03 |     | high         | RL2 | M                | Yes      | P6                            | No       |
| ...                           | ...     | ...  |     | ...          | ... | ...              | ...      | ...                           | ...      |

Figure 7.4 If we accept p12, it will remain in the statistical model. Its definition will be added to the background knowledge, allowing for reuse in the future.

5. An improvement threshold,  $p$ , to decide which predicates to retain in the model. A new predicate must improve the model's performance by at least  $p\%$  in order to be kept. We used  $p = 2$  in all experiments.
6. An initial feature set, which is optional.

Algorithm 4 shows pseudo code for the SAYU-VISTA algorithm.

The clause search proceeds as follows. We randomly select an arity for the predicate. To limit the search space, we restrict the arity to be either the arity of the target relation, or the arity of the target relation plus one. Next, we randomly select the types for the variables that appear in the head of the clause. The clause search uses a top-down, breadth-first refinement search. We define the space of candidate literals to add using modes, as in Progol (Muggleton, 1995) or Aleph (Srinivasan, 2001). We score each proposed clause by adding it as a variable in the statistical model. To construct the feature, we first link the predicate back to the example key as described in Subsection 7.1.2. Then we perform the necessary aggregation, discussed in Subsection 7.1.1, to convert the clause into a feature. By default, the algorithm first tries existence

```

Input: Train Set Labels  $T$ , Tune Set Labels  $S$ , Distinguished Types  $D$ , Background
          Knowledge  $B$ , Improvement Threshold  $p$ , Initial Feature Set  $F_{init}$ 
Output: Feature Set  $F$ , Statistical Model  $M$ 
 $F = F_{init}$ ;
 $BestScore = 0$ ;
while time remains do
  Randomly select the arity of predicate to invent;
  Randomly select types from  $D$  for each variable in the head of the predicate;
   $SelectedFeature = false$ ;
  while not(SelectedFeature) do
     $Predicate = \text{Generate next clause according to breadth first search}$ ;
    /* Link the predicate back to the target relation */
     $LinkedClause = \text{Link}(Predicate, B)$ ;
    /* Convert the LinkedClause into a feature that the
       statistical model can use */
     $NewFeature = \text{aggregate}(LinkedClause, T, S)$ ;
     $F_{new} = F \cup NewFeature$ ;
     $M_{new} = \text{BuildTANNNetwork}(T, F_{new})$ ;
     $NewScore = \text{AreaUnderPRCurve}(M, S, F_{new})$ ;
    /* Retain this feature */
    if ( $NewScore > (1 + p) * BestScore$ ) then
       $F = F_{new}$ ;
       $BestScore = NewScore$ ;
       $M = M_{new}$ ;
      Add predicate into background knowledge;
       $SelectedFeature = true$ ;
    end
  end
end

```

**Algorithm 4:** SAYU-VISTA Algorithm

aggregation and then tries count aggregation. The clause search *terminates* in three cases: **(i)** it finds a clause that meets the improvement threshold; **(ii)** it fully explores the search space; **(iii)** it exceeds the clause limit. After satisfying one of these conditions, the algorithm re-initializes the search process. The algorithm adds every clause that meets the improvement threshold into the background knowledge. Therefore, future predicate definitions can re-use previously learned

predicates. As in prior work (Davis et al., 2005a), the algorithm terminates when it exceeds the global time limit.

## 7.2 Data and Methodology

**Cora.** The objective of this dataset is to predict whether two citations refer to the same paper. The dataset was originally constructed by McCallum et al. (2000). We use the same version of the data as Kok and Domingos (2005). Cora includes 1295 citations to 112 Computer Science papers, resulting in 25072 positive examples and 597310 negative examples. The background knowledge includes data on title, venue, author(s), and year for each citation. We define *paper*, *title*, *venue*, *author* and *year* as keys that can appear in heads of clauses. We link a paper to its title, venue, author(s) and year fields. We aggregate over papers and authors.

**UW-CSE.** This is the same dataset used in Chapter 5. We defined *students*, *professors*, *courses* and *publications* as keys that could appear in the head of a clause. We link a course to a graduate student by the TA relationship, and we link papers to a graduate student by the author relationship. We link a course to a professor by the teaches relationship and we link papers to a professor by the author relationship. We aggregate over students, professors, papers and courses.

**Mammography.** This is the same dataset as used in Chapter 4 and Chapter 5. We define *abnormality*, *visit* and *patient* as keys that can appear in the head of the clause. We aggregate over abnormalities.

## 7.3 Experiments and Results

We want to test three hypotheses in our experiments. First, we hypothesize that SAYU-VISTA will learn more accurate models than SAYU. Second, we hypothesize that SAYU-VISTA will perform competitively with other leading SRL systems. Third, SAYU-VISTA will learn relevant predicates.

To test our hypotheses we will use the following three algorithms:

1. **SAYU** is the state-of-the-art view learning implementation. However, SAYU only learns additional fields for existing tables; these fields are defined by learned rules that are approximations to the target concept.
2. **Markov Logic Networks**, or MLNs (Richardson & Domingos, 2006), are a leading SRL system that has received considerable attention. Furthermore, they have already been applied with success (as measured by cross-validated precision-recall curves) to two of our application tasks. MLNs are publicly available as the Alchemy system.
3. **SAYU-VISTA** is this chapter’s proposed approach.

We evaluate all three SRL systems by precision-recall curves estimated by cross-validation with significance of differences tested by a paired two-tailed t-test on areas under the precision-recall curves (AUC-PR) across the different folds. We are careful to repeat any tuning of parameters on each fold of cross-validation, without looking at the test set for that fold, by dividing the data into a training set and tuning set. In this we follow the methodology of the developers of both MLNs and SAYU. For SAYU-VISTA, as for SAYU, we use the training set to learn the network parameters, while we use the tuning set to score potential clauses. For all datasets we use AUC-PR as our score metric. However, we only look at AUC-PR for recalls  $\geq 0.5$ . We do this for two reasons. First, precision can have high variance at low levels of recall. Second, in domains such as Mammography, we are only interested in high levels of recall. A practicing radiologist would need to achieve at least this level of recall. A clause must improve the AUC-PR (for recall  $\geq 0.5$ ) by at least 2% in order to be retained in the network. This is an arbitrary parameter setting; in fact we did not try any other thresholds. We had a time-based stop criteria for both SAYU and SAYU-VISTA. For UW-CSE each fold was given two hours to run, whereas for Mammography and Cora each fold received three hours run-time. We gave UW-CSE less time because it was a smaller data set. In practice, the time is not a limiting factor because few changes occur after the first 30 minutes for any of the tasks. MLN runs were not time-bounded. To offset potential differences in computer speeds, all experiments were run on identically configured machines.

|             | MLN    | SAYU   | SAYU-VISTA | p-value<br>vs. SAYU | p-value<br>vs. MLN    |
|-------------|--------|--------|------------|---------------------|-----------------------|
| Cora        | 0.468  | 0.371  | 0.461      | 0.0109              | 0.309                 |
| UW-CSE      | 0.0622 | 0.0975 | 0.167      | 0.0581              | 0.165                 |
| Mammography | 0.0172 | 0.103  | 0.104      | 0.969               | $5.89 \times 10^{-6}$ |

Table 7.1 Average AUC-PR for Recall  $\geq 0.5$  for Each Task Using TAN as the Statistical Model

We employ the default structure learning algorithm for MLNs and perform limited manual tuning of the parameters of the system to maximize AUC-PR, while maintaining acceptable execution times. We report the best AUC-PR values we obtained, over all attempted parameter settings. Note that we did not do any parameter tuning for SAYU-VISTA. The average, per-fold run-times for MLNs were all significantly longer than for either SAYU or SAYU-VISTA. The average, per fold run times for learning structure were five hours for Cora, seven hours for UW-CSE and three hours for Mammography.

### 7.3.1 Discussion of Results

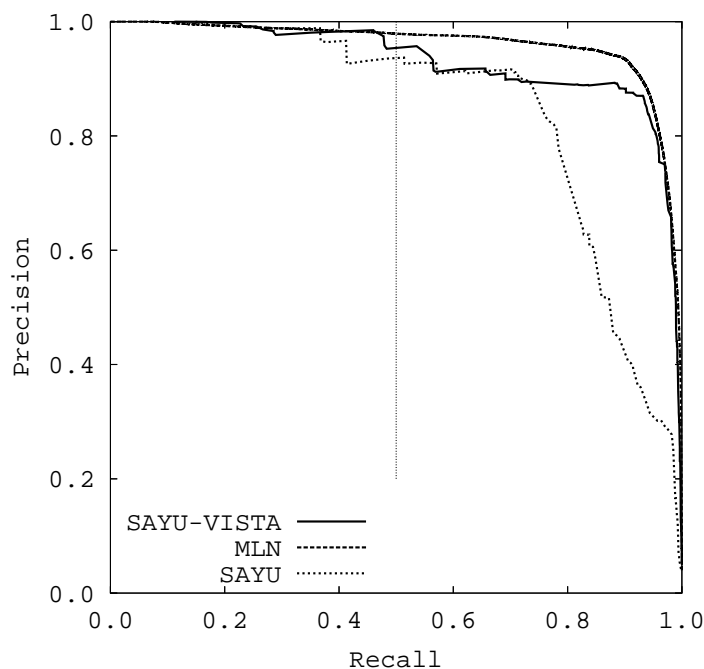


Figure 7.5 Cora Precision-Recall Curves

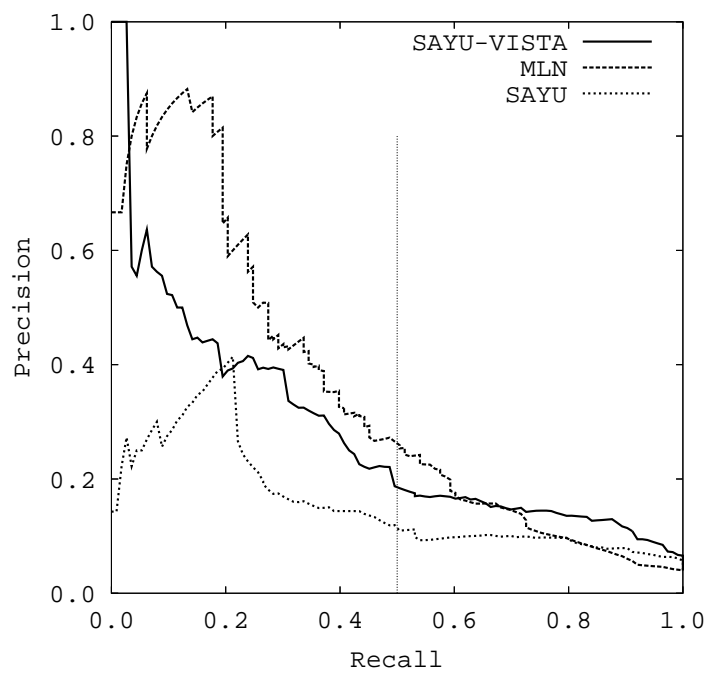


Figure 7.6 UW-CSE Precision-Recall Curves

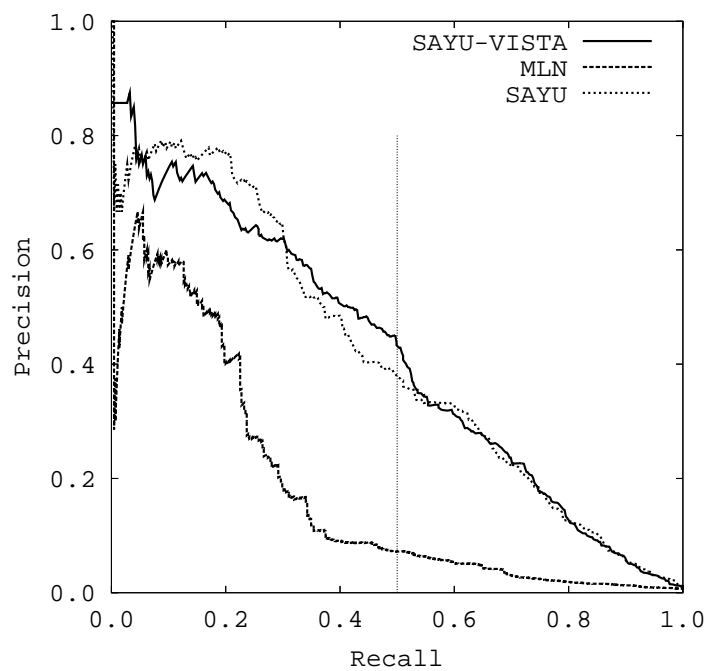


Figure 7.7 Mammography Precision-Recall Curves

**Cora.** Following Kok and Domingos (2005) we perform two-fold cross validation on this dataset for five different random train-test splits. We divided the training set in half, to form a new training set and a tuning set. Each fold receives three hours of CPU time to run. SAYU and SAYU-VISTA could evaluate up to 300 clauses, before selecting a new seed or clause head.

Table 7.1 reports the average AUC-PR (recall  $\geq 0.5$ ) for Cora and the p-value for a two-tailed paired t-test between SAYU-VISTA and the other two algorithms. SAYU-VISTA performs significantly better than SAYU on this domain. Figure 7.5 shows precision-recall curves for all algorithms on this dataset. We pool results across all folds to generate the curves. SAYU-VISTA dominates SAYU throughout precision-recall space. However, MLNs have a slightly higher average AUC-PR than SAYU-VISTA does, although the difference is not significant. MLNs receives an advantage over SAYU and SAYU-VISTA in this task, as MLNs start with an expert knowledge base.

Let us discuss two predicates found by SAYU-VISTA:

```
p1(Venue1,Venue2):-
    commonWordsInVenue80(Venue1,Venue2).
```

```
p4(Paper1,Paper2,Paper3):-
    paperTitle(Paper2,Title),
    paperTitle(Paper1,Title),
    paperTitle(Paper3,Title).
```

Predicate p1 captures a crucial piece of partial knowledge in the citation matching domain by checking how similar the venue field is between two citations. This rule demonstrates how SAYU-VISTA can learn predicates whose “distinguished variables” have a different type than “distinguished variables” in the target relation. Predicate p4 establishes a transitive relationship between papers with exactly the same title. It also is an example of a learned predicate that has a higher-arity than the target concept.

**UW-CSE.** Following Richardson and Domingos (2006), we perform five-fold cross validation on the UW-CSE dataset. We use two folds for the training set and two folds for a tuning set. Each

approach could evaluate up to 10000 clauses, before either selecting a new seed (SAYU) or a new predicate head (SAYU-VISTA).

Table 7.1 reports the average AUC-PR for UW-CSE and the p-value for a two-tailed paired t-test comparing SAYU-VISTA to the other approaches. SAYU-VISTA comes close to performing significantly ( $0.05 < p < 0.06$ ) better than SAYU on this domain. Although performance varies widely between the 5 folds, SAYU-VISTA had a higher AUC-PR than SAYU on each fold. SAYU-VISTA also comes close to outperforming MLNs on this data set, winning on four out of five folds.

Figure 7.6 shows precision-recall curves for SAYU, SAYU-VISTA and MLNs on this dataset. We pool results across all five folds to generate the curves. Even though we measured AUC-PR for recall  $\geq 0.5$ , SAYU-VISTA dominates SAYU for most levels of recall. However, MLNs dominate SAYU-VISTA for low levels of recall, whereas SAYU-VISTA tends to dominate for the high levels of recall. We also compare the performance of SAYU-VISTA (average AUC-PR of 0.468) and MLNs (average AUC-PR of 0.355) for AUC-PR for all levels of recall. Again, there is no significant difference. SAYU-VISTA has a higher variation in per fold AUC-PR score than MLNs do. One reason for SAYU-VISTA's increased performance for high recall is that we are expressly optimizing for this metric. MLNs also receive one advantage over SAYU and SAYU-VISTA in this domain, in that they start with an expert defined knowledge base.

Let us examine two predicates learned by SAYU-VISTA:

```
p16(Student,Professor):-
    ta(Course,Student,Date),
    taughtby(Course,Professor,Date).
```

```
p22(Student,Professor,Paper):-
    publication(Paper,Student),
    publication(Paper,Professor).
```

Both p16 and p22 capture intuitive knowledge about the advisee-advisor relationship. First, graduate students often TA a class taught by their advisor. Second, a graduate student will usually co-author a paper with their advisor. Predicate p22 makes use of SAYU-VISTA's ability to learn a

predicate with a higher-arity than the target predicate. SAYU-VISTA actually learned a variant of p22 on four folds and a variant of p22 on all five folds.

**Mammography.** Following our prior work, we perform ten-fold cross validation on this dataset. We use four folds for a training set and five folds as a tuning set. Each algorithm could evaluate at most 300 clauses for a given seed (SAYU) or clause head (SAYU-VISTA).

For the previous two datasets, we initially started with a Bayesian network that only contains a feature for the target predicate. However, in the Mammography domain we have access to a set of expert defined features (from the NMD). Furthermore, we could define a set of aggregate features as done in Chapters 4 and 5. Opposed to starting with an empty network structure, we begin with a network that contains both NMD features and the aggregate features.

Table 7.1 reports the average AUC-PR over all folds. We use a two-tailed paired t-test to compute significant results, and the p-value for the test can also be found in Table 7.1. We find no significant difference between SAYU-VISTA and SAYU on this task, yet SAYU-VISTA does not perform any worse than SAYU. However, both SAYU and SAYU-VISTA significantly outperform MLNs on this domain.

Figure 7.7 shows precision-recall curves for both algorithms on this dataset. We pool results across all folds to generate the curves. On this dataset, SAYU-VISTA and SAYU have comparable performance for all levels of recalls. SAYU-VISTA and SAYU both dominate MLNs for all levels of recall. Note that, as in the UW-CSE domain, MLNs tend to have better performance for low levels of recall. We feel there are several potential reasons that SAYU-VISTA did not perform significantly better than SAYU on this domain. First, for this application, MLNs and SAYU receive a large number of features—the precomputed aggregates—that SAYU-VISTA could potentially learn, but MLNs and SAYU cannot easily capture. Second, this domain contains many more constants than other domains, thus by leveraging Aleph, SAYU has a smaller and more directed search. Finally, the mammography domain contains only three relations, while the other domains each have approximately twenty relations. Thus, on those domains there is more room to exploit the ability to learn predicates that (1) have different types in the head and (2) represent new tables.

In order to allow MLNs to run on this domain, we had to do drastic sub-sampling of the negative examples. MLNs struggled with having to ground out a network with the large number of examples that this data set contains. Another possible explanation for SAYU and SAYU-VISTA's better performance is that we seed the algorithm with an initial feature set. However, we ran the experiments where we started SAYU and SAYU-VISTA with an empty network structure and it still significantly outperformed MLNs.

Finally, here is a sample predicate found by SAYU-VISTA:

```
p23(Ab1,Ab2) :-
    aggregate(Ab2,count,same_loc(Ab1,Ab2))
```

This predicate counts the number of prior abnormalities that were in the same location as the current abnormality. This predicate displays SAYU-VISTA's ability to learn clauses that compute aggregate information about prior abnormalities.

**Hypotheses revisited.** In our experiments we found support for all three of our hypotheses. SAYU-VISTA generates more accurate models than both SAYU and MLNs. Additionally, it is able to build these models much faster than MLNs. SAYU-VISTA constructs interesting intermediate concepts. For example, it discovers the *CoAuthor* and *TA'ed For* relationships in the UW-CSE domain.

### 7.3.2 Further Investigation of SAYU-VISTA

SAYU-VISTA adds several components to SAYU. First, it adds count aggregation: the ability to handle many-to-many and one-to-many relationships by adding a feature to the statistical model that counts the number of satisfying assignments for a predicate. Second, linkages allow us to learn entirely new tables. Third, we allow for previously invented predicates to appear in the definitions of new predicates. Without linkages, SAYU-VISTA reduces to SAYU. To discover the extent to which the other two features contribute to SAYU-VISTA's performance, we consider removing the first and third components from SAYU-VISTA and observe the resulting performance.

The first component, counting the number of satisfying assignments, does not help in either Cora or the Mammography domain. It is never used in Cora and it is only used twice in Mammography. Consequently, we do not need to consider removing it on these domains. However, it appears 11 times, or about twice per fold in the UW-CSE domain. Removing it reduces the AUC-PR for this domain from 0.152 to 0.142. This degrades performance on four out of five folds, yet the change is not significant, having a p-value of 0.16. However, it seems that even though counting does not help on two out of three domains, it can potentially be useful for an SRL system.

The other component of SAYU-VISTA we remove is the third, that of adding the learned predicates into background knowledge. Disabling this feature slightly improves performance in Mammography, increasing AUC-PR from 0.104 to 0.105. However in Cora it decreases AUC-PR from 0.461 to 0.448 and in UW-CSE the performance declines from 0.152 to 0.148. Across all these experiments none of the changes are significant.

On Cora, the benefit comes only from the introduction of linkages. On UW-CSE, the benefit comes from both linkages and the count aggregation. In a sense linkages are the key innovation of SAYU-VISTA. Linkages allow us to both learn new tables and to learn concepts that are not simply approximations to the target concept. Reusing learned predicates does not seem to provide a win. Asserting each learned predicate might unnecessarily widen the search space.

## 7.4 Related Work

We already have discussed how the present chapter advances the state-of-the-art in view learning. The chapter also is related to propositionalization within ILP (Lavrač et al., 1991), particularly to propositionalization approaches that incorporate aggregation (Kroegel & Wrobel, 2001; Knobbe et al., 2001; Popescul et al., 2003; Popescul & Ungar, 2004). These approaches construct clause bodies that define new features or propositions. The value of such a feature for a data point, or example, is obtained by binding one of the variables in the clause body to the example’s key, and then aggregating over the remaining features. In this fashion, the definition of a feature is equivalent to a definite clause whose head is “ $p(X)$ ”, where  $p$  is an arbitrary predicate name and  $X$  is the body variable that is bound in turn to each example’s key. Both existential and count aggregation

have been employed before (Kroegel & Wrobel, 2001). In fact, all the approaches cited above have used more complex aggregations than does SAYU-VISTA, and these could be incorporated easily into SAYU-VISTA.

The novel properties of SAYU-VISTA relative to propositionalization by aggregation are the following. First, *subsets* of the variables in the clause body may be mapped back to an example's key, via the domain-specific linkage relations. This enables learning of new tables or *non-unary* predicates that have different arities and types than the examples. Second, to score each potential new table or predicate, SAYU-VISTA constructs an entire statistical model, and only retains the new predicate if it yields an improved model. Third, learned predicates are available for use in the definitions of further new predicates.

Other general areas of related work are of course constructive induction (Rendell, 1985) and predicate invention (Muggleton & Buntine, 1988; Zelle et al., 1994), as well as learning latent or hidden variables in Bayesian networks (Connolly, 1993). Predicate invention is a specific type of constructive induction, where a new predicate is defined not based directly on examples of that predicate, but on the ability of that predicate to help in learning the definitions of other predicates for which examples are available. The classic difficulties with predicate invention are that, unless predicate invention is strongly constrained: (1) the search space of possible predicates is too large, (2) too many new predicates are retained, thus reducing efficiency of learning, and (3) the ability to invent arbitrary new predicates leads to overfitting of training data.

The present work can be seen as a type of predicate invention, because arbitrary clauses are constructed whose heads do not have to unify with the examples—they may have arities and types different from the examples. SAYU-VISTA is analogous to CHILLIN (Zelle et al., 1994) and Closed World Specialisation (Srinivasan et al., 1992). Both of those systems search for an intensional definition of a clause based on existing predicates, just like SAYU-VISTA. One key difference is that those systems don't directly search for new predicates. CHILLIN is demand driven, and Closed World Specialisation invents predicates to handle exceptions to the theory, whereas SAYU-VISTA directly searches for new predicates. The other important difference is how the systems evaluate new predicates. The other systems use traditional ILP metrics, such as compaction. The approach

in the present chapter constrains predicate invention by requiring invented predicates to be of immediate value to the statistical learner in order to be retained for further use. The empirical success of SAYU-VISTA—that it does not hurt performance and it sometimes helps—indicates that this efficacy test is a successful constraint on predicate invention.

