Machine Learning, 57, 61–81, 2004

Multi-Relational Learning, Text Mining, and Semi-Supervised Learning for Functional Genomics

MARK-A. KROGEL krogel@iws.cs.uni-magdeburg.de
Otto-von-Guericke-Universität Magdeburg, FIN/IWS, Universitätsplatz 2, 39106 Magdeburg, Germany

TOBIAS SCHEFFER scheffer@informatik.hu-berlin.de
Humboldt-Universität zu Berlin, Department of Computer Science, Unter den Linden 6, 10099 Berlin, Germany

Editors: Nada Lavrač, Hiroshi Motoda, Tom Fawcett

Abstract. We focus on the problem of predicting functional properties of the proteins corresponding to genes in the yeast genome. Our goal is to study the effectiveness of approaches that utilize all data sources that are available in this problem setting, including relational data, abstracts of research papers, and unlabeled data. We investigate a propositionalization approach which uses relational gene interaction data. We study the benefit of text classification and information extraction for utilizing a collection of scientific abstracts. We study transduction and co-training for using unlabeled data. We report on both, positive and negative results on the investigated approaches. The studied tasks are KDD Cup tasks of 2001 and 2002. The solutions which we describe achieved the highest score for task 2 in 2001, the fourth rank for task 3 in 2001, the highest score for one of the two subtasks and the third place for the overall task 2 in 2002.

Keywords: propositionalization, information extraction, co-training

1. Introduction

One of the principal challenges of bioinformatics is to generate models which describe the relation between genetic information and the corresponding cellular processes. Such models have to explain—and can be derived from—available experimental data. Available data in functional genomics include relational information (gene interactions), textual repositories like MEDLINE, and unlabeled data (e.g., genes with unknown functional properties). The goal of our work is to study the effectiveness of methods for learning from these types of data in the application field of functional genomics.

Our empirical studies focus on a set of problems in our application field. We aim at predicting the high-level function and the localization of the protein corresponding to a given yeast gene. We furthermore aim at predicting whether a given gene is involved in the regulation of the aryl hydrocarbon receptor (AhR) signaling pathway. The latter problem serves as a prototypical problem of building a functional model from data generated by gene deletion experiments. The data which we use have been provided for the KDD Cups 2001 (Cheng et al., 2002) and 2002 (Craven, 2002).

Propositional machine learning algorithms require the instances to consist of a fixed set of attribute values. This requirement, however, is not met by intrinsically relational data such as

gene interactions. Inductive logic programming algorithms address this problem by learning logic programs (e.g., Džeroski & Lavrač, 2001). Approaches have been developed out of inductive logic programming which *propositionalize* relational data—i.e., cast a controlled amount of relational information into attributes (e.g., Kramer, Lavrač, & Flach, 2001; Krogel & Wrobel, 2001). The latter approach allows to use efficient and accurate learning algorithms such as the Support Vector Machine (SVM) after the propositionalization step.

Abstracts of scientific papers that are available in the MEDLINE collection contain information that can be helpful for model building. Many researchers study algorithms that extract information from literature (Hirschmann et al., 2002; Leek, 1997; Fukuda et al., 1998; Craven et al., 2000; Hahn, Romacker, & Schulz, 2002). We will see that even simple dictionary-based extractors (Fukuda et al., 1998) can effectively support model building in our application area.

For the focused application area, unlabeled data is usually inexpensive and readily available. Here, an unlabeled instance is a gene with unknown function or localization, or a gene whose deletion has an unknown effect, respectively. Approaches to semi-supervised learning include transduction (Joachims, 1999b) and co-training (Blum & Mitchell, 1998). Recent results seem to indicate that semi-supervised learning is typically beneficial when the labeled sample is small (e.g., Nigam, Lafferty, & McCallum, 1999; Bruce, 2001; Kockelkorn, Lüneburg, & Scheffer, 2003). We will challenge this hypothesis for the functional genomics application area.

Our studies contribute several results. We obtain positive results on the effectiveness of propositionalization for learning in functional genomics, and on using dictionary-based information extraction to generate attributes which improve the model building step. We refute the general belief of some researchers, that transduction and co-training improve learning in all practically relevant cases (at least when few labeled data are available), and investigate on the reason for the failure of co-training. Finally, we describe a system which effectively utilizes the available data sources to solve a range of functional genomics problems. The effectiveness of our solution is assessed by its scores in several tasks of KDD Cup competitions.

The rest of this paper is organized as follows. In Section 2, we discuss the application and data set in more detail, and describe the experimental setting. In Section 3, we discuss the utilization of the relational gene-interaction data. Section 4 focuses on our studies on using text mining techniques to utilize information in MEDLINE abstracts, while Section 5 presents results on using unlabeled data. A discussion of our competition results is sketched in Section 6; a discussion of related work and lessons learned is provided in Section 7.

2. Problem description and experimental setting

Our task is to predict properties of the proteins corresponding to a given yeast gene. These properties are (1) one (or several) of 15 categories of protein functions, (2) the localization (one of 15 different parts of the cell), and (3) the involvement in the regulation of the 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

physiological role. Problems (1) and (2) have been addressed in KDD Cup 2001 whereas problem (3) is one of the tasks of KDD Cup 2002.

The available training data for problem (1) and (2) contain 862 training and 381 test instances (Cheng et al., 2002). The available training data consist of tables "gene-relation" and "gene-interaction". Table "gene-relation" relates the training genes to attributes which refer to the chromosome on which the genes appear, to whether the gene is essential for survival, observable characteristics of the phenotype, structural category of the protein, the existence of characteristic motifs in the amino acid sequence of the protein, and whether the protein forms larger proteins with others. The relation "gene-interaction" specifies which genes interact with one another.

The data for problem (3) has been obtained in experiments with yeast strains using a gene deletion array (Craven, 2002). Each instance in the data set represents a trial in which a single gene is "knocked out" and the activity of a target system (AhR signaling) is measured. We distinguish genes whose deletion affects the target system (class "change"), affects the entire cell ("control"), or does not have an effect ("no change"). We learn two discriminators: *change* vs. *control* and *no change* ("narrow positive class" problem) and *change* and *control* vs. *no change* ("broad positive class").

The data contain 3,018 training and 1,489 test examples. 2,934 fall into the class *no change*, 38 into *change* and 45 into *control*. The attributes describe function, localization and protein class of each gene (hierarchical attributes with four to five levels). Again, a relation describes gene interactions. A table relates genes to 15,235 relevant abstracts in the MEDLINE repository. 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.

We assess the hypotheses by their area under the ROC curve. The *Receiver Operating Characteristic* (ROC) curve (Bradley, 1997; Provost, Fawcett, & Kohavi, 1998) 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 will assign a higher value to the positive example than to the negative. Hence, the area under the ROC curve (called the *AUC performance*) is a very natural measure of the ability of a decision function to separate positive from negative examples. An AUC performance of 0.5 corresponds to random guessing whereas an area of 1 is obtained by a perfect separator. We estimate the standard deviation of the AUC performance using the Wilcoxon statistics (Bradley, 1997).

When assessing the benefit of methods, we compare ROC curves of decision functions for some of the classes. For problems (1) and (2)—KDD Cup 2001—we have to restrict our comparative studies to the three most frequent class values because the remaining classes are represented by too few instances to obtain reliable performance estimates. For problem (3)—KDD Cup 2002—we study the areas under two ROC curves for the "narrow positive class" (*change* vs. *control* and *no change*) and the "broad positive class" problem (*change* and *control* vs. *no change*).

After some initial cross validation experiments on the training data with SVM^{light} (Joachims, 1999a) and J48 from the WEKA library (Witten & Frank, 1999), 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 for all problems. 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 to creating new attributes from the available data on the performance of the resulting classifiers. In each experiment, we add or remove one (or a set of) attributes to or from the attribute configuration which we used for our competition submissions and compare the resulting classifier to this competition classifier. We used cross validation on the training data to select the attribute configuration for our competition submissions; but since the test data is available now, we use all available data for our retrospective studies.

3. Propositionalization

The gene interaction data contain pairs of names of interacting genes. In order to integrate information about interaction pairs 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 with foreign key relationships between them. We use the RELAGGS algorithm (Krogel & Wrobel, 2001) that implements ideas of extending the usual framework of propositionalization (Kramer, Lavrač, & Flach, 2001) 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. The algorithm exploits foreign key relationships to compute (user selected) outer 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. For problem (3), 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 *Function* with attributes *Gene_id* and *Fct*. Figure 1, first line, shows an example database with three genes, *G1*, *G2*, and *G3*.

Our goal is to create a table that summarizes, for each example gene, information about its functions as well as about the functions of all genes that interact with that example. We compute two joins, displayed in the second line of figure 1. We join *Train_class* and *Function*, resulting in Table *Join_1*. Now we compute *Join_2* which provides information about the functions of all genes interacting with a given example gene. *Join_2* is based on the SQL statement "`select t1.Gene_id, t2.Gene_id2, t3.Fct from Train_class as t1, Interaction as t2, Function as t3 where t1.Gene_id = t2.Gene_id1 and t2.Gene_id2 = t3.Gene_id`".

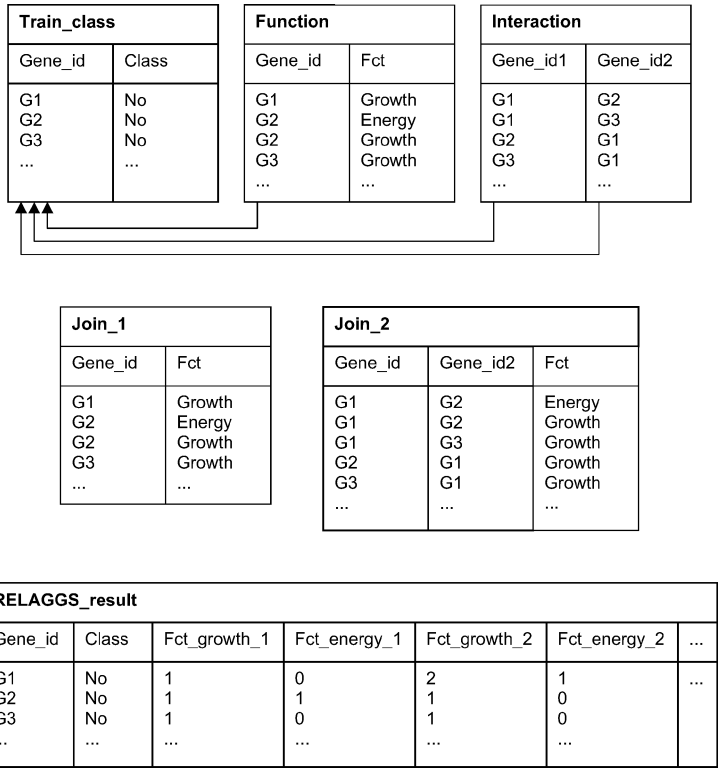

Figure 1. (First line) Excerpt of the original database with artificial entries; arrows indicate foreign key relationships. (Second line) Results of joins of tables *Train_class*, *Interaction* and *Function*. (Third line) Result after aggregation.

The result may generally contain multiple lines for a single entry of the target table. Here, this is the case for *G1*, because it interacts with gene *G2* which has multiple functions, and it also interacts with *G3* which has a single function. These several lines for one example have to be collapsed into one single line. This is achieved in the *aggregation* step by generating a column for each possible value of *Fct* and then counting the number of occurrences of those values per example from the target relation.

The final result is obtained after joining the aggregates to the target table; it can be found in the third line of figure 1. Here, attribute names with an extension of "*_1*" indicate functions of the instances themselves, whereas an extension of "*_2*" refers to functions of genes that interact with the example gene described in the current line. For instance, value "2" in column *Fct_growth_2* for *G1* means here that there are two interaction partners of *G1* of function "growth"; that is, *G2* and *G3*. Value "1" for *Fct_growth_1* in the same line indicates that *G1* has function "growth" itself, too.

We generally handle set-valued and hierarchical attributes by introducing one attribute per value on each hierarchy level. We enrich table *interaction* by making symmetry explicit; i.e., we introduce an entry (B, A) for every (A, B) in the original table.

Table 1. AUC performance for problems (1) and (2) with and without relational information.

Class	Without	With
Function growth	0.872 ± 0.01	0.882 ± 0.014
Function transcription	0.886 ± 0.005	0.899 ± 0.011
Function transport	0.893 ± 0.0099	0.918 ± 0.013
Localization cytoplasm	0.861 ± 0.0078	0.865 ± 0.014
Localization mitochondria	0.909 ± 0.013	0.948 ± 0.0098
Localization nucleus	0.941 ± 0.0054	0.944 ± 0.011

Table 2. AUC performance for problem (3) with and without 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

For problems (1) and (2), we compare the decision functions with and without the interaction attributes, generated by the RELAGGS algorithm in Table 1 (using 10-fold cross validation). The observed AUC performance obtained when using the interaction information is in every single case higher; the improvement exceeds two standard deviations in two cases and one standard deviation in two more cases. These data rule out the null hypothesis that the interaction information does not improve recognition performance. For problem (3), we compare the performance without and with attributes that reflect first, second, and third level interactions in Table 2; an n-th level interaction exists between genes A and B if we have to traverse n interaction relations to reach B from A. We see that first level interactions perform best for the narrow, and second level interactions are best for the broad positive class. A significant improvement is achieved for the narrow class, we see a smaller, insignificant improvement for the broad class.

4. Text mining

In this section, we study two approaches to utilizing relevant information contained in more than 15,000 available MEDLINE abstracts. We study two approaches: we use a text classifier to learn the relation between abstracts that are related 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. Assessing classifiers in the presence of textual background knowledge

In this subsection, we discuss a particular difficulty that occurs with the assessment of classifiers that use information extracted from a text archive. This problem can lead to

extremely optimistic performance estimates. It can clearly be observed for problems (1) and (2) (the 2001 KDD Cup problems) and also applies to a broader range of learning tasks.

When assessing the performance of a classifier empirically, it is necessary to use a database S of instances (x, y) with already known class values as test set. For regular classification problems, masking the class values is very simple; instances x drawn from S with masked class values y are identically distributed to "new" instances that are drawn from the underlying distribution $p(x)$. Hence, unbiased performance estimates can be obtained.

However, the situation is different when, in addition to S, a document collection Doc_T is available that may change over time. Doc_T (or, when the classifier is started in the future, Doc_t with $t > T$) now becomes part of the input to the classifier $f : (x, Doc_T) \mapsto y$. Drawing hold-out examples from the sample S always yields instances with class labels that *may* be mentioned in Doc_T. When the classifier is started at time t ($t > T$), then it may be applied to instances x whose class labels are not yet disclosed in Doc_t. Therefore, the hold-out examples (x, Doc_T) are not necessarily *identically distributed* to *new* instances (x, Doc_t), whose class labels *may not yet* be mentioned in Doc_t.

To illustrate this problem, consider a paper which contains the sentence "APR4 localizes in the cytoplasm" $\in Doc_T$; a classifier which predicts protein localization can use these keywords in Doc_T to predict class label $f(APR4, Doc_T) =$ "cytoplasm" for held-out instance APR4. However, when the same classifier is applied at time $t > T$ by a biological research scientist to a protein with currently *unknown* localization then—by the definition of unknown—no available paper in Doc_t can contain a similar sentence for that new protein. A keyword based classifier will inevitably perform worse for truly new instances than for held-out instances in S.

An unbiased performance estimate could be obtained by training with $(x_1, Doc_1), \ldots, (x_t, Doc_t)$ and testing with (x_{t+1}, Doc_{t+1}) (successively for growing values of t), where Doc_t contains all documents that have been available when x_t has been drawn from $p(x)$. This procedure is similar to the standard procedure for assessing time series predictors. It requires each instance in the sample to be marked with the time when its classification was requested, and each document to possess a time stamp.

For the function and localization prediction problem of the 2001 challenge—problems (1) and (2)—this problem prevents us from obtaining useful performance estimates. We can use the function and localization extractor described in Section 4.2 to look up these known properties in MEDLINE abstracts. Using these extracted attributes, it is trivial to predict function and localization of proteins. But the resulting classifier would be useless for predicting function and localization of the approximately 40% yeast genes whose functions and localizations are yet unknown. For task (3), the problem is less severe. The data have been generated for a research project on the AhR signaling pathway and we can assume that the class value (information on whether deleting the gene will have an impact on the target system) is not to be found in the MEDLINE abstracts.

4.2. Information extraction

The attributes of the original data set for problem (3) contain very many missing values; in particular, many function and localization values are missing. 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 (Fukuda et al., 1998). 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 superordinate 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".
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*. Here, we do not have to resolve to which of several proteins that are mentioned in a paper each property refers. However, we do not consider larger syntactic structures than our search terms, which imposes the risk of being misled in cases such as "not in the nucleus". For every protein, we scan the corresponding abstracts for occurrences of the terms in the thesaurus. On finding a term, the corresponding database tables are appropriately enriched.

Table 3 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").

Table 3. Problem (3), 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	0.610 ± 0.044

Table 4. Problem (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

4.3. Text classification

In this section, we study whether the solution to problem (3) can be enhanced by a text classifier which uses all abstracts that relate to a given gene in order to classify that gene.

For each protein that is mentioned in at least one abstract, we first build a bag of abstracts referring to that protein. Abstracts may occur in more than one bag if they refer to multiple proteins. Thereby, for both, the narrow and broad positive class problem of task (3), we construct a new text classification problem and a corresponding training set. Each instance x is a bag of abstracts, the corresponding class label is the protein's class label. Note, however, that 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 (Porter, 1980), and infer the TFIDF vector from each bag. Each component of a TFIDF vector corresponds to a word in the dictionary and counts how often that word occurs in the document (the *term frequency*), multiplied by the *inverse document frequency*, a logarithmically scaled measure of the rareness of that word. The TFIDF vectors are normalized. Using the TFIDF vectors as training set, we train an SVM classifier. The trained classifiers yield a prediction for the hold-out instances that we would like to utilize. 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 4 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 additionally using the text classification attribute.

5. Utilizing unlabeled data

Several approaches are known that can utilize unlabeled examples available in addition to labeled positive and negative samples. For the Support Vector Machine, the transduction approach (Joachims, 1999b) applies. Independent of the learning algorithm used, co-training (Blum & Mitchell, 1998) can be applied in all cases in which the available attributes can be split into two independent and compatible attribute subsets.

5.1. Transduction

The transductive SVM (Joachims, 1999b) 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 $y_i(wx_i + b) + \xi_i \geq 1$ for all examples (all instances lie on the "correct" side of the plane), and $|w|$ be minimized (i.e., the margin is maximized). The SVM$^{\text{light}}$ software package (Joachims, 1999a) 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* $|w|^2 + \sum_i \xi_i$, *subject to constraints* $\forall_{i=1}^m y_i(wx_i + b) + \xi_i \geq 1$ *and* $\forall_{i=1}^m \xi_i \geq 0$.

The transductive Support Vector Machine (TSVM; Joachims, 1999b) furthermore considers unlabeled data. This unlabeled data can (but need not) be a holdout set 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 hyperplane is maximized. However, only for the labeled examples we know on which side of the hyperplane 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* $|w|^2 + \sum_i \xi_i$, *subject to the constraints* $\forall_{i=1}^m y_i(wx_i + b) + \xi_i \geq 1$, $\forall_{i=1}^m y_i^*(wx_i^* + b) + \xi_i \geq 1$, *and* $\forall_{i=1}^m \xi_i \geq 0$.

The TSVM algorithm starts by learning parameters from the labeled data and labels the unlabeled data using these parameters. It iterates a training step and switches the labels of the unlabeled data such that optimization criterion 2 is maximized. The influence of the unlabeled data is increased incrementally.

We compare the "vanilla" Support Vector Machine (using the attributes which gave so far the best experimental results) to the transductive SVM (using identical attributes) which utilizes the test instances (with removed class labels) for training. Training and test instances are equal for both runs of the SVM.

For problems (1) and (2), Table 5 compares 10-fold cross validation results of the "vanilla" SVM to the TSVM, using the most frequent class value for each of the two classification problems. In both cases, transduction *significantly deteriorates* performance although it has additional information (the unlabeled hold-out instances) available. Given these negative results and our previous positive experiences with TSVM for text classification problems as well as results by Joachims (1999a), we hypothesized that transduction is only beneficial if only few labeled data are available. In order to validate this hypothesis, we averaged 10 iterations in which we drew only 5 labeled positive examples (and 12 and 20 negatives, respectively) and used all remaining instances as unlabeled data. Table 5 shows that transduction still dramatically decreases classifier performance and thus refutes our hypothesis.

Table 5. Transduction results for problems (1) and (2).

	SVM	TSVM
Function growth, 150 positives	0.839 ± 0.0055	0.817 ± 0.0081
Location cytoplasm, all data	0.833 ± 0.0095	0.695 ± 0.016
Function growth, 5 positives	0.665 ± 0.005	0.546 ± 0.0068
Location cytoplasm, 5 positives	0.616 ± 0.00651	0.554 ± 0.011

For problem (3), the transductive SVM decreased the AUC for the broad class from 0.063 ($\pm$0.039) to 0.60 ($\pm$0.04) and increased AUC for the narrow class from 0.685 ($\pm$0.052) to 0.695 ($\pm$0.05). Both differences are well below the standard deviations and are therefore insignificant. This result is disappointing; in particular, as the transductive SVM dramatically increases computation time.

5.2. *Co-training*

Blum and Mitchell (1998) have proposed the co-training algorithm which splits the available attributes V 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 idea of co-training is to learn two classifiers $f^1(x_1)$ 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—$P(x_1 \mid f(x), x_2) = P(x_1 \mid f(x))$—then the co-training algorithm labels unlabeled examples in a way that is essentially equivalent to drawing labeled data at random (Blum & Mitchell, 1998). Empirical studies (Muslea, Kloblock, & Minton, 2002a; Kiritchenko & Matwin, 2002) show that co-training can improve classifier performance even when the assumptions are violated to some extent.

Let, for instance, V_1 be the items of the database and V_2 the words occurring in the abstracts that relate to a protein. $f^1(x_1)$ 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 6.

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

For problems (1) and (2), there is no "natural" split of the attributes that seems likely to lead to independence between the subsets. We therefore split the available attributes at

Table 6. Co-training algorithm.

Input: Labeled examples D_l in views V_1 and V_2; unlabeled data D_u; number of iterations T; "step size" n_p and n_n.

1. Train f_0^1 and f_0^2 on D_l using attribute sets V_1 and V_2, respectively.
2. **For** $i = 1 \ldots T$ until $D_u = \emptyset$:
   a) **For** $v = 1 \ldots 2$: Remove n_p elements with greatest $f_{i-1}^v(x_j^*)$, from D_u and add $(x_j^*, +1)$ to D_l.
   b) **For** $v = 1 \ldots 2$: Remove n_n elements with smallest $f_{i-1}^v(x_j^*)$, from D_u, add $(x_j^*, -1)$ to D_l.
   c) Train f_i^1 and f_i^2 on D_l using attribute sets V_1 and V_2, respectively.
3. **Return** the combined classifier $f(x) = f_T^1(x_1) + f_T^2(x_2)$.

random. In each experiment, we average ten co-training curves for distinct attribute splits. Figure 4 (left hand side) shows that the performance of the decision functions decreases with the co-training iterations, depending on the labeled sample size ("157 + 340" indicates 157 positive and 340 negative labeled examples).

For problem (3), we try to minimize the dependence between the two attribute sets by choosing the information extracted from the abstracts together with relational attributes extracted from the protein interaction tables as the first set, and all other attributes as the second set of attributes (referred to as the "natural" attribute split in the following). As control strategy, we randomly partition the attributes into two subsets.

Figure 2 (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 2, right hand side, shows the average AUC over six random splits), the average AUC decreases significantly over the co-training iterations for the broad and seems to decrease (but not significantly) for the narrow class.

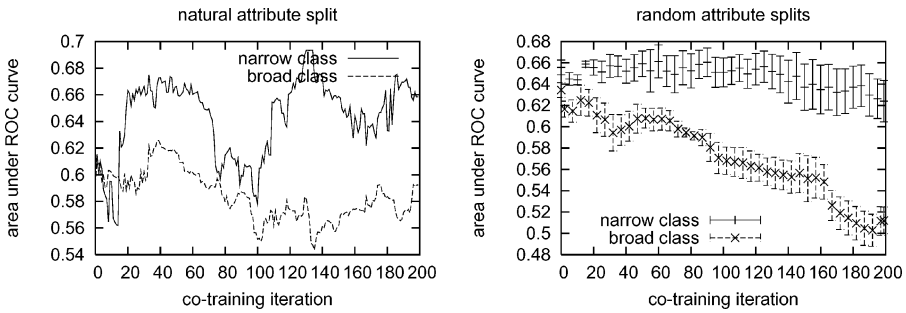

Figure 2. Co-training results for problem (3), yeast gene deletion.

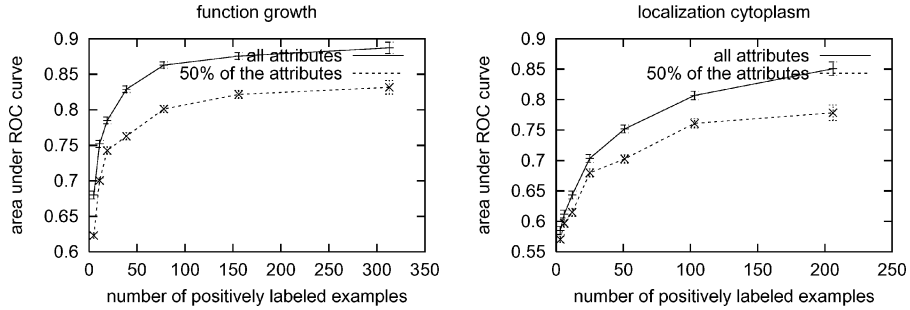

Figure 3. Compatibility of views for function growth and localization cytoplasm.

Like for transduction, our hypothesis was that co-training should at least not deteriorate classifier performance, and should improve performance when only few labeled training data are available. These hypotheses are refuted for the studied problems.

Why does co-training fail to improve classification results in this application domain? We investigate the degree to which the compatibility and independence assumptions are satisfied; we focus our investigations on recognizing the functional class "growth" and the localization class "cytoplasm" because, for these classes, sufficiently many labeled data are available to obtain reliable AUC estimates. Co-training depends on the existence of accurate decision functions using either attribute subset. Figure 3 compares the AUC performance of a decision function that uses all available attributes (depending on the labeled sample size) to the decision function that uses a randomly drawn 50% of the available attributes, averaged over 10 different randomly drawn attribute subsets. If the compatibility assumption was satisfied, then both learning curves would converge to an AUC performance of one. We observe a modest violation of the compatibility assumption: all attributes yield an asymptotic AUC performance of about 0.9 for both problems whereas 50% of the attributes allow for an asymptotic performance of roughly 0.85 for "growth" and 0.8 for "cytoplasm".

In each iteration, co-training can only improve the performance if at least one classifier labels at least one unlabeled instance correctly for which the other classifier currently errs. When both classifiers agree on every single unlabeled instance, then labeling them does not create new information for the peer classifier. Let E_1 and E_2 be binary random variables that indicate whether either classifier errs for a randomly drawn unlabeled instance. We use the Φ^2 statistics to measure whether E_1 and E_2 are independent. When E_1 and E_2 are independent, then $P(E_1, E_2) = P(E_1)P(E_1)$. Otherwise, Φ^2 (Eq. (1)), a natural measure of the dependency of the two classifiers, becomes positive.

$$\Phi^2 = \sum_{i=0}^{1} \sum_{j=0}^{1} \frac{(P(E_1 = i, E_1 = j) - P(E_1 = i)P(E_2 = j))^2}{P(E_1 = i)P(E_2 = j)} \tag{1}$$

When E_1 is always equal to E_2, then $\Phi^2 = 1$; when E_1 and E_2 are independent, then $\Phi^2 = 0$. Note that $X^2 = m\Phi^2$ is governed by the χ^2 distribution. The larger Φ^2 and the

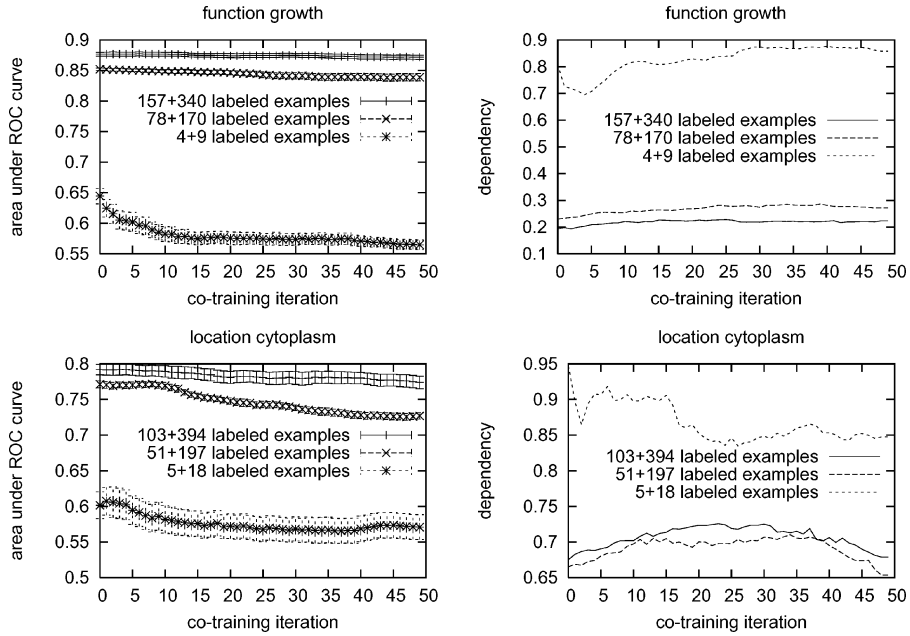

Figure 4. Co-training performance and dependency (Φ^2) for function and localization.

sample size m, the less likely are E_1 and E_2 to be independent. We focus on Φ^2 instead of X^2 because its dimension is independent of the sample size.

Figure 4 shows the dependency, Φ^2, over the co-training iterations and for various labeled sample sizes for problems (1) and (2). We see that, for both classification problems, the dependency is very strong for very small sample sizes. No clear trend in the development of Φ^2 over the co-training iterations can be observed. Our interpretation of this finding is that, for very small samples, the separator is dominated by the inductive bias of the Support Vector Machine. For larger samples, Φ^2 converges to a value that is dictated by the (lack of) independence of the attributes.

In order to identify the range of compatibility levels and Φ^2 values for which co-training works, we conduct experiments with a semi-artificial control problem. We use an experimental setting described in detail by Muslea, Kloblock, and Minton (2002a)—here, we sketch it only briefly: starting from the 20 newsgroups data set, we use the messages of four classes as positive (negative) examples in view V_1, of four distinct classes as positive (negative) examples of view V_2. We generate instances of the resulting classification problem by concatenating positive (negative) examples in V_1 with randomly drawn positive (negative) examples in view V_2. This random combination assures that the independence assumption is perfectly satisfied.

Muslea, Kloblock, and Minton (2002a) introduce a lack of independence by decomposing positive and negative examples into two or four "clumps" (corresponding to the classes of the original multi-class problem) and concatenating examples from V_1 only with examples

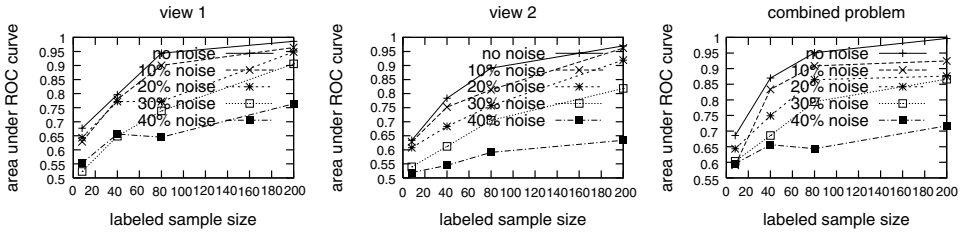

Figure 5. Compatibility for semi-artificial co-training problem.

of a corresponding clump in V_2. We found that this procedure does not introduce sufficiently strong dependencies to substantially deteriorate the performance of co-training. We therefore use a different mechanism: Each attribute in view V_2 is assigned a "peer attribute" in V_1 (V_1 and V_2 have equally many attributes). With probability p_{dep}, each attribute of V_2 assumes the value of its peer attribute in V_1; it assumes its own value with probability $1 - p_{dep}$. We obtain perfect independence for $p_{dep} = 0$ and complete dependence (each attribute in V_2 mirrors its peer from V_1) for $p_{dep} = 1$. We introduce a lack of compatibility by randomly flipping class labels.

Figure 5 displays the compatibility of the views, depending on the amount of introduced noise. Figure 6 shows how the effectiveness of co-training deteriorates when we increase the level of incompatibility (i.e., noise) and the dependencies between the view ("50% dependency" corresponds to $p_{dep} = 0.5$). Every curve of figure 6 is an average over 10 trials with different randomly drawn labeled samples.

We see that co-training performs extremely well when the compatibility and independence assumptions are met, its performance deteriorates as incompatibility and dependencies between the attribute sets are increased. We see that dependencies between the views lead to higher dependencies (higher values of Φ^2) between the classifiers. Figure 7 summarizes the relation between the Φ^2 value of the initial classifiers and the benefit of co-training (measured in terms of the difference between the AUC performance in the last iteration and the AUC performance of the initial decision function). Every point in figure 7 corresponds to one of the co-training curves of figure 6 (for the newsgroups problem) or figure 4 (for gene function and localization; problems 1 and 2).

We can clearly see a negative correlation between the AUC improvement obtained by co-training and the value of Φ^2; Figure 7 also shows the result of a linear regression over all data points. Co-training tends to improve the performance when the dependence between the initial decision function Φ^2 is below 50%; in our experiments, co-training is always beneficial for $\Phi^2 < 10\%$ and always detrimental for $\Phi^2 > 60\%$. For problems (1) and (2), the initial machines have dependencies Φ^2 of between roughly 0.2 and 0.85, causing co-training to deteriorate the performance.

These observations indicate that the dependence Φ^2 of the initial decision functions is a crucial parameter that determines whether co-training is able to improve the decision functions. Note, however, that in order to measure Φ^2 a labeled hold-out set is required. This requirement constrains the applicability of Φ^2 as a criterion for deciding whether co-training should be applied in a given situation.

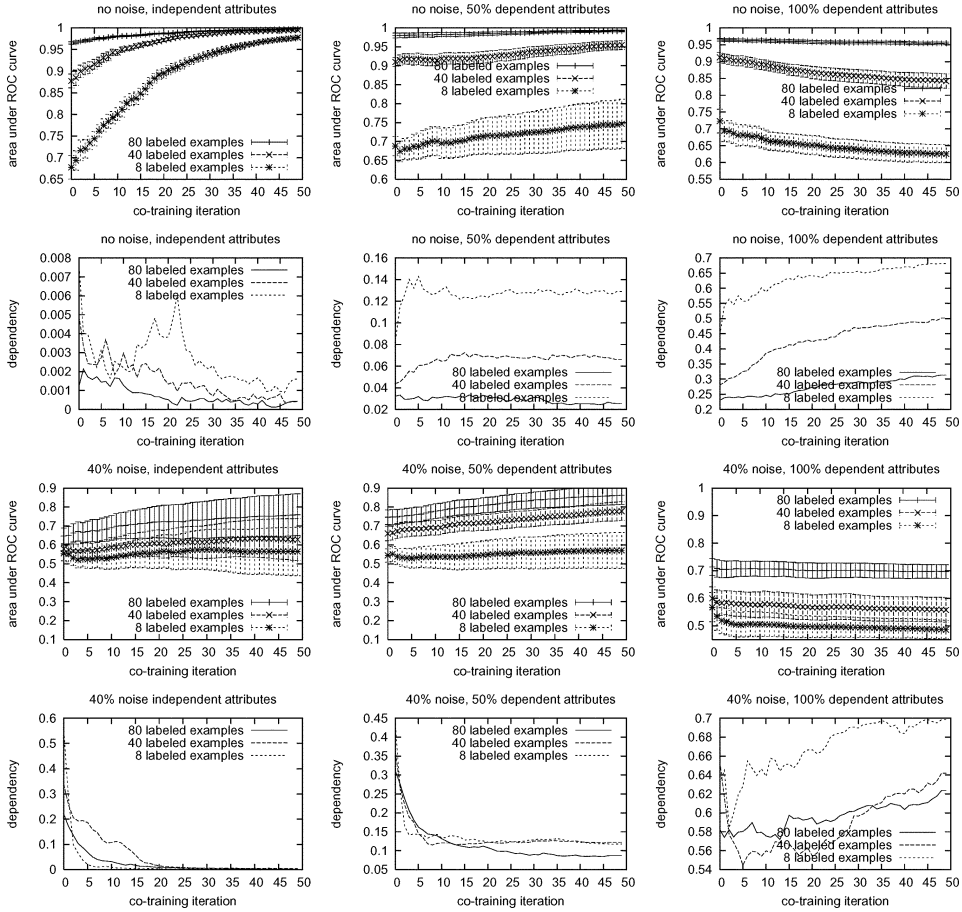

Figure 6. Co-training performance and classifier dependency (Φ^2) for the semi-artificial newsgroup problem.

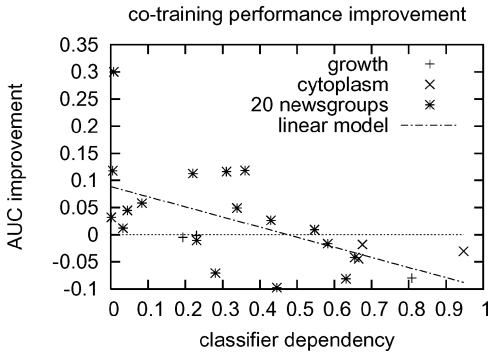

Figure 7. Relationship between Φ^2 (classifier dependency) and the performance improvement obtained by co-training.

6. Competitive evaluation

In this section, we discuss our competition results and review other winning solutions for the KDD Cup 2002.

For problems (1) and (2) (KDD Cup 2001 tasks 2 and 3, prediction of gene function and localization), Mark Krogel submitted classification results of a Support Vector Machine that used the interaction attributes generated by RELAGGS. Since the gene names were anonymized, we could not use text mining results; we did not use transduction or co-training either. The gene interaction attributes generated by RELAGGS made the essential contribution to the highest score achieved for function prediction, and the fourth highest score for localization prediction (Cheng et al., 2002).

For problem (3) (KDD Cup 2002 task 2), Krogel et al. (2003a) handed in an ordered list of gene identifiers that was produced by SVM^{light} for the narrow positive class problem. The data used here contained gene name information, information about interacting genes over two levels, and entries generated by information extraction. We did not include the text classification attribute and did not use transduction or co-training.

Like for all other teams, the tight competition schedule was a limiting factor for us. We only generated the classifier for the narrow class problem (we obtained the highest score of 0.68 among the participants), and used this classifier for both, the narrow and broad positive class. Using the narrow classifier for the broad class, we obtained an AUC performance of 0.62 which was still enough for a third rank for the overall task. Figure 8 depicts a comparison of the solutions handed in by all participants, provided by Craven (2002). Each dot corresponds to a participant; only the winning and honorably mentioned participants are named. 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.—which sums up to more than any team could achieve within the competition time frame.

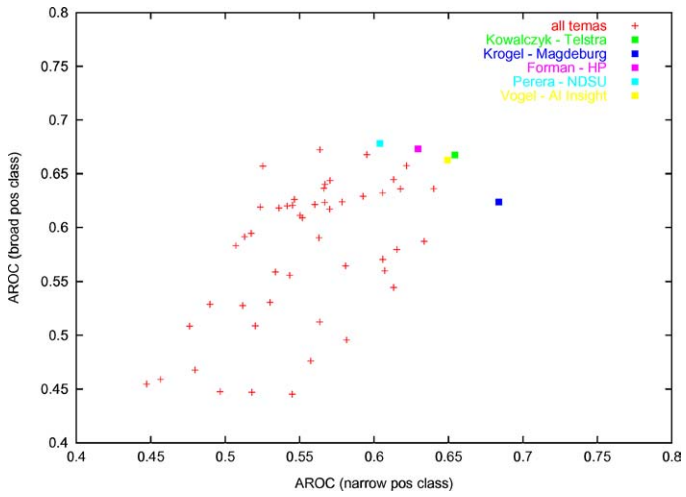

Figure 8. Results of KDD Cup 2002 participants on task 2 (graphics provided by Mark Craven).

The winning solution for KDD Cup 2002 (here, problem 3) used the function and localization attributes, a subset of the words occurring in the relevant abstracts, and one attribute for each gene that a given gene interacts with (Kowalczyk and Raskutti, 2002). Hence, this solution uses somewhat less relational information (no explicit information about attributes of interacting genes is represented) and uses a different approach to utilizing the abstracts. This winning solution obtains an AUC performance that is only slightly lower than the performance of the solution described here. Similarly, Perera et al. (2002) included attributes representing hand-selected words in the abstracts and one interaction attribute for each gene, a value of one expressing an interaction.

Vogel and Axelrod (2002) used function and localization attributes, information extracted from the gene name, the occurrence of 23 manually selected words in the abstracts that correspond to the given gene, and variables that indicate whether a gene interacts with certain groups of genes. Forman (2002), finally, also used function and localization attributes, the number of genes that a given gene interacts with, and text classification attributes. These results support our findings that the interaction information and information from the abstracts are of prominent importance for an accurate prediction.

All winning and "honorably mentioned" solutions of the KDD Cup 2002 generate attributes from both, the gene interaction information and MEDLINE abstracts. Our winning solution for the KDD Cup 2001 (where textual information could not be utilized as the genes were anonymized) generates attributes from the gene interaction relation. This provides strong evidence that these data sources provide most relevant information for a range of functional genomics problem and shows that propositionalization is an effective and scalable approach to utilizing this information.

7. Related results and lessons learned

One key element of a good solution to the focused problems lies in effective utilization of the gene interaction data. Since inductive logic programming involves several computationally hard problems (e.g., Scheