Effectiveness of Information Extraction, Multi-Relational, and Multi-View Learning for Predicting Gene Deletion Experiments

Mark-A. Krogel
University of Magdeburg, FIN/IWS
Universitätsplatz 2, 39106 Magdeburg, Germany
krogel@iws.cs.uni-magdeburg.de

Tobias Scheffer
Humboldt University, Computer Science
Unter den Linden 6, 10099 Berlin, Germany
scheffer@informatik.hu-berlin.de

ABSTRACT

We focus on the problem of predicting gene deletion experiments. In order to build a model that describes the underlying biological system well, our goal is to effectively utilize all data sources that are available, including unlabeled data, relational data, and abstracts of research papers. We study the effectiveness of transduction and co-training for exploiting unlabeled data. We investigate a propositionalization approach which uses gene interaction data. We study the benefit of text classification and information extraction for utilizing scientific abstracts. The studied task is one of the two data mining problems of the KDD Cup 2002; the solution that we describe achieved the highest score in one of the two subtasks and received an "Honorable Mention" for the overall task. Our results shed light on the benefits and limitations of several machine learning techniques for this large-scale application.

1. INTRODUCTION

DNA microarray technology allows to measure the gene expression of several thousand genes in parallel. These expression vectors are generally viewed as the functional state of a cell; the state-trajectories contain information about the regulatory system underlying the cellular processes. A central goal of computational biology is to understand – *i.e.,* to build models of – these underlying systems, using data obtained in microarray experiments.

We focus on learning to predict gene deletion experiments [19] which relate to regulation of the aryl hydrocarbon receptor (AhR) signaling pathway. In a gene deletion experiment, a single gene is knocked out; the problem is to predict whether this will have an effect on a particular target system. This problem constitutes one of the KDD Cup 2002 tasks [4].

The available data contains attributes that describe properties of a protein as well as relational data that describes interactions among proteins. Both, labeled and unlabeled data are available. Furthermore, there is a large body of relevant scientific publications available in the MEDLINE repository. Focusing on the goal of building as accurate a model of the biological system as possible, we explore the effectiveness of several approaches that allow us to utilize these available sources of unlabeled, multi-relational, and textual data.

Traditional, propositional machine learning algorithms require the instances to consist of a fixed set of attributes. This requirement, however, is not met by intrinsically relational data such as gene interactions. Since relational learning (*e.g.,* [6]) involves several computationally hard problems, approaches have been studied which *propositionalize* relational data – *i.e.,* cast a controlled amount of relational information into attributes (*e.g.,* [11; 13]). This approach allows to use efficient and accurate learning algorithms such as the Support Vector Machine.

For the focused problem, unlabeled data is inexpensive and readily available. Here, an unlabeled instance is a gene, the deletion of which has an unknown effect. Approaches to learning from both labeled and unlabeled data that have been studied include active learning algorithms (*e.g.,* [3]), the EM algorithm [16], transduction [8] and the multi-view framework [1].

Abstracts of scientific papers that are available in the MEDLINE collection, contain large amounts of relevant information that can be helpful for model building. Many researchers have studied algorithms that extract information from literature (*e.g.,* [14; 7]). Most popular are dictionary-based extractors (*e.g.,* [7]), but other approaches such as rule learning [5] and hidden Markov models [14] have also been explored.

The rest of this paper is organized as follows. Section 2 discusses the task and data in more detail and describes the experimental setting. In Section 3, we describe our propositionalization approach. Section 4 focuses on our studies on using text mining techniques to exploit information from the scientific abstracts, while Section 5 presents results on using unlabeled data. A discussion of our competition results and lessons learned is provided in Section 6.

2. PROBLEM DESCRIPTION AND EXPERIMENTAL SETTING

The experimental data for the KDD Cup 2002 [4] deals with the characterization and regulation of the aryl hydrocarbon receptor (AhR) signaling pathway. The AhR is a basic helix-loop-helix transcription factor with the ability to bind both synthetic chemicals such as dioxins and naturally-occurring phytochemicals, sterols and heme breakdown products. This receptor plays an important developmental and physiologi-

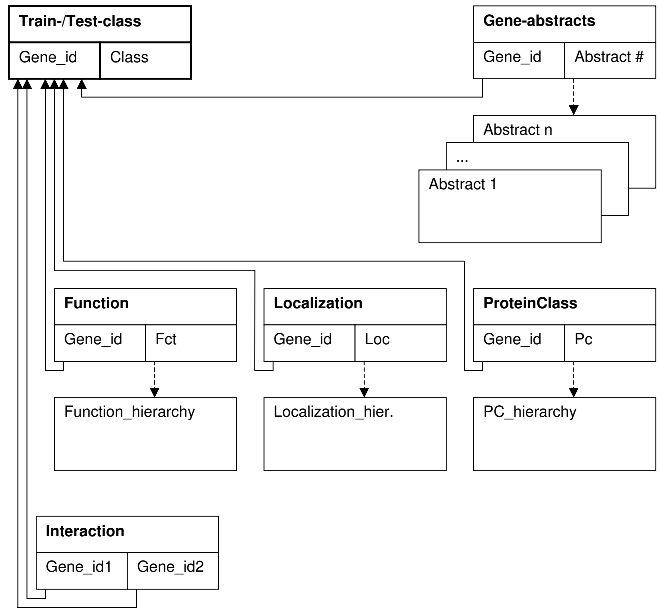

Figure 1: The data provided for KDD Cup 2002: tables (structured rectangles with table names above and column names below) and text files (simple rectangles with file names), with solid arrows representing foreign key relationships between tables and dashed arrows representing quasi-foreign key relationships between tables and file names (abstracts) or text contents (others)

cal role. In AhR-null mice, altered liver pathology and accelerated rates of apoptosis are observed.

Our aim is to generate a model that is able to predict whether deleting a gene will have an effect on AhR signaling in the cell. In order to generate the model, experimental data is available that has been obtained in recent experiments with a set of *S. cerevisiae* (yeast) strains using a gene deletion array. Each instance in the data set represents a trial in which a single of several thousand genes is knocked out and the activity of target system (AhR signaling) is measured. The model is to discriminate between genes that, when being knocked out, affect the target system (class "change"), affect the entire cell ("control"), and those which do not have an effect on the target system ("no change").

The available data describes 6,397 genes. 3,018 of these are labeled training instances. Another 1,489 of the whole set of genes serve as test examples (the class labels have been withheld until the end of the competition). The class distribution for this classification problem is highly skewed: for the majority class *no change*, 2,934 instances are available; there are 38 instances for class *change* and 45 instances for *control*.

Figure 1 illustrates the available attributes and relations. The functions of proteins encoded by genes are described by a hierarchical attribute with five levels of function names. The localization of proteins is a hierarchical attribute with two levels of *loci*. Protein classes are encoded by four hierarchical levels of protein class names. A list of pairs of gene identifiers describes gene interactions. A table relates genes to 15,235 relevant abstracts in the MEDLINE repository; for roughly half the genes, at least one relevant abstract is available. We find many missing values in the tables: for all 6,397 genes described by the database, we have information about functions for only 3,831, about localizations for 2,357, about protein classes for 999, interactions for 1,447, and abstracts for 3,329 cases.

The hypotheses are assessed by their area under the ROC curve. The *Receiver Operating Characteristic* (ROC) curve [2; 18] details the performance of a decision function in terms of the rates of true positives and false positives that are obtained by comparing the decision function for the positive class against decreasingly large threshold values. The area under the ROC curve is equal to the probability that, when we draw one positive and one negative example at random, the decision function assigns a higher value to the positive than to the negative example. Hence, the area under the ROC curve (the *AUC performance*) is a very natural measure of the ability of a decision function to separate positives from negatives. The task here is to maximize two AUC values: for the classification task *change* vs. *control* and *no change* (the "narrow positive class" problem) and for *change* and *control* vs. *no change* (the "broad positive class" problem).

After some initial cross validation experiments with SVM^{light} [8] and J48 from the WEKA library [20], we selected the Support Vector Machine SVM^{light} with linear kernel and parameter settings $c = 2$, $j = \frac{|negative\ examples|}{|positive\ examples|}$ as core machine learning algorithm. SVM^{light} requires the training data to consist of (potentially high dimensional) numerical attribute vectors.

In the following experiments, we study the influence of approaches of creating new attributes from the available data on the performance of the resulting classifiers. By comparing several attribute configurations using cross validation on the 3,017 training data, we obtained an apparently optimal configuration X^* that maximizes the sum of AUC performances for both classification problems on the training data. In order to study the benefit of some attribute x generated by one of the discussed approaches, we compare the performance of the configuration $X^* \cup \{x\}$ to configuration $X^* \setminus \{x\}$ on the withheld test data – *i.e.,* we compare the configuration with highest cross validation performance with and without the focused attribute. Note that we used cross validation on the training data to select an attribute configuration; but we use the test data to evaluate the benefit of various attributes. In order to estimate the standard deviation of the AUC performance, we used the Wilcoxon statistics [2] based on the test set performance.

3. PROPOSITIONALIZATION

The gene interaction data contains pairs of names of interacting genes. In order to integrate this information into our solution, we have to generate attributes from these relations. This situation is rather typical as relational databases usually contain more than one table. We use the RELAGGS algorithm [13] that implements ideas of extending the usual framework of propositionalization [11] with the application of SQL aggregation functions.

The RELAGGS algorithm takes as input a set of database tables, with one attribute of one table being marked as target. The target table is to describe one instance per line. In addition, the algorithm exploits foreign key relationships to compute (user selected) joins that always include the target table. Note that, while in the target table each instance was represented by a single line, the result of a join will generally contain multiple lines per instance; the lines representing an instance may differ in several attributes. The algorithm now summarizes these lines in one single line per instance, collapsing the set of values of non-unique attributes into one single value by means of aggregation functions.

In order to understand this process, consider the following example. In the application at hand, we have a table *train-class* with attributes *gene-id* and the target attribute *class*. Furthermore, we have a table *interaction* with *gene-id1* and *gene-id2* and, slightly simplified, a table *localization* with attributes *gene-id*, *mitochondria*, *cytoplasm*, and one attribute for all other possible localizations. Assume that gene “1” interacts with genes “2” and “3”, where “2” has value 1 for attribute *mitochondria* and 0 for *cytoplasm*, and “3” has value 1 both for *mitochondria* and *cytoplasm*. After joining the three tables, we obtain two lines starting with *gene-id* “1”; the first has value 1 for *mitochondria* and 0 for *cytoplasm*, the second has value 1 for both these attributes.

We can now collapse these two lines into one by applying aggregation functions such as *min, max, avg*, or *sum*; in this case, *sum* is appropriate. This leads to one line with values 2 and 1 for attributes *mitochondria* and *cytoplasm*, respectively, indicating that gene “1” interacts with two genes localizing in the mitochondria and one gene localizing in the cytoplasm.

We handle set-valued attributes by introducing one attribute per value (such as for the example of *localization* which could, for instance, be both *mitochondria* and *cytoplasm* for a single gene. Furthermore, we enrich table *interaction* by making symmetry explicit; *i.e.,* we introduce an entry (B, A) for every (A, B) in the original table. We experiment with “n-th level” interactions; an n-th level interaction exists between genes A and B if we have to traverse n interaction relations to reach B from A. For the problem at hand, the RELAGGS output consists of a single table with more than 1,000 columns. This rather high number of columns is caused first of all by the number of different values for functions, localizations, and protein classes.

We compare the performance of the classifier without and with attributes that reflect first, second, and third level interactions. Table 1 shows the resulting AUC values; Figure 2 plots the corresponding ROC curves. We see that first level interactions perform best for the narrow, second level interactions are best for the broad positive class. A significant improvement ($p \approx 0.05$) is achieved for the narrow class, we see no significant improvement for the broad class.

4. TEXT MINING

It seems likely that the archive of more than 15,000 scientific abstracts contains information that is relevant to predicting the behaviour of the focused proteins. We study two approaches to exploiting the information in the abstracts. We use a text classifier to learn the relation between abstracts relating to a gene and the effect of that gene being deleted, and we use an information extractor to identify additional gene properties in the abstracts.

4.1 Information Extraction

The attributes of the original data set contain very many missing values. We therefore want to study whether an information extraction algorithm can effectively be used to search for missing information in the abstracts. We follow a dictionary-based approach [7]. From the hierarchical text files that contain possible values for the attributes function, localization, and protein class, we manually define a thesaurus that lists, for each of the possible values of these attributes, a number of plausible terms that can be used to refer to this value. These terms have to be so specific that they are not used with different meanings in the abstracts than those searched for. At the same time, they have to be general enough to have a chance to occur in the abstracts.

The terms were constructed according to a few principles that proved to be useful according to some preliminary investigations into the search results produced.

1. Simple words and word groups as names for functions, localizations, or protein classes are entered as such into the thesaurus, *e.g.,* “nucleus”.

2. For central singular terms in the thesaurus, we also introduce the plural form, and vice versa; *e.g.,* “nuclei” for “nucleus”.

3. Multiply occurring words in the hierarchy files are equipped with some descriptive word derived from the super-ordinate name before adding it to the thesaurus; *e.g.,* “alpha adaptin” instead of “alpha”.

4. Long word groups are split at connectives such as “and”, “or”, and only the (possibly enhanced) splitting results were entered into the thesaurus; *e.g.,* “nitrogen and sulfur utilization” becomes “nitrogen utilization” and “sulfur utilization”.

5. Short word groups are also given in paraphrased variants to the thesaurus; *e.g.,* in addition to “DNA replication” we introduce “replication of DNA”.

Table 1: Additional information from gene interactions

	without	first level	second level	third level
narrow	0.617 ± 0.0592	0.707 ± 0.050	0.685 ± 0.0527	0.6546 ± 0.055
broad	0.599 ± 0.040	0.598 ± 0.049	0.630 ± 0.039	0.597 ± 0.040

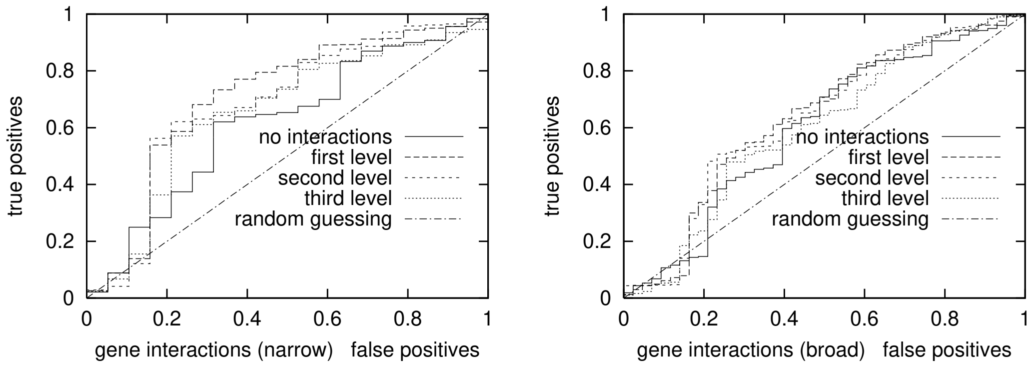

Figure 2: ROC curves with and without information from gene interactions.

Table 2: Additional information from information extraction

	without	with	IE only
narrow	0.590 ± 0.061	0.685 ± 0.052	0.654 ± 0.055
broad	0.597 ± 0.040	0.630 ± 0.039	0.510 ± 0.044

6. Explanations in brackets are extracted and dealt with according to the previous points; *e.g.,* for "amino acid degradation (catabolism)".

We concentrate on abstracts that are related to just one gene according to table *gene-abstracts* to avoid ambiguity problems. Here, we do not have to resolve to which of several proteins that are mentioned in a paper each property refers. Table 2 shows that the information extractor yields a substantial performance improvement (the base line "without" is an attribute set without the extracted information). Surprisingly, the problem can even be solved to some degree using *only* the information extracted from the abstracts ("IE only"). Figure 3 shows the corresponding ROC curves.

4.2 Text Classification

For each protein that is mentioned in at least one abstract, we first build a bag of abstracts that refer to that protein (abstracts may occur in more than one bag). Hence, each instance x is now a bag of abstracts, the corresponding class label is the protein's class label. Only roughly half of the proteins are mentioned in at least one abstract. Therefore, the training set is smaller and the resulting text classifier can only be applied to proteins that are mentioned in at least one paper.

In order to train text classifiers from the generated training sets, we first tokenize the bags, apply Porter's stemming algorithm [17], and infer the TFIDF vector from each bag. Using the TFIDF vectors as training set, we train an SVM classifier using SVM^{light} with default parameters.

Table 3: Additional information from text classification

	without	with
narrow	0.685 ± 0.052	0.657 ± 0.055
broad	0.630 ± 0.039	0.618 ± 0.039

The trained classifiers yield a prediction for the hold-out instances that we would like to exploit. The value of the decision function now serves as an additional attribute to the top-level SVM that processes the categorical attributes and propositionalized relational data.

Table 3 compares the performance of the SVM decision function with and without the additional attribute generated by the text classifier. For both classification problems, we observe a *decrease* in accuracy. The differences are not significant, but we do not achieve an improvement by using the text classification attribute.

5. UTILIZING UNLABELED DATA

Several approaches can exploit unlabeled examples available in addition to labeled positive and negative samples. For the Support Vector Machine (SVM), the transduction approach [9] applies. Independent of the learning algorithm used, the multi-view framework [1] can be applied in all cases in which the attributes can be split into two independent and sufficient subsets.

5.1 Transduction

The transductive SVM [9] uses unlabeled examples to refine the weight vector that maximizes the margin between separating hyperplane and labeled and unlabeled examples. The optimization problem which the SVM learning procedure solves is to find w and b such that $y_i(wx_i + b)$ is positive for all examples (all instances lie on the "correct" side of the plain) and the smallest margin (over all examples) is maximized. Equivalently to maximizing $y_i(\frac{w}{|w|}x_i + b)$, it is usually demanded that $y_i(wx_i + b) \geq 1$ for all (x_i, y_i)

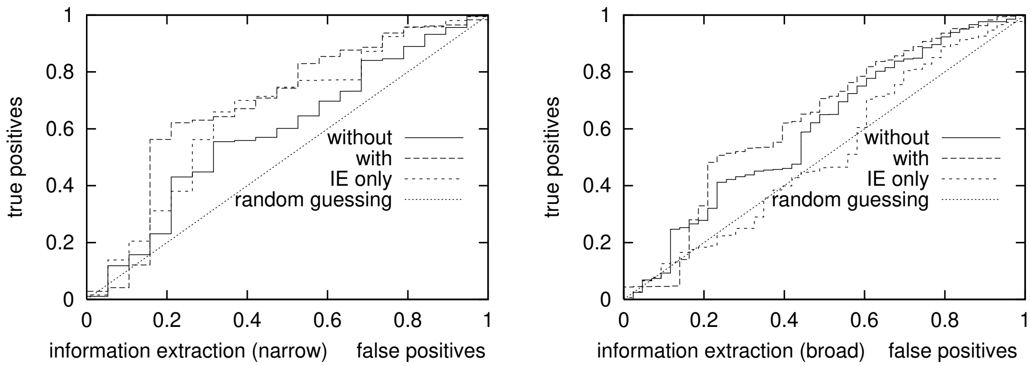

Figure 3: ROC curves with and without information from information extraction.

and $|w|$ be minimized. The SVM^{light} software package [8] implements an efficient optimization algorithm which solves optimization problem 1.

Optimization Problem 1. *Given data* $((x_1, y_1), \ldots, (x_m, y_m))$*; over all* w*,* b*, minimize* $|w|^2$*, subject to constraint* $\forall_{i=1}^{m} y_i(wx_i + b) \geq 1$.

The transductive Support Vector Machine [9] (TSVM) furthermore considers unlabeled data. This unlabeled data can (but need not) be new instances which the SVM is to classify. In transductive support vector learning, the optimization problem is reformulated such that the margin between all (labeled and unlabeled) examples and hyper-plain is maximized. However, only for the labeled examples we know on which side of the hyper-plain the instances have to lie.

Optimization Problem 2. *Given labeled data* $((x_1, y_1), \ldots, (x_m, y_m))$ *and unlabeled data* $(x_1^*, \ldots, x_k^*)$*; over all* w*,* b*,* $(y_1^*, \ldots, y_k^*)$*, minimize* $|w|^2$*, subject to the constraints* $\forall_{i=1}^{m} y_i(wx_i + b) \geq 1$ *and* $\forall_{i=1}^{m} y_i^*(wx_i^* + b) \geq 1$.

The TSVM algorithm starts by learning parameters from the labeled data and labels the unlabeled data using these parameters. It iterates a training step (corresponding to the "M" step of EM) and switches the labels of the unlabeled data such that optimization criterion 2 is maximized (resembling the "E" step).

The transductive SVM decreased the AUC for the broad class from .063 (± 0.039) to 0.60 (± 0.04) and increased AUC for the narrow class from 0.685 (± 0.052) to 0.695 (± 0.05). Both differences are well below the standard deviations and are therefore insignificant. Our data does not provide any convincing evidence that transduction has an influence on the quality of the resulting decision function at all. This result is disappointing; in particular, as the transductive SVM dramatically increases computation time.

5.2 Multi-View Learning

Blum and Mitchell [1] have proposed the multi-view approach. The available attributes V are split into two disjoint subsets V_1 and V_2. A labeled example (x, a) is then viewed as (x_1, x_2, a) where x_1 contains the values of the attributes in V_1 and x_2 the values of attributes in V_2.

The co-training algorithm is the most prominent multi-view algorithm. The idea of co-training is to learn two classifiers

Table 4: Co-training algorithm

Given positive examples $(x_1, x_2, +)$, negative examples $(x_1, x_2, -)$ and unlabeled examples in two different views V_1 and V_2; number of iterations k.

1. Loop for k iterations
   (a) Train f_1 and f_2 using the labeled positive and negative examples.
   (b) Let f_1 and f_2 select the positive and negative example for which they make the most confident prediction. Remove the examples from the unlabeled data and add them to the labeled data.
2. Return the combined classifier $f(x) = f_1(x_1) + f_2(x_2)$.

$f_1(x_1)$ and $f_2(x_2)$ which bootstrap each other by providing each other with labels for the unlabeled data. Co-training is applicable when either attribute set suffices to learn the target f – *i.e.*, there are classifiers f_1 and f_2 such that for all x: $f_1(x_1) = f_2(x_2) = f(x)$ (the *compatibility* assumption). When the views are furthermore *independent* given the class labels – $P(x_1|f(x), x_2) = P(x_1|f(x))$ – then the co-training algorithm labels unlabeled examples in a way that is essentially equivalent to drawing labeled data at random [1]. However, empirical studies [15; 10] show that co-training can improve classifier performance even when the assumptions are violated to some extent.

We let V_1 be the items of the database and V_2 the attribute extracted from the abstracts together with the relational attributes (referred to as the "natural" attribute split in the following). $f_1(x_1)$ and $f_2(x_2)$ are trained from the same positive and negative examples. Now f_1 selects two examples from the unlabeled data that it most confidently rates positive and negative, respectively, and adds them to the labeled examples for f_2. If the representations in the two views are truly independent, then the new examples are randomly drawn positive and negative examples for f_2. Now f_2 selects two unlabeled examples for f_1, the two hypotheses are retrained, and the process recurs. The algorithm is presented in Table 4.

Our goal in this set of experiments is to validate whether co-training can effectively exploit the information contained in the unlabeled data and thereby increase the quality of the

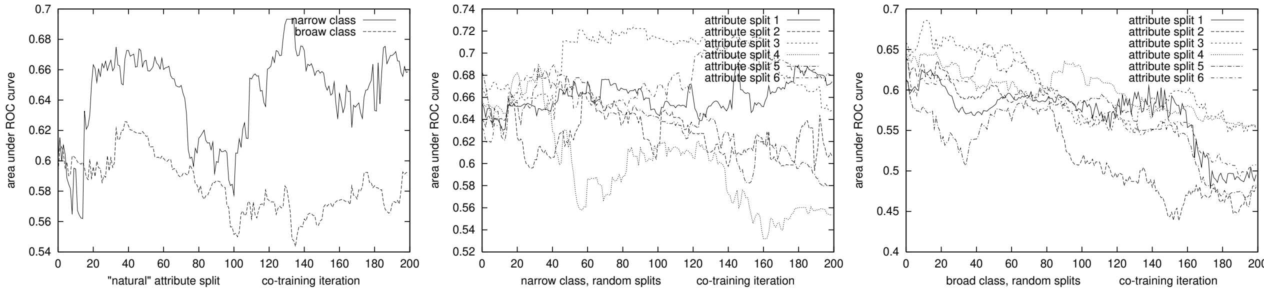

Figure 4: Co-training results

resulting decision function. As control strategy to the "natural attribute split", we randomly partition the attributes into two subsets. After each iteration in which each classifier labels two unlabeled instances, we measure the performance of the combined classifier using the test set.

Figure 4 (left) shows how the AUC develops over 200 iterations of co-training using the "natural" attribute split. Unfortunately, the performance does not improve over the co-training iterations; the standard deviations are around 0.05, the differences between initial and final AUC are insignificant. Furthermore, the combined decision function (which is the average of two decision functions based on the two distinct attribute sets) is significantly worse than one single decision function which can base its decision on all attributes (this baseline classifier achieves 0.63 ± 0.04 for the broad and 0.685 ± 0.05 for the narrow class)! In case of randomly partitioned attribute sets (Figure 4, middle, for narrow and 4, right, for broad), the average AUC decreases significantly over the co-training iterations ($p < 0.05$) for the broad and seems to decrease (but not to a sufficient significance level) for the narrow class.

Our experiments with co-training show interesting results. In most (published) previous studies co-training has led to an increase in accuracy for real-world data, even though the underlying independence assumption has largely been violated. While these findings raise the question whether co-training can perhaps always increase performance to some extent, our results clearly answer this question negatively. Not only can the combination of the two initial decision functions perform poorer than one single decision function that accesses all attributes, but also the performance can decrease further during co-training.

6. DISCUSSION AND LESSONS LEARNED

For the competition, we [12] used attributes generated by RELAGGS with two interaction levels, and entries acquired by information extraction. We did not include the text classification attribute and did not use transduction or co-training. The competition schedule was a limiting factor for us. We only generated the classifier for the narrow class problem (here we obtained the highest score), and used this classifier for both, the narrow and broad positive class. Using the narrow classifier for the broad class, we still obtained a third rank for the overall task. Figure 5 depicts a comparison of the solutions handed in by the different teams [4].

Retrospectively, we can now obtain AUC performances of 0.707 for the narrow and 0.63 for the broad class using two different decision functions and the optimal attribute configuration – which is more than any team could achieve within the competition time frame. From our experimental results, we draw the following lessons learned.

1. In microarray data, the interactions between genes play a crucial role. Propositionalizing the relational data by computing joins of the tables, and collapsing the joins by aggregation functions proved to be both an effective and efficient means of utilizing this information.

2. Semi-supervised learning techniques such as the transductive SVM and co-training are less generally applicable than – at least we – expected. Our expectation was that taking unlabeled data into account should at least not decrease, and perhaps even modestly improve, performance. For mining microarray data, this assumption is not true.

3. MEDLINE abstracts contain important knowledge that can help to build better models and thus to perform better on classification tasks. Our data not only supports this hypothesis, but also shows that even fairly simple, dictionary-based extractors can generate attributes from abstracts that substantially improve classification performance. We are confident that more sophisticated extractors will further improve biological models.

7. ACKNOWLEDGEMENTS

Tobias Scheffer is supported by grant SCHE540/10-1 of the German Science Foundation DFG.

8. REFERENCES

[1] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, 1998.

[2] A. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[3] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. In *Advances in Neural Information Processing Systems*, volume 7, pages 705–712, 1995.

[4] M. Craven. The 2002 KDD Cup competition results for gene regulation prediction. *SIGKDD Explorations*, 4(2), 2003.

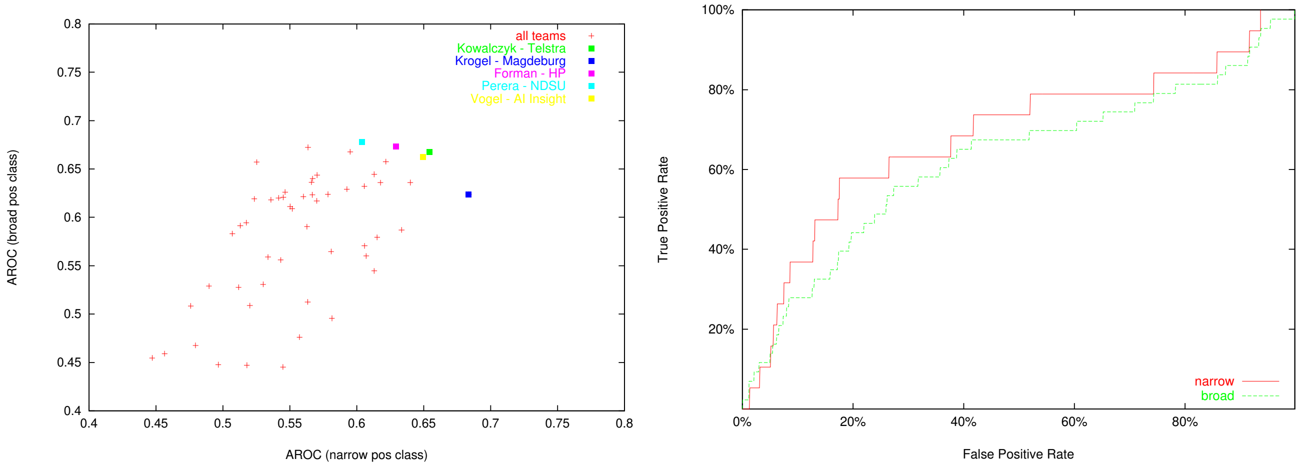

Figure 5: Results of KDD Cup 2002 participants on task 2 (graphics provided by Mark Craven)

[5] Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118(1-2):69–113, 2000.

[6] S. Dzeroski and N. Lavrac, editors. *Relational Data Mining*. Springer, 2001.

[7] K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. Towards information extraction: Identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing*, 1998.

[8] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, 1999.

[9] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the International Conference on Machine Learning*, 1999.

[10] S. Kiritchenko and S. Matwin. Email classification with co-training. Technical report, University of Ottawa, 2002.

[11] S. Kramer, N. Lavrač, and P. A. Flach. Propositionalization Approaches to Relational Data Mining. In N. Lavrač and S. Džeroski, editors, *Relational Data Mining*. Springer, 2001.

[12] M.-A. Krogel, M. Landwehr, M. Denecke, and T. Scheffer. Using data and text mining techniques for yeast gene regulation prediction: A case study. *SIGKDD Explorations*, 4(2), 2003.

[13] M.-A. Krogel and S. Wrobel. Transformation-Based Learning Using Multirelational Aggregation. In *Proceedings of the Eleventh International Conference on Inductive Logic Programming (ILP)*. Springer, 2001.

[14] T. Leek. Information extraction using hidden Markov models. Master's thesis, University of California at San Diago, 1997.

[15] I. Muslea, C. Kloblock, and S. Minton. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the International Conference on Machine Learning*, 2002.

[16] Kamal Nigam, Andrew K. McCallum, Sebastian Thrun, and Tom M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3), 2000.

[17] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[18] F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation in comparing classifiers. In *Proceedings of the International Conference on Machine Learning*, 1998.

[19] E. Winzeler, D. Schoemaker, A. Astromoff, and H. Liang *et al.* Functional characterization of saccharomyces cerevisiae genome by gene deletion and parallel analysis. *Science*, 285, 1999.

[20] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.