Cigol (Muggleton & Buntine, 1988) was the first system to perform predicate invention in the predicate calculus. It is an interactive algorithm that invents new predicates through inverse-resolution. Cigol starts with a set of examples, and generalizes the examples to form a theory. Cigol relies on two main operators to form a theory. The first operator is called the  $V$  operator. Assume that two clauses,  $C_1$  and  $C_2$ , resolve to  $C$ . The  $V$  operator, working from  $C$  and  $C_2$ , performs the inverse of resolution to recover  $C_1$ . The  $W$  operator works by placing two  $V$  operations side by side and is able to invent new predicates. Assume that clauses  $C_1$  and  $A$  resolve to  $B_1$  and that clauses  $A$  and  $C_2$  resolve to  $B_2$ . The  $W$  operation takes  $B_1$  and  $B_2$  and reconstructs  $C_1$ ,  $C_2$  and  $A$ . Now because  $C_1$ ,  $C_2$  and  $A$  all resolve together, they could contain a literal  $L$  that is not present in either  $B_1$  or  $B_2$ . Thus, Cigol is able to invent arbitrary predicates  $L$  that are not given to the system by a user. Cigol asks a user if the predicate it has invented represents a valid generalization for the target concept that Cigol is trying to learn. This allows Cigol to disregard irrelevant predicates. SAYU-VISTA ignores predicates that do not improve the statistical model and does not need human interaction to decide which predicates to retain.

Another algorithm that performs predicate invention is Craven and Slattery's (2001) FOIL-PILFS system. FOIL-PILFS extends FOIL (Quinlan, 1990) to invent statistical predicates. Craven and Slattery are interested in classification of text, so the invented predicates correspond to a Naïve Bayes classifier. The features for the classifier are derived from the words in a documents. The training set consists of the set of documents currently covered by the clause under construction. In a sense, this can be seen as performing the inverse of what SAYU-VISTA does, as FOIL-PILFS turns a Bayes net into a feature to included in a clause. On the other hand, SAYU-VISTA invents a predicate that can be included as a feature in a Bayes net.

More recent work on predicate invention includes deriving clusterings and treating cluster membership as an invented feature. Popescul and Ungar (2004) define new predicates based on

clusterings that are constructed in an initial pre-processing step. This is done before learning predicates that define the features for the statistical model; the latter features are never re-used. Kemp et al. (2006) propose using an infinite relational model (IRM) to cluster entities in a domain. The cluster that an entity is assigned to should be predictive of the relationships it satisfies. A weakness to this approach is that each entity can belong to only one cluster. Xu et al. (2006) simultaneously came up with a similar idea. Kok and Domingos (2007) propose an algorithm that learns multiple relational clusters (MRC). The MRC algorithm goes beyond IRM in two important ways. First, the algorithm can cluster both relations and entities. Second, the relations and entity can belong to more than one cluster. The primary weakness to MRC is that it is a transductive approach.

The topic of learning Bayesian network structures with the introduction of new (latent) variables faces similar obstacles to predicate invention. Because the new variables are unconstrained by the data, their introduction into Bayesian network structure learning permits overfitting of the training data, in addition to increasing search complexity. SAYU-VISTA may be seen as introducing new variables into the structure learning task; nevertheless, by requiring these new variables to be defined using existing (pre-defined or recently learned) relations, these variables are partially constrained. The empirical success of SAYU-VISTA provides some evidence that this constraint on the new variables helps to avoid overfitting. SAYU-VISTA's use of TAN Bayes nets also helps to reduce the search space. Both predicate invention and Bayes net learning with the introduction of new variables are widely noted to be extremely difficult tasks. This chapter provides some evidence that attempting to address both tasks at the same time, within an SRL framework, can actually make both tasks somewhat easier.

## 7.5 Chapter Summary

This chapter presents the SAYU-VISTA algorithm, which extends view learning in three substantial ways. First, it creates predicates that have a higher-arity than the target concept, which

capture many-to-many relations and require a new table to represent. Second, it constructs predicates that operate on different types than the target concept, allowing it to learn relevant, intermediate concepts. Third, it permits newly-invented predicates to be used in the invention of other new relations.

SAYU-VISTA's view learning capability provides a mechanism for predicate invention, a type of constructive induction investigated within ILP. As with predicate invention, the space of new views one can define for a given relational database is vast, leading to problems of overfitting and search complexity. SAYU-VISTA constrains this space by

- Learning definitions of new relations (tables or fields) one at a time.
- Considering only new relations that can be defined by short clauses expressed in terms of the present view of the database (including background knowledge relations provided as intensional definitions).
- Re-constructing the SRL model when testing each potential new relation, and keeping a new relation only if the resulting SRL model significantly outperforms the previous one.

The last step requires matching a subset of the arguments in the relation with the arguments in the data points, or examples, and aggregating away the remaining arguments in the relation.

We empirically demonstrated that SAYU-VISTA leads to more accurate models than both the SAYU algorithm and Markov Logic Networks (MLNs), another leading SRL framework. Furthermore, SAYU-VISTA constructed useful and relevant predicates, such as consistently discovering the *CoAuthor* and *TA'ed for* relations in the UW-CSE domain.

a

## Chapter 8

### The Relationship between PR Space and ROC Space

This thesis has exclusively focused on using PR curves to present results for classification algorithms. Chapter 4, noted that we prefer PR curves to ROC curves when evaluating domains that have a highly skewed class distribution. Our use of PR curves led us to study the connection between these two spaces, and determine whether some of the interesting properties of ROC space also hold for PR space. Our investigations result in four important contributions. First, we show that for any dataset, and hence a fixed number of positive and negative examples, the ROC curve and PR curve for a given algorithm contain the “same points.” Based on this equivalence for ROC and PR curves, we prove that a curve dominates in ROC space if and only if it dominates in PR space. Second, we introduce the PR space analog to the convex hull in ROC space, which we call the achievable PR curve. We show that due to the equivalence of these two spaces we can efficiently compute the achievable PR curve. Third we offer theoretical justification for the fact that it is insufficient to linearly interpolate between points in PR space. Finally, we prove that an algorithm that optimizes the area under the curve in one space is not guaranteed to optimize the area under the curve in the other space.

This work originally appeared in Davis and Goadrich (2006).

#### 8.1 Why We Use PR Curves

In machine learning, current research has shifted away from simply presenting accuracy results when performing an empirical validation of new algorithms. This is especially true when evaluating algorithms that output probabilities of class values. Provost et al. (1998) have argued that

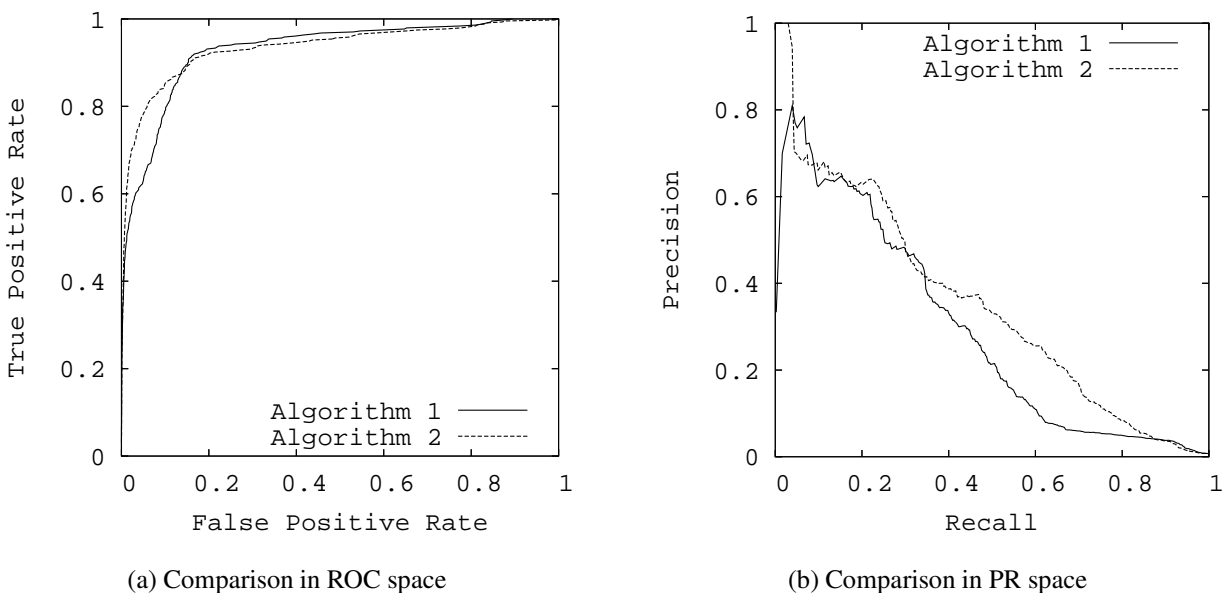


Figure 8.1 The Difference between Comparing Algorithms in ROC vs PR Space

simply using accuracy results can be misleading. They recommended using ROC curves when evaluating binary decision problems. However, ROC curves can present an overly optimistic view of an algorithm's performance if there is a large skew in the class distribution. Drummond and Holte (2000; 2004) have recommended using cost curves to address this issue. Cost curves are an excellent alternative to ROC curves, but discussing them is beyond the scope of this chapter.

Precision-Recall (PR) curves, often used in Information Retrieval (Manning & Schütze, 1999; Raghavan et al., 1989), have been cited as an alternative to ROC curves for tasks with a large skew in the class distribution (Bockhorst & Craven, 2005; Bunescu et al., 2005; Davis et al., 2005b; Goadrich et al., 2004; Kok & Domingos, 2005; Singla & Domingos, 2005). An important difference between ROC space and PR space is the visual representation of the curves. Looking at PR curves can expose differences between algorithms that are not apparent in ROC space. Figures 8.1(a) and 8.1(b) show sample ROC curves and PR curves respectively. These curves, taken from the same learned models on a highly-skewed cancer detection dataset, highlight the visual difference between these spaces (Davis et al., 2005b). The goal in ROC space is to be in the upper-left-hand corner, and when one looks at the ROC curves in Figure 8.1(a) they appear to be fairly

close to optimal. In PR space the goal is to be in the upper-right-hand corner, and the PR curves in Figure 8.1(b) show that there is still vast room for improvement.

The performances of the algorithms appear to be comparable in ROC space, however, in PR space we can see that Algorithm 2 has a clear advantage over Algorithm 1. This difference exists because in this domain the number of negative examples greatly exceeds the number of positive examples. Consequently, a large change in the number of false positives can lead to a small change in the false positive rate used in ROC analysis. Precision, on the other hand, by comparing false positives to true positives rather than true negatives, captures the effect of the large number of negative examples on the algorithm's performance.

## 8.2 Relationship between ROC Space and PR Space

ROC and PR curves are typically generated to evaluate the performance of a machine learning algorithm on a given dataset. Each dataset contains a fixed number of positive and negative examples. We show here that there exists a deep relationship between ROC and PR spaces.

**Theorem 8.2.1.** *For a given dataset of positive and negative examples, there exists a one-to-one correspondence between a curve in ROC space and a curve in PR space, such that the curves contain exactly the same confusion matrices, if Recall  $\neq 0$ .*

**Proof.** Note that a point in ROC space defines a unique confusion matrix when the dataset is fixed. Since in PR space we ignore  $TN$ , one might worry that each point may correspond to multiple confusion matrices. However, with a fixed number of positive and negative examples, given the other three entries in a matrix,  $TN$  is uniquely determined. If Recall = 0, we are unable to recover  $FP$ , and thus cannot find a unique confusion matrix.  $\square$

Consequently, we have a one-to-one mapping between confusion matrices and points in PR space. This implies that we also have a one-to-one mapping between points (each defined by a confusion matrix) in ROC space and PR space; hence, we can translate a curve in ROC space to PR space and vice-versa.

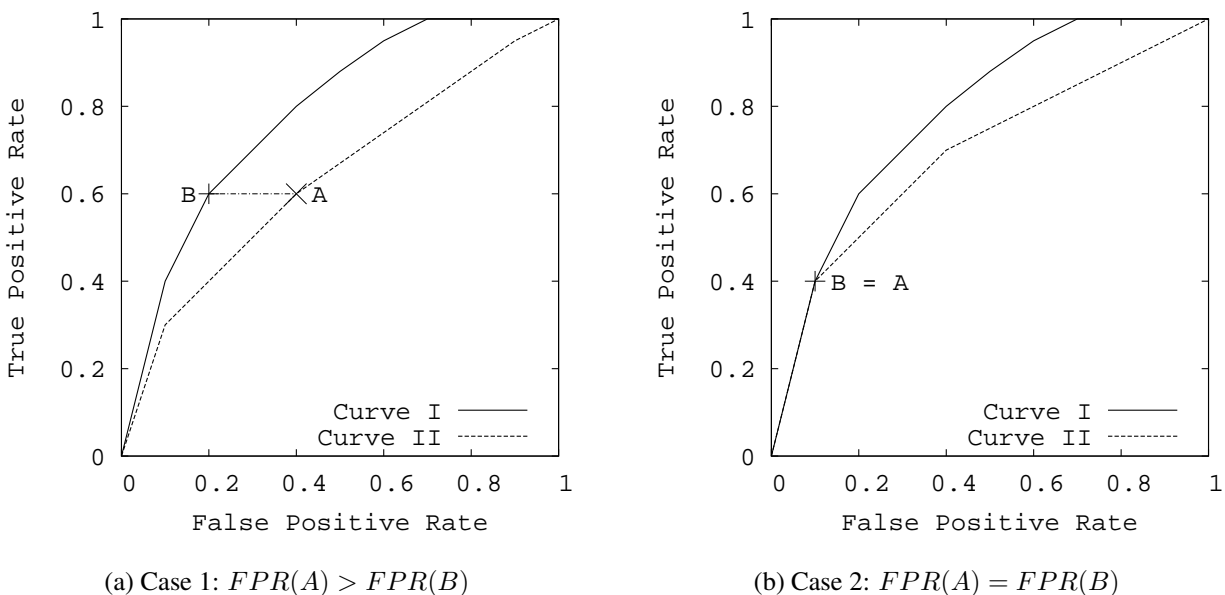


Figure 8.2 Two Cases for Claim 1 of Theorem 8.2.2

One important definition we need for our next theorem is the notion that one curve dominates another curve, “meaning that all other...curves are beneath it or equal to it (Provost et al., 1998).”

**Theorem 8.2.2.** *For a fixed number of positive and negative examples, one curve dominates a second curve in ROC space if and only if the first dominates the second in Precision-Recall space.*

**Proof.**

**Claim 1 ( $\Rightarrow$ ): If a curve dominates in ROC space then it dominates in PR space.** Proof by contradiction. Suppose we have curve I and curve II (as shown in Figure 8.2) such that curve I dominates in ROC space, yet, once we translate these curves in PR space, curve I no longer dominates. Since curve I does not dominate in PR space, there exists some point  $A$  on curve II such that the point  $B$  on curve I with identical Recall has lower Precision. In other words,  $PRECISION(A) > PRECISION(B)$  yet  $RECALL(A) = RECALL(B)$ . Since  $RECALL(A) = RECALL(B)$  and Recall is identical to  $TPR$ , we have that  $TPR(A) = TPR(B)$ . Since curve I dominates curve II in ROC space  $FPR(A) \geq FPR(B)$ . Remember that total positives and total negatives are fixed

and since  $TPR(A) = TPR(B)$ :

$$TPR(A) = \frac{TP_A}{\text{Total Positives}}$$

$$TPR(B) = \frac{TP_B}{\text{Total Positives}}$$

we now have  $TP_A = TP_B$  and thus denote both as  $TP$ . Remember that  $FPR(A) \geq FPR(B)$  and

$$FPR(A) = \frac{FP_A}{\text{Total Negatives}}$$

$$FPR(B) = \frac{FP_B}{\text{Total Negatives}}$$

This implies that  $FP_A \geq FP_B$  because

$$PRECISION(A) = \frac{TP}{FP_A + TP}$$

$$PRECISION(B) = \frac{TP}{FP_B + TP}$$

we now have that  $PRECISION(A) \leq PRECISION(B)$ . But this contradicts our original assumption that  $PRECISION(A) > PRECISION(B)$ .

**Claim 2 ( $\Leftarrow$ ): If a curve dominates in PR space then it dominates in ROC space.** Proof by contradiction. Suppose we have curve I and curve II (as shown in Figure 8.3) such that curve I dominates curve II in PR space, but once translated in ROC space curve I no longer dominates. Since curve I does not dominate in ROC space, there exists some point  $A$  on curve II such that point  $B$  on curve I with identical  $TPR$  yet  $FPR(A) < TPR(B)$ . Since  $RECALL$  and  $TPR$  are the same, we get that  $RECALL(A) = RECALL(B)$ . Because curve I dominates in PR space we know that  $PRECISION(A) \leq PRECISION(B)$ . Remember that  $RECALL(A) = RECALL(B)$  and

$$RECALL(A) = \frac{TP_A}{\text{Total Positives}}$$

$$RECALL(B) = \frac{TP_B}{\text{Total Positives}}$$

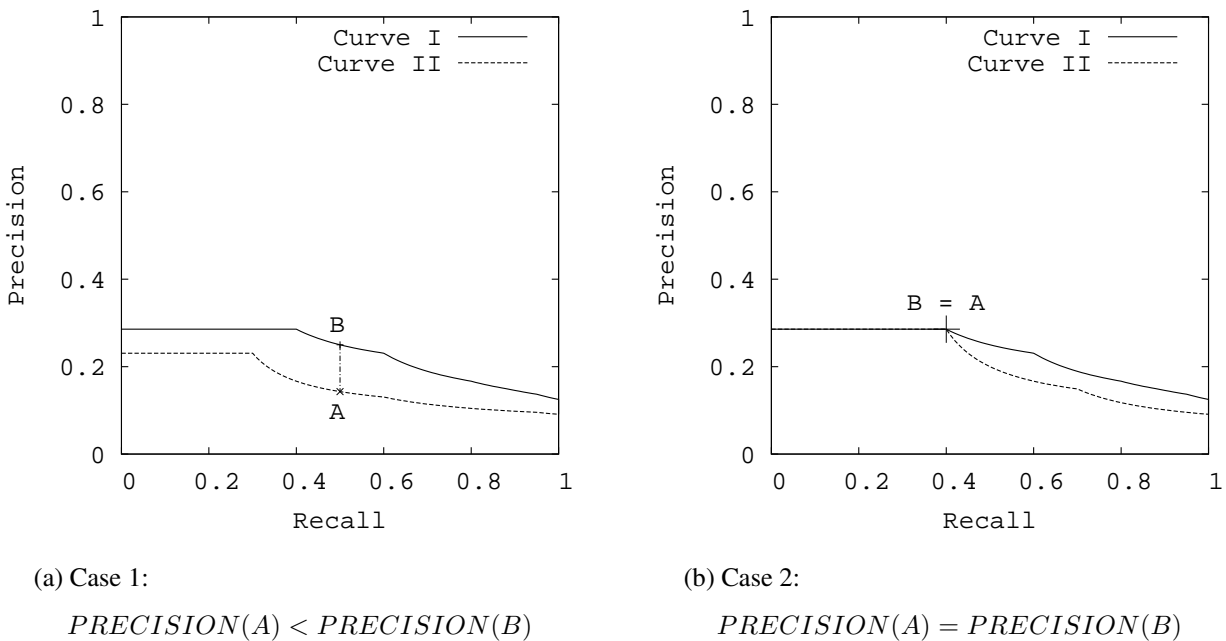


Figure 8.3 Two Cases of Claim 2 of Theorem 8.2.2

We know that  $TP_A = TP_B$ , so we will now denote them simply as  $TP$ . Because  $PRECISION(A) \leq PRECISION(B)$  and

$$PRECISION(A) = \frac{TP}{TP + FP_A}$$

$$PRECISION(B) = \frac{TP}{TP + FP_B}$$

we find that  $FP_A \geq FP_B$ . Now we have

$$FPR(A) = \frac{FP_A}{\text{Total Negatives}}$$

$$FPR(B) = \frac{FP_B}{\text{Total Negatives}}$$

This implies that  $FPR(A) \geq FPR(B)$  and this contradicts our original assumption that  $FPR(A) < FPR(B)$ .  $\square$

In ROC space the convex hull is a crucial idea. Given a set of points in ROC space, the convex hull must meet the following three criteria:

1. Linear interpolation is used between adjacent points.

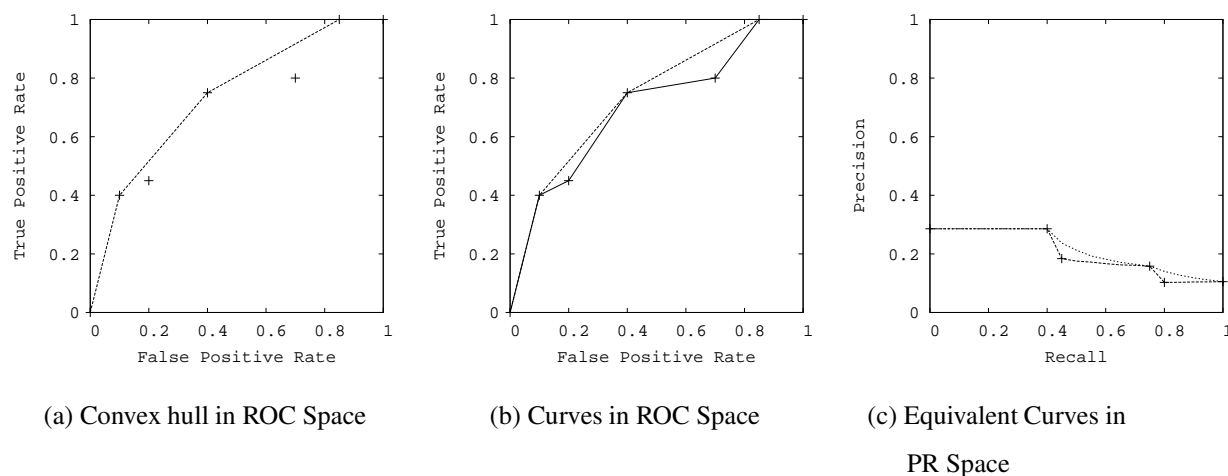


Figure 8.4 Convex hull and its PR analog dominate the naïve method for curve construction in each space. Note that this achievable PR curve is not a true convex hull due to non-linear interpolation. Linear interpolation in PR space is typically not achievable.

2. No point lies above the final curve.
3. For any pair of points used to construct the curve, the line segment connecting them is equal to or below the curve.

Figure 8.4(a) shows an example of a convex hull in ROC space. For a detailed algorithm of how to efficiently construct the convex hull, see Cormen et al. (1990).

In PR space, there exists an analogous curve to the convex hull in ROC space, which we call the achievable PR curve, although it cannot be achieved by linear interpolation. The issue of dominance in ROC space is directly related to this convex hull analog.

**Corollary 8.2.1.** *Given a set of points in PR space, there exists an achievable PR curve that dominates the other valid curves that could be constructed with these points.*

**Proof.** First, convert the points into ROC space (Theorem 3.1), and construct the convex hull of these points in ROC space. By definition, the convex hull dominates all other curves that could be constructed with those points when using linear interpolation between the points. Thus converting the points of the ROC convex hull back into PR space will yield a curve that dominates in PR space

as shown in Figures 8.4(b) and 8.4(c). This follows from Theorem 8.2.2. The achievable PR curve will exclude exactly those points beneath the convex hull in ROC space.  $\square$

The convex hull in ROC space is the best legal curve that can be constructed from a set of given ROC points. Many researchers, ourselves included, argue that PR curves are preferable when presented with highly-skewed datasets. Therefore it is surprising that we can find the achievable PR curve (the best legal PR curve) by first computing the convex hull in ROC space and then converting that curve into PR space. Thus the best curve in one space gives you the best curve in the other space.

An important methodological issue must be addressed when building a convex hull in ROC space or an achievable curve in PR space. When constructing a ROC curve (or PR curve) from an algorithm that outputs a probability, the following approach is usually taken: first find the probability that each test set example is positive, next sort this list and then traverse the sorted list in ascending order. To simplify the discussion, let  $class(i)$  refer to the true classification of the example at position  $i$  in the array and  $prob(i)$  refer to the probability that the example at position  $i$  is positive. For each  $i$  such that  $class(i) \neq class(i + 1)$  and  $prob(i) < prob(i + 1)$ , create a classifier by calling every example  $j$  such that  $j \geq i + 1$  positive and all other examples negative.

Thus each point in ROC space or PR space represents a specific classifier, with a threshold for calling an example positive. Building the convex hull can be seen as constructing a new classifier, as one picks the best points. Therefore it would be methodologically incorrect to construct a convex hull or achievable PR curve by looking at performance on the test data and then constructing a convex hull. To combat this problem, the convex hull must be constructed using a tuning set as follows: First, use the method described above to find a candidate set of thresholds on the tuning data. Then, build a convex hull over the tuning data. Finally use the thresholds selected on the tuning data, when building an ROC or PR curve for the test data. While this test-data curve is not guaranteed to be a convex hull, it preserves the split between training data and testing data.

### 8.3 Interpolation and AUC

A key practical issue to address is how to interpolate between points in each space. It is straightforward to interpolate between points in ROC space by simply drawing a straight line connecting the two points. One can achieve any level of performance on this line by flipping a weighted coin to decide between the classifiers that the two end points represent.

However, in Precision-Recall space, interpolation is more complicated. As the level of Recall varies, the Precision does not necessarily change linearly due to the fact that  $FP$  replaces  $FN$  in the denominator of the Precision metric. In these cases, linear interpolation is a mistake that yields an overly-optimistic estimate of performance. Corollary 8.2.1 shows how to find the achievable PR curve by simply converting the analogous ROC convex hull; this yields the correct interpolation in PR space. However, a curve consists of infinitely many points, and thus we need a practical, approximate method for translation. We expand here on the method proposed by Goadrich et al. (2004) to approximate the interpolation between two points in PR space.

Remember that any point  $A$  in a Precision-Recall space is generated from the underlying true positive ( $TP_A$ ) and false positive ( $FP_A$ ) counts. Suppose we have two points,  $A$  and  $B$  which are far apart in Precision-Recall space. To find some intermediate values, we must interpolate between their counts  $TP_A$  and  $TP_B$ , and  $FP_A$  and  $FP_B$ . We find out how many negative examples it takes to equal one positive, or the local skew, defined by  $\frac{FP_B - FP_A}{TP_B - TP_A}$ . Now we can create new points  $TP_A + x$  for all integer values of  $x$  such that  $1 \leq x \leq TP_B - TP_A$ , i.e.  $TP_A + 1, TP_A + 2, \dots, TP_B - 1$ , and calculate corresponding FP by linearly increasing the false positives for each new point by the local skew. Our resulting intermediate Precision-Recall points will be

$$\left( \frac{TP_A + x}{\text{Total Pos}}, \frac{TP_A + x}{TP_A + x + FP_A + \frac{FP_B - FP_A}{TP_B - TP_A} x} \right).$$

For example, suppose we have a dataset with 20 positive examples and 2000 negative examples. Let  $TP_A = 5$ ,  $FP_A = 5$ ,  $TP_B = 10$ , and  $FP_B = 30$ . Table 8.3 shows the proper interpolation of the intermediate points between  $A$  and  $B$ , with the local skew of 5 negatives for every 1 positive. Notice how the resulting Precision interpolation is not linear between 0.50 and 0.25.

|   | TP | FP | REC  | PREC  |
|---|----|----|------|-------|
| A | 5  | 5  | 0.25 | 0.500 |
| . | 6  | 10 | 0.30 | 0.375 |
| . | 7  | 15 | 0.35 | 0.318 |
| . | 8  | 20 | 0.40 | 0.286 |
| . | 9  | 25 | 0.45 | 0.265 |
| B | 10 | 30 | 0.50 | 0.250 |

Table 8.1 Correct interpolation between two points in PR space for a dataset with 20 positive and 2000 negative examples.

Often, the area under the curve is used as a simple metric to define how an algorithm performs over the whole space (Bradley, 1997; Davis et al., 2005b; Goadrich et al., 2004; Kok & Domingos, 2005; Macskassy & Provost, 2005; Singla & Domingos, 2005). The area under the ROC curve (AUC-ROC) can be calculated by using the trapezoidal areas created between each ROC point, and is equivalent to the Wilcoxon-Mann-Whitney statistic (Cortes & Mohri, 2003). By including our intermediate PR points, we can now use the composite trapezoidal method to approximate the area under the PR curve (AUC-PR).

The effect of incorrect interpolation on the AUC-PR is especially pronounced when two points are far away in Recall and Precision and the local skew is high. Consider a curve (Figure 8.5) constructed from a single point of  $(0.02, 1)$ , and extended to the endpoints of  $(0, 1)$  and  $(1, 0.008)$  as described above (for this example, our dataset contains 433 positives and 56,164 negatives). Interpolating as we have described would have an AUC-PR of 0.031; a linear connection would severely overestimate with an AUC-PR of 0.50.

Now that we have developed interpolation for PR space, we can give the complete algorithm for finding the achievable PR curve. First, we find the convex hull in ROC space (Corollary 3.1). Next, for each point selected by the algorithm to be included in the hull, we use the confusion matrix that defines that point to construct the corresponding point in PR space (Theorem 3.1). Finally, we perform the correct interpolation between the newly created PR points.

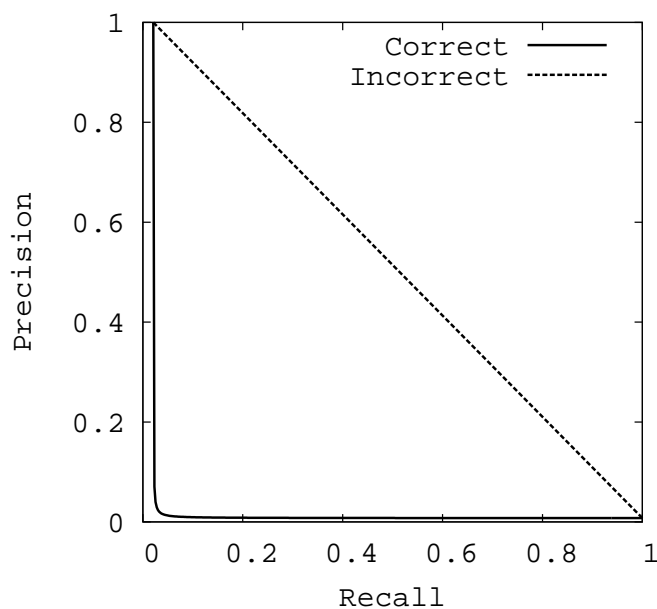
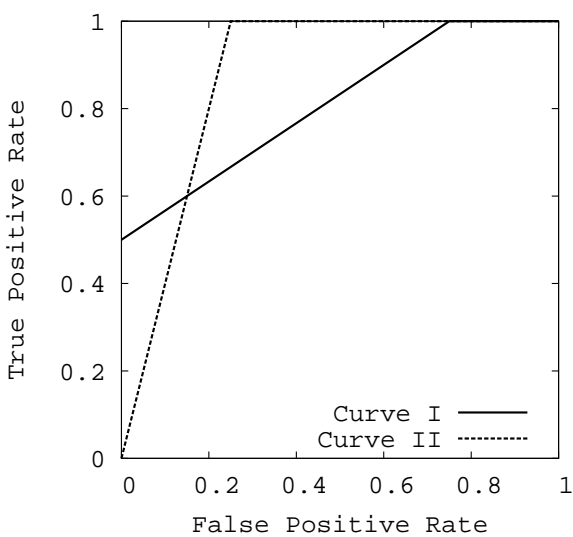
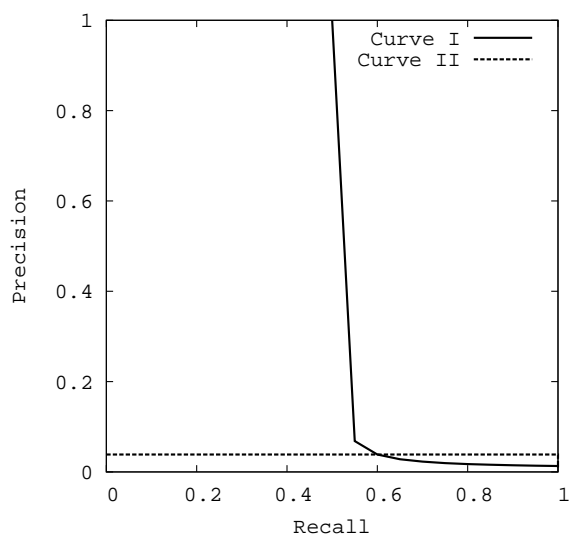


Figure 8.5 The Effect of Incorrect Interpolation in PR Space



(a) Comparing AUC-ROC for Two Algorithms



(b) Comparing AUC-PR for Two Algorithms

Figure 8.6 Difference in Optimizing Area Under the Curve in Each Space

## 8.4 Optimizing Area Under the Curve

Several researchers have investigated using AUC-ROC to inform the search heuristics of their algorithms. Ferri et al. (2002) alter decision trees to use the AUC-ROC as their splitting criterion,

Cortes and Mohri (2003) show that the boosting algorithm RankBoost (Freund et al., 1998) is also well-suited to optimize the AUC-ROC, Joachims (2005) presents a generalization of Support Vector Machines which can optimize AUC-ROC among other ranking metrics, Prati and Flach (2005) use a rule selection algorithm to directly create the convex hull in ROC space, and both Yan et al. (2003) and Herschtal and Raskutti (2004) explore ways to optimize the AUC-ROC within neural networks. Also, ILP algorithms such as Aleph (Srinivasan, 2001) can be changed to use heuristics related to ROC or PR space, at least in relation to an individual rule.

Knowing that a convex hull in ROC space can be translated into the achievable curve in Precision-Recall space leads to another open question: do algorithms which optimize the AUC-ROC also optimize the AUC-PR? Unfortunately, the answer generally is no, and we prove this by the following counter-example. Figure 8.6(a) shows two overlapping curves in ROC space for a domain with 20 positive examples and 2000 negative examples, where each curve individually is a convex hull. The AUC-ROC for curve I is 0.813 and the AUC-ROC for curve II is 0.875, so an algorithm optimizing the AUC-ROC and choosing between these two rankings would choose curve II. However, Figure 8.6(b) shows the same curves translated into PR space, and the difference here is drastic. The AUC-PR for curve I is now 0.514 due to the high ranking of over half of the positive examples, while the AUC-PR for curve II is far less at 0.038, so the direct opposite choice of curve I should be made to optimize the AUC-PR. This is because in PR space the main contribution comes from achieving a lower Recall range with higher Precision. Nevertheless, based on Theorem 8.2.2 ROC curves are useful in an algorithm that optimizes AUC-PR. An algorithm can find the convex hull in ROC space, convert that curve to PR space for an achievable PR curve, and score the classifier by the area under this achievable PR curve.

## 8.5 Chapter Summary

This work makes four important contributions. First, for any dataset, the ROC curve and PR curve for a given algorithm contain the same points. This equivalence leads to the surprising theorem that a curve dominates in ROC space if and only if it dominates in PR space. Second, as a corollary to the theorem we show the existence of the PR space analog to the convex hull in

ROC space, which we call an achievable PR curve. Remarkably, when constructing the achievable PR curve one discards exactly the same points omitted by the convex hull in ROC space. Consequently, we can efficiently compute the achievable PR curve. Third, we show that simple linear interpolation is insufficient between points in PR space. Finally, we show that an algorithm that optimizes the area under the ROC curve is not guaranteed to optimize the area under the PR curve.

Additionally, we have made code publicly available for correct PR space interpolation and computing AUC in both PR and ROC space. The code can be found at:

<http://pages.cs.wisc.edu/~richm/programs/AUC/>

## Chapter 9

### Conclusions and Future Work

#### 9.1 Summary

The bulk of this dissertation developed view learning for SRL. SRL algorithms provide a substantial extension to existing statistical learning algorithms, such as Bayesian networks, by permitting statistical learning to be applied directly to relational databases with multiple tables. Nevertheless, the schemas for relational databases often are defined based on criteria other than effectiveness of machine learning. If a schema is not the most appropriate for a given learning task, it may be necessary to change it—by defining a new view—before applying other SRL techniques. View learning filled this gap by providing an automated capability to alter the schema of a database through the addition of derived fields and tables.

We started by working on determining identity equivalence, otherwise known as alias detection, in intelligence analysis domains. In intelligence analysis, textual similarity between names is often uninformative as individuals purposely attempt to disguise their identity. Consequently, we needed to rely on attributes and contextual information to detect aliases. We proposed a two-step methodology for detecting aliases. The first step employed an ILP system to learn rules that predict whether two identifiers refer to the same person. The second step represented each learned rule as an attribute in a Bayesian network. We evaluated our approach on series of synthetic datasets developed as part of a U.S. Air Force project. Empirically, this approach yielded significantly more accurate models than treating each rule as a classifier in an unweighted voting scheme.

Working on the ILP-based feature construction led us to realize that this idea could be applied in a broader context within SRL. Specifically, it could alleviate many SRL approaches' inability

to alter a given database schema. It is possible to augment the database schema by treating each learned rule as an additional field definition. We proposed two different algorithms for learning new fields.

The first algorithm was a multi-step framework for defining new fields. The first step used an ILP algorithm to construct a large set of relational features. The second step selected which relational features to include in the model. The final step constructed a statistical model. We evaluated this process in the specific context of providing decision support to radiologists who interpret mammograms. For this task, the American College of Radiology has designed a database schema to standardize data collection for mammography practices in the United States. This schema contains one record per abnormality. However, data in other rows of the table may also be relevant: radiologists may also consider other abnormalities on the same mammogram or previous mammograms when making a diagnosis. The need to consider this relational information makes this task a natural fit for SRL.

We evaluated our algorithm on a collection of radiology reports obtained from an actual mammography practice. Empirically, SRL techniques significantly improved over propositional methods for this domain. Nevertheless, the improvement obtained through the use of aggregation, as might be performed for example by a PRM, was roughly equivalent to view learning. View learning did generate features that were interesting and useful for the radiology community. A radiologist collaborator reviewed several features and was particularly intrigued by one that suggested a hitherto unknown relationship between malignancy and high density masses. In general, mass density was not previously thought to be a highly predictive feature.

The second algorithm we proposed for view learning integrates the multi-step algorithm into one cohesive process by *constructing the classifier during the rule learning procedure*. This methodology, called *Score As You Use* or *SAYU*, addressed the following two weaknesses of the multi-step algorithm. First, choosing how many rules to include in the final model is a difficult tradeoff between completeness and overfitting. Second, the best rules according to coverage may not yield the most accurate classifier. SAYU overcame these drawbacks by scoring each potential

view by how much it improves the accuracy of the classification model. To score each potential view, or rule, SAYU re-computed the model and checked whether it fits the data significantly better than the model that did not use the candidate rule. SAYU only retained those views that significantly improved the fit of the model to the data.

Employing the SAYU approach, we learned a new model from the mammography data that yielded superior performance, as measured by area under the precision-recall curve. The model's performance exceeded the performances of a hand-crafted Bayesian network system, as well as the performances of standard Bayesian network structure learners and other SRL systems. The SAYU algorithm was successfully applied to other classification problems including predicting a graduate student's advisor and which genes are involved in metabolism.

Next, we demonstrated the feasibility and utility of extending SAYU to other types of task, such as multiple-instance, real-valued prediction. The problem of predicting 3D-QSAR, a well-known family of tasks in research into drug design, motivated this work. We improved upon a prior ILP-plus-regression approach in two important ways. Our first improvement involved explicitly representing the multiple-instance nature of the task through the use of multiple-instance regression. Our second improvement involved integrating the feature invention and model building step by adapting the SAYU approach previously discussed to this regression task. We evaluated our algorithm with three activity prediction problems: thermolysin inhibitors, dopamine agonists and thrombin inhibitors. We observed that each extension by itself improved the accuracy of our real-valued predictions. Further, our proposed approach, which used both extensions, resulted in more accurate predictions than either extension by itself. Our approach also discovered and used biologically relevant pharmacophores when predicting activity.

The SAYU-VISTA system was the final piece of work on view learning. It offered three significant extensions. First, it learned predicates that have a higher-arity than the target concept, some of which capture many-to-many relations and require a new table to represent. Second, it learned predicates that apply to different types than the target concept, allowing for the invention of predicates that are not merely approximations to the target concept. Third, it permitted newly-invented predicates to be used in the invention of other new relations. SAYU-VISTA performed

a type of predicate invention, because it constructed arbitrary clauses whose heads do not have to unify with the examples and whose heads may have arities and types different from the examples. SAYU-VISTA constrained the search by requiring the invented predicates to be of immediate value to the statistical learner in order to be retained for future use. The empirical success of the algorithm indicates that this efficacy test is a successful constraint on predicate invention. Empirically, SAYU-VISTA led to more accurate models than both the SAYU algorithm and Markov Logic Networks (MLNs), another leading SRL framework. Furthermore, SAYU-VISTA constructed useful and relevant predicates, such as consistently discovering the *CoAuthor* and *TA'ed for* relations in the domain where the task was to predicate a graduate student's advisor.

Finally, the preceding work led us to use precision-recall (PR) curves as an evaluation metric. We investigated the connection between these two measures to ascertain whether some of the interesting properties of ROC curves also hold for PR curves. We demonstrated four important facts about ROC space and PR space. First, for any dataset, and hence a fixed number of positive and negative examples, the ROC curve and PR curve for a given algorithm contain the "same points." Based on this equivalence for ROC and PR curves, we proved that a curve dominates in ROC space if and only if it dominates in PR space. Second, we introduced the PR space analog to the convex hull in ROC space, which we called the achievable PR curve. Due to the equivalence of these two spaces we can efficiently compute the achievable PR curve. Third we offered theoretical justification for the fact that it is insufficient to linearly interpolate between points in PR space. Finally, we proved that an algorithm that optimizes the area under the ROC curve is not guaranteed to optimize the area under the PR curve.

## 9.2 Future Work

The advent and prevalence of high-throughput techniques, such as gene-expression microarrays, single-nucleotide polymorphism (SNP) chips and high-throughput screening of molecules for biological activity, has greatly increased the quantity of biological data available. Furthermore, it has allowed people to collect data for the same phenomena from multiple sources. The most natural place to store this burgeoning collection of diverse data is in relational databases with multiple

tables, one for each data type. SRL's ability to learn from multiple-relational tables and integrate multiple data sources into one cohesive model will increase its importance in the coming years. This wealth of diverse data types requires the development of novel computational techniques. The following are three concrete directions we intend to pursue.

**Mining combined biological and clinical databases.** In the not so distant future, it will be possible to have access to an unprecedented amount of clinical and biological data. With the advent of electronic medical records, researchers could have ready access to clinical records for patients participating in various research studies. Furthermore, single-nucleotide polymorphism and gene-expression microarray data may be available for a subset of these individuals. Access to this combination of data will allow researchers to pose and investigate the following types of questions. Can one develop a model to predict the efficacy of a potential drug for a given individual? Can clinical and genetic data provide insight into which individuals will have adverse reactions to a drug? We are currently involved with a project that is trying to address similar questions.

Answering these types of questions will require the abilities to reason about uncertainty, to consider complex relationships and to integrate multiple data sources. SRL provides all three of these capabilities. In addition, the ability developed in this dissertation to automatically identify novel patterns and relationships within the data will be among the chief assets SRL will bring to the task. The challenges this domain poses will help drive the next round of innovation in SRL technology. A key issue will be scaling up SRL algorithms to handle the large amounts of data that will be available. It will be crucial to develop algorithms that can efficiently work with large datasets. Furthermore, the characteristics of the data will be significantly different than current testbeds. First, in this type of setting, a researcher will have access to very few patients or examples. It will be crucial to address the issue of overfitting to ensure that learned models generalize to unseen examples. Second, SRL often focuses on domains that have many relations, but each relation only has a few attributes. The medical domain will be much more feature-centric. For example, microarray experiments yield information about tens of thousands of features. Meeting

these challenges will be crucial for making SRL viable in these types of domains.

### **Incorporating negative information.**

Often in biology, information about events that do not or should not occur can provide useful information to a learning system. Unfortunately, the number of events that do not occur is quite large, making it difficult to discern the relevant negative information. For a concrete example, consider the notion of “excluded volumes” for pharmacophores. Excluded volumes place additional restrictions on potential pharmacophores by declaring that no part of the molecule can occupy a particular region or volume. A potential pharmacophore may have all the relevant features in order to be active and yet, in practice, does not exhibit the desired activity level. A pharmacophore only models a 3D substructure of the molecule, and it is possible that the part of the molecule that is not modeled interferes with its ability to bind to the target protein. For example, the unmodeled portion of the pharmacophore could physically occupy part of the binding area for the target protein. It is also possible that the unconstrained part of the pharmacophore could have an element, such as a hydrophobic group, close enough to the binding site that it interferes with the molecule’s ability to bind to the target protein. Therefore, it could potentially be useful to include information about excluded volumes in the pharmacophore search process.

The relational learning approach taken in our previous work readily lends itself to the ability to incorporate background knowledge about excluded volumes. Unfortunately, it is difficult and time consuming to obtain information about excluded volumes. Often, it is unclear which regions should be declared as excluded volumes. If the excluded volumes are known, the information has to be introduced by hand as background knowledge for the learning algorithm. Therefore, an important innovation would be the ability to automatically generate and incorporate excluded volumes during the search process. This is an extraordinarily difficult problem to address because it requires being able to determine a region a molecule should not occupy. The fact that an active molecule does not occupy a given region of space does not imply that a requirement for activity is that the molecule not occupy the given region. The trick becomes generating and selecting the

relevant regions that a molecule should not occupy in order to be active. We would like to extend our drug activity prediction framework to automatically generate and incorporate information about excluded volumes during the search process. One possible approach would be to examine a pharmacophore that covers a reasonably large number of active molecules, but also multiple inactive molecules. Using this pharmacophore, it may be possible to extract an excluded volume. If the inactives covered by this pharmacophore all occupied part of a volume that none of the actives occupy, then this volume could be a candidate for an excluded volume. You could further evaluate the candidate by requiring that it satisfy other conditions, such as its volume being less than some predetermined threshold. Candidates that pass all the tests would be incorporated into the search space.

**Addressing inter-observer variability.** Oftentimes in biology or medicine, data are collected through a process which is subject to the variability of a data collector or an observer. For example, the outcome of microarray experiments can be influenced by laboratory conditions or personnel. This is even more true of experiments involving mass spectrometry. The problem will be particularly acute when combining experiments done at different laboratories. In mammography, the quality of the interpretation of a mammogram is affected by the variability in the training, skill and experience of the radiologist who reads it. An important research direction will be to account for variability. For a concrete example, I will focus on the mammography domain. An assumption underlying our prior work is that if two radiologists read the same mammogram, they will generate identical reports about it. In reality, this is not the case, especially for mammograms that contain suspicious findings. One way to account for this weakness would be through the use of collective classification. In the ordinary use of a classifier obtained by machine learning or data mining, we run the classifier on one record, or test case, at a time, and we fill in the value of one field or variable at a time. *Collective classification* (Jensen et al., 2004; Neville & Jensen, 2000) is an alternative in which we use a model or classifier to label all unknown fields in all records or test cases together, or collectively, with each label taking the other labels into account.

We can use collective classification to improve predictions in the following manner. We can use our SRL model to analyze the abnormalities in a held-out validation set. Some abnormalities in the validation set will themselves be misclassified. If a larger than expected fraction of the mislabeled abnormalities was studied by a particular radiologist, this may be indicative that this radiologist is not labeling features, such as calcification types, in a manner consistent with the other radiologists. If we add a table that associates with each radiologist a feature indicating performance, then we can estimate this feature for each radiologist from how accurately our SRL model performs on his or her cases. The process can be iterated, so that our SRL model for labeling abnormalities takes into account the radiologist. In this way, when labeling an abnormality in the test set, we allow the set of previous mammograms analyzed by that radiologist to influence our label.

The preceding procedure can be carried even further, so that the gold standard outcomes of prior mammograms can be used to update our beliefs about the individual fields in a current mammogram. For example, assume a specific radiologist consistently found pleomorphic calcifications across all mammograms, regardless of whether the abnormality is benign or malignant. However, since pleomorphic calcifications are more indicative of malignancy, the SRL model might assign the abnormality a higher probability of being malignant than is actually warranted given the radiologist's track record. Through collective classification, we can decrease the weight placed on this feature and potentially lower the posterior probability of malignancy for the current abnormality.

## Bibliography

- Adams, C. P., & Brantner, V. V. (2004). Estimating the costs of new drug development: Is it really \$802m? [http://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=640563](http://papers.ssrn.com/sol3/papers.cfm?abstract_id=640563).
- Adibi, J., Chalupsky, H., Melz, E., & Valente, A. (2004). The kojak group finder: Connecting the dots via integrated knowledge-based and statistical reasoning. *Proceedings of the Sixteenth Innovative Applications of Artificial Intelligence Conference* (pp. 800–807). San Jose, California.
- Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large data bases. *Proceedings of the ACM SIGMOD Conference on Management of Data* (pp. 207–216). Washington, D.C.
- Alphonse, E., & Rouveirol, C. (2000). Lazy propositionalisation for relational learning. *Proceedings of the 14th European Conference on Artificial Intelligence* (pp. 256–260). Berlin, Germany.
- American College of Radiology (2004). *Breast imaging reporting and data system (BI-RADS)*. Reston VA.
- Bilenko, M., & Mooney, R. (2003). Adaptive duplicate detection using learnable string similarity measures. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 39–48). Washington D.C.
- Bockhorst, J., & Craven, M. (2005). Markov networks for detecting overlapping elements in sequence data. *Advances in Neural Information Processing Systems 17*. MIT Press.
- Bradley, A. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30, 1145–1159.

- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24, 123–140.
- Brint, A., & Willett, P. (1987). Algorithms for the identification of three-dimensional maximal common substructures. *Journal of Chemical Information and Computer Science*, 27, 152–158.
- Brown, M., Houn, F., Sickles, E., & Kessler, L. (1995). Screening mammography in community practice: Positive predictive value of abnormal findings and yield of follow-up diagnostic procedures. *American Journal of Roentgenology*, 165, 1373–1377.
- Buechi, M., Borthwick, A., Winkel, A., & Goldberg, A. (2003). Cluemaker: A language for approximate record matching. *Proceedings of the 8th International Conference on Data Quality* (pp. 207–223). Cambridge, Massachusetts.
- Bunescu, R., Ge, R., Kate, R., Marcotte, E., Mooney, R., Ramani, A., & Wong, Y. (2005). Comparative experiments on learning information extractors for proteins and their interactions. *Journal of Artificial Intelligence in Medicine*, 33, 139–155.
- Burnside, E., Davis, J., Costa, V. S., Dutra, I. C., Kahn, C., Fine, J., & Page, D. (2005). Knowledge discovery from structured mammography reports using inductive logic programming. *Proceedings of the American Medical Informatics Association Fall Symposium* (pp. 96–100). Washington, D.C.
- Burnside, E., Pan, Y., Kahn, C., Shaffer, K., & Page, D. (2004a). Training a probabilistic expert system to predict the likelihood of breast cancer using a large dataset of mammograms (abstr). *Proceedings of the Radiological Society of North America Scientific Assembly and Annual Meeting Program*. Oak Brook, Illinois.
- Burnside, E., Rubin, D., & Shachter, R. (2000). A Bayesian network for screening mammography. *Proceedings of the American Medical Informatics Association Fall Symposium* (pp. 106–110). Los Angeles, California.

- Burnside, E., Rubin, D., & Shachter, R. (2004b). Using a Bayesian network to predict the probability and type of breast cancer represented by microcalcifications on mammography. *Proceedings of the 11th World Congress on Medical Informatics* (pp. 13–18). San Francisco, California.
- Cheng, J., Hatzis, C., Hayashi, H., Krogel, M.-A., Morishita, S., Page, D., & Sese, J. (2002). KDD Cup 2001 report. *SIGKDD Explorations*, 3, 47–64.
- Cohen, W. W., & Richman, J. (2002). Learning to match and cluster large high-dimensional data sets for data integration. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 475–480). Edmonton, Alberta.
- Connolly, D. (1993). Constructing hidden variables in Bayesian networks via conceptual clustering. *Proceedings of the Tenth International Conference on Machine Learning* (pp. 65–72). Amherst, Massachusetts.
- Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9, 309–347.
- Cormen, T. H., Leiserson, Charles, E., & Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press.
- Cortes, C., & Mohri, M. (2003). AUC optimization vs. error rate minimization. *Advances in Neural Information Processing Systems 15*. MIT Press.
- Cramer, R. D., Patterson, D. E., & Bunce, J. D. (1988). Comparative molecular field analysis (ComFA). Effect on binding of steroids to carrier proteins. *Journal of the American Chemical Society*, 110, 5959–5967.
- Craven, M., & Slattery, S. (2001). Relational learning with statistical predicate invention: Better models for hypertext. *Machine Learning*, 43, 97–119.
- Cussens, J. (2001). Parameter estimation in stochastic logic programs. *Machine Learning Journal*, 44, 245–271.

- Davis, J., Burnside, E., Dutra, I., Page, D., & Costa, V. S. (2005a). An integrated approach to learning Bayesian networks of rules. *Proceedings of the 16th European Conference on Machine Learning* (pp. 84–95). Porto, Portugal.
- Davis, J., Burnside, E., Dutra, I. C., Page, D., Ramakrishnan, R., Costa, V. S., & Shavlik, J. (2005b). View learning for statistical relational learning: With an application to mammography. *Proceedings of the 19th International Joint Conference on Artificial Intelligence* (pp. 677–683). Edinburgh, Scotland.
- Davis, J., Burnside, E., Dutra, I. C., Page, D., Ramakrishnan, R., Costa, V. S., & Shavlik, J. (2007a). Learning a new view of a database: With an application to mammography. In L. Getoor and B. Taskar (Eds.), *An Introduction to Statistical Relational Learning*. MIT Press.
- Davis, J., Burnside, E., Page, D., & Costa, V. S. (2006). View learning extended: Inventing new tables for statistical relational learning. *Proceeding of Open Problems in Statistical Relational Learning Workshop*. Pittsburgh, Pennsylvania.
- Davis, J., Costa, V. S., Ray, S., & Page, D. (2007b). An integrated approach to feature construction and model building for drug activity prediction. *Proceedings of the 24th International Conference on Machine Learning*. Corvallis, Oregon.
- Davis, J., Dutra, I. C., Page, D., & Santos Costa, V. (2005c). Establishing entity equivalence in multi-relation domains. *Proceedings of the International Conference on Intelligence Analysis*. Vienna, Virginia.
- Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd International Conference on Machine Learning* (pp. 233–240). Pittsburgh, Pennsylvania.
- Davis, J., Ong, I., Struyf, J., Burnside, E., Page, D., & Costa, V. S. (2007c). Change of representation for statistical relational learning. *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 2719–2726). Hyderabad, India.

- Davis, J., Santos Costa, V., Ong, I. M., Page, D., & Dutra, I. C. (2004). Using Bayesian classifiers to combine rules. *Proceeding of the 3rd Workshop on Multi-Relational Data Mining*. Seattle, Washington.
- Dietterich, T., Lathrop, R., & Lozano-Perez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89, 31–71.
- DiMasi, J. A., Hansen, R. W., & Grabowski, H. G. (2003). The price of innovation: New estimates of drug development costs. *Journal of Health Economics*, 22, 151–185.
- Drummond, C., & Holte, R. (2000). Explicitly representing expected cost: An alternative to ROC representation. *Proceeding of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 198–207). Boston, Massachusetts.
- Drummond, C., & Holte, R. C. (2004). What ROC curves can't do (and cost curves can). *Proceedings of the 1st Workshop on ROC Analysis in AI* (pp. 19–26). Valencia, Spain.
- Dutra, I., Page, D., Santos Costa, V., & Shavlik, J. (2002). An empirical evaluation of bagging in inductive logic programming. *Proceeding of the 12th International Conference on Inductive Logic Programming* (pp. 48–65). Sydney, Australia.
- Eklund, G. W. (2000). Shortage of qualified breast imagers could lead to crisis. *Diagnostic Imaging*, 22, 31–33.
- Fellegi, I., & Sunter, A. (1969). Theory of record linkage. *Journal of the American Statistical Association*, 64, 1183–1210.
- Ferri, C., Flach, P., & Henrandez-Orallo, J. (2002). Learning decision trees using area under the ROC curve. *Proceedings of the 19th International Conference on Machine Learning* (pp. 139–146). Sydney, Australia.
- Fierens, D., Blockeel, H., Bruynooghe, M., & Ramon, J. (2005). Logical Bayesian networks and their relation to other probabilistic logical models. *Proceeding of the 15th International Conference on Inductive Logic Programming* (pp. 121–135). Bonn, Germany.

- Finn, P., Muggleton, S., Page, D., & Srinivasan, A. (1998). Pharmacophore discovery using the inductive logic programming system PROGOL. *Machine Learning*, 30, 241–270.
- Fisher, R. (1922). On the interpretation of  $\chi^2$  from contingency tables, and the calculation of P. *Journal of the Royal Statistical Society*, 85, 87–94.
- Fletcher, R. (1980). *Practical Methods of Optimization*, vol. 1: Unconstrained Optimization, chapter 3. John Wiley and Sons.
- Freund, Y., Iyer, R., Schapire, R., & Singer, Y. (1998). An efficient boosting algorithm for combining preferences. *Proceedings of the 15th International Conference on Machine Learning* (pp. 170–178). Madison, Wisconsin.
- Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian networks classifiers. *Machine Learning*, 29, 131–163.
- Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999a). Learning probabilistic relational models. *Proceedings of the 16th International Joint Conference on Artificial Intelligence* (pp. 1300–1309). Stockholm, Sweden.
- Friedman, N., Nachman, I., & Pe’er, D. (1999b). Learning Bayesian network structure from massive datasets: The “sparse candidate” algorithm. *Proceedings of the 17th Conference of Uncertainty in Artificial Intelligence* (pp. 206–215). Stockholm, Sweden.
- Getoor, L., Friedman, N., Koller, D., & Pfeffer, A. (2001). Learning probabilistic relational models. In S. Džeroski and N. Lavrač (Eds.), *Relational Data Mining*. Springer.
- Goadrich, M., Oliphant, L., & Shavlik, J. (2004). Learning ensembles of first-order clauses for recall-precision curves: A case study in biomedical information extraction. *Proceedings of the 14th International Conference on Inductive Logic Programming* (pp. 98–115). Porto, Portugal.
- Greiner, R., & Zhou, W. (2002). Structural extensions to logistic regression: Discriminative parameter learning of belief net classifiers. *Proceedings of the 18th National Conference on Artificial Intelligence* (pp. 167–173). Edmonton, Alberta.

- Grossman, D., & Domingos, P. (2004). Learning Bayesian network classifiers by maximizing conditional likelihood. *Proceedings of the 21st International Conference on Machine Learning* (pp. 361–368). Banff, Canada.
- Heckerman, D., Meek, C., & Koller, D. (2004). *Probabilistic entity-relationship models, PRMs, and plate models, technical report msr-tr-2004-30, microsoft research* (Technical Report). Microsoft Research.
- Herschtal, A., & Raskutti, B. (2004). Optimising area under the ROC curve using gradient descent. *Proceedings of the 21st International Conference on Machine Learning* (p. 49). Banff, Alberta.
- Hoche, S., & Wrobel, S. (2001). Relational learning using constrained confidence-rated boosting. *Proceeding of the 11th International Conference on Inductive Logic Programming* (pp. 51–64). Strasbourg, France.
- Hsiung, P., Moore, A., Neill, D., & Schneider, J. (2004). Alias detection in link data sets. Master's thesis, Carnegie Mellon University. Carnegie Mellon University.
- Jain, A., Dietterich, T., Lathrop, R., Chapman, D., Critchlow, R., Bauer, B., Webster, T., & Lozano-Pérez, T. (1994a). Compass: A shape-based machine learning tool for drug design. *Journal of Computer-Aided Molecular Design*, 8, 635–652.
- Jain, A., Koile, K., Bauer, B., & Chapman, D. (1994b). Compass: Predicting biological activities from molecular surface properties. *Journal of Medicinal Chemistry*, 37, 2315–2327.
- Jensen, D., Neville, J., & Gallagher, B. (2004). Why collective inference improves relational classification. *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 593–598). Seattle, Washington.
- Joachims, T. (2005). A support vector method for multivariate performance measures. *Proceedings of the 22nd International Conference on Machine Learning* (pp. 377–384). Bonn, Germany: ACM Press.

- Kahn, C., Roberts, L., Shaffer, K., & Haddawy, P. (1997). Construction of a Bayesian network for mammographic diagnosis of breast cancer. *Computers in Biology and Medicine*, 27, 19–29.
- Kemp, C., Tenenbaum, J., Griffiths, T., Yamada, T., & Ueda, N. (2006). Learning systems of concepts with an infinite relational model. *Proceedings of the 21st National Conference on Artificial Intelligence*. Boston, Massachusetts.
- Kersting, K., & Raedt, L. D. (2002). *Basic principles of learning Bayesian logic programs* (Technical Report). Institute for Computer Science, University of Freiburg, Germany.
- King, R. (1991). PROMIS: Experiments in machine learning and protein folding. In D. Michie (Ed.), *Machine intelligence 12*. Oxford University Press.
- Knobbe, A., de Haas, M., & Siebes, A. (2001). Propositionalisation and aggregates. *Proceeding of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 277–288). Freiburg, Germany.
- Kok, S., & Domingos, P. (2005). Learning the structure of Markov Logic Networks. *Proceedings of the 22nd International Conference on Machine Learning* (pp. 441–448). Bonn, Germany.
- Kok, S., & Domingos, P. (2007). Statistical predicate invention. *Proceedings of the 24th International Conference on Machine Learning*. Corvallis, Oregon.
- Kroegel, M.-A., & Wrobel, S. (2001). Transformation-based learning using multirelational aggregation. *Proceedings of the 11th Conference on Inductive Logic Programming* (pp. 142–155). Strausbourg, France.
- Landwehr, N., Kersting, K., & Raedt, L. D. (2005). nFOIL: Integrating Naive Bayes and FOIL. *Proceeding of the 20th National Conference on Artificial Intelligence* (pp. 795–800). Pittsburgh, Pennsylvania.
- Landwehr, N., Passerini, A., Raedt, L. D., & Frasconi, P. (2006). kFOIL: Learning simple relational kernels. *Proceedings of the 21st National Conference on Artificial Intelligence*. Boston, Massachusetts.

- Lavrač, N., & Džeroski, S. (1992). Inductive learning of relations from noisy examples. In S. Mugleton (Ed.), *Inductive Logic Programming*, 495–516. Academic Press.
- Lavrač, N., & Džeroski, S. (Eds.). (2001). *Relational Data Mining*. Springer.
- Lavrač, N., Džeroski, S., & Grobelnik, M. (1991). Learning non-recursive definitions of relations with LINUS. *Proceedings of the Fifth European Working Session on Learning* (pp. 265–281). Porto, Portugal.
- Lawrence, S., Giles, C. L., & Bollacker, K. (1999). Autonomous citation matching. *Proceedings of the 3rd International Conference on Autonomous Agents* (pp. 392–393). Seattle, Washington.
- Lloyd, J. (1984). *Foundations of Logic Programming*. Berlin, Germany: Springer-Verlag.
- Macskassy, S., & Provost, F. (2005). Suspicion scoring based on guilt-by-association, collective inference, and focused data access. *Proceedings of the International Conference on Intelligence Analysis*. Vienna, Virginia.
- Manning, C., & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.
- Marchand-Geneste, N., Watson, K., Alsberg, B., & King, R. (2002). New approach to pharmacophore mapping and QSAR analysis using inductive logic programming. Application to thermolysin inhibitors and glycogen phosphorylase *b* inhibitors. *Journal of Medicinal Chemistry*, 45, 399–409.
- Maron, O. (1998). *Learning from ambiguity*. Doctoral dissertation, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.
- Martin, Y., Bures, M., Danaher, E., DeLazzer, J., Lico, I., & Pavlik, P. (1993). A fast new approach to pharmacophore mapping and its application to dopaminergic and benzodiazepine agonists. *Journal of Computer-Aided Molecular Design*, 7, 83–102.

- McCallum, A., Nigam, K., Rennie, J., & Seymore, K. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval Journal*, 3, 127–163. [www.research.whizbang.com/data](http://www.research.whizbang.com/data).
- McCallum, A., & Wellner, B. (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. *Proceedings of IJCAI Workshop on Information Integration on the Web* (pp. 79–84). Acapulco, Mexico.
- McGaughey, G. B., & Mewshaw, R. E. (1999). Application of comparative molecular field analysis to dopamine d2 partial agonists. *Bioorganic Medical Chemistry*, 7, 2453–2456.
- Mewes, H. W., Frishman, D., Gruber, C., Geier, B., Haase, D., Kaps, A., Lemcke, K., Mannhaupt, G., Pfeiffer, F., Schüller, C., Stocker, S., & Weil, B. (2000). MIPS: A database for genomes and protein sequences. *Nucleic Acids Research*, 28, 37–40.
- Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D., & Kolobov, A. (2005). BLOG: Probabilistic models with unknown objects. *Proceedings of the 19th International Joint Conference on Artificial Intelligence* (pp. 1352–1359). Edinburgh, Scotland.
- Mitchell, T. M. (1997). *Machine Learning*. New York, New York: McGraw-Hill.
- Monge, A. E., & Elkan, C. (1996). The field matching problem: Algorithms and applications. *Proceeding of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 267–270). Portland, Oregon.
- Morton, T. S. (2000). Coreference for NLP applications. *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*. Hong Kong, China.
- Muggleton, S. (1995). Inverse entailment and Progol. *New Generation Computing*, 13, 245–286.
- Muggleton, S. (2000). Learning stochastic logic programs. *Electronic Transactions in Artificial Intelligence*, 4, 141–153.

- Muggleton, S., & Buntine, W. (1988). Machine invention of first-order predicates by inverting resolution. *Proceedings of the Fifth International Conference on Machine Learning* (pp. 339–352). Ann Arbor, Michigan.
- Neville, J., & Jensen, D. (2000). Iterative classification in relational data. *Proceedings of the AAAI-2000 Workshop on Learning Statistical Models from Relational Data..* Austin, Texas.
- Neville, J., & Jensen, D. (2004). Dependency networks for relational data. *Proceedings of the 4th IEEE International Conference on Data Mining* (pp. 170–177). Brighton, United Kingdom.
- Neville, J., Jensen, D., Friedland, L., & Hay, M. (2003). Learning relational probability trees. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 625–630). Washington, D.C.
- Pasula, H., Marthi, B., Milch, B., Russell, S., & Shpitser, I. (2003). Identity uncertainty and citation matching. *Advances in Neural Information Processing Systems 15* (pp. 1401–1408). MIT Press.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, California: Morgan Kaufmann.
- Perlich, C., & Provost, F. (2003). Aggregation-based feature invention and relational concept classes. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 167–176). Washington, D.C.
- Pfeffer, A. (2001). IBAL: A probabilistic rational programming language. *Proceeding of the 17th International Joint Conference on Artificial Intelligence* (pp. 733–740). Seattle, Washington.
- Pompe, U., & Kononenko, I. (1995). Naïve Bayesian classifier within ILP-R. *Proceeding of the 4th International Workshop on Inductive Logic Programming* (pp. 417–436). Toyko, Japan.
- Popescul, A., & Ungar, L. (2004). Cluster-based concept invention for statistical relational learning. *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 665–670). Seattle, Washington.

- Popescul, A., Ungar, L., Lawrence, S., & Pennock, D. (2003). Statistical relational learning for document mining. *Proceeding of the 3rd IEEE International Conference on Data Mining* (pp. 275–282). Melbourne, Florida.
- Prati, R., & Flach, P. (2005). ROCCER: An algorithm for rule learning based on ROC analysis. *Proceeding of the 19th International Joint Conference on Artificial Intelligence* (pp. 823–828). Edinburgh, Scotland.
- Provost, F., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. *Proceeding of the 15th International Conference on Machine Learning* (pp. 445–453). Madison, Wisconsin.
- Quinlan, J. R. (1990). Learning logical definitions from relations. *Machine Learning*, 5, 239–266.
- Quinlan, J. R. (1996). Boosting first-order learning. *Proceedings of the 7th International Workshop on Algorithmic Learning Theory*, 1160, 143–155.
- Raghavan, V., Bollmann, P., & Jung, G. S. (1989). A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions Information System*, 7, 205–229.
- Ray, S., & Page, D. (2001). Multiple instance regression. *Proceedings of the 18th International Conference on Machine Learning* (pp. 425–432). Williamstown, Massachusetts.
- Rendell, L. (1985). Substantial constructive induction using layered information compression: Tractable feature formation in search. *Proceeding of the 9th International Joint Conference on Artificial Intelligence* (pp. 650–658). Los Angeles, California.
- Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62, 107–136.
- Santos Costa, V., Page, D., Qazi, M., & Cussens, J. (2003). CLP(BN): Constraint logic programming for probabilistic knowledge. *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence* (pp. 517–524). Acapulco, Mexico.

- Sato, T. (1995). A statistical learning method for logic programs with distributional semantics. *Proceedings of the Twelfth International Conference on Logic Programming* (pp. 715–729). Tokyo, Japan.
- Schrag, R. C. (2004). *EAGLE Y2.5 performance evaluation laboratory (PE Lab) documentation version 1.5* (Technical Report). Information Extraction & Transport Inc.
- Sickles, E., Wolverton, D., & Dee, K. (2002). Performance parameters for screening and diagnostic mammography: Specialist and general radiologists. *Radiology*, 224, 861–869.
- Singla, P., & Domingos, P. (2005). Discriminative training of Markov Logic Networks. *Proceedings of the 20th National Conference on Artificial Intelligence* (pp. 868–873). Pittsburgh, Pennsylvania.
- Srinivasan, A. (2001). *The Aleph Manual*.
- Srinivasan, A., & King, R. (1997). Feature construction with inductive logic programming: A study of quantitative predictions of biological activity aided by structural attributes. *Proceeding of the 7th International Workshop on Inductive Logic Programming* (pp. 89–104). Prague, Czech Republic.
- Srinivasan, A., Muggleton, S., & Bain, M. (1992). Distinguishing exceptions from noise in non-monotonic learning. *Proceedings of the 2nd Workshop on Inductive Logic Programming* (pp. 97–107). Tokyo, Japan.
- Srinivasan, A., Page, D., Camacho, R., & King, R. (2006). Quantitative pharmacophore models with Inductive Logic Programming. *Machine Learning Journal*, 64, 65–90.
- Taskar, B., Abbeel, P., & Koller, D. (2002). Discriminative probabilistic models for relational data. *Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence* (pp. 485–492). Edmonton, Alberta.
- Vapnik, V. (1999). *The Nature of Statistical Learning Theory*, chapter 5. Statistics for Engineering and Information Science. Springer.

- Vens, C., Assche, A. V., Blockeel, H., & Džeroski, S. (2004). First order random forests with complex aggregates. *Proceedings of the 14th International Conference on Inductive Logic Programming* (pp. 323–340). Porto, Portugal.
- Xu, Z., Tresp, V., Yu, K., & Kriegel, H.-P. (2006). Infinite hidden relational models. *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*. Boston, Massachusetts.
- Yan, L., Dodier, R., Mozer, M., & Wolniewicz, R. (2003). Optimizing classifier performance via the Wilcoxon-Mann-Whitney statistics. *Proceedings of the 20th International Conference on Machine Learning* (pp. 848–855). Washington, D.C.
- Zelenko, D., Aone, C., & Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3, 1083–1106.
- Zelle, J., Mooney, R., & Konvisser, J. (1994). Combining top-down and bottom-up techniques in inductive logic programming. *Proceedings of the 11th International Conference on Machine Learning* (pp. 343–351). Tahoe City, California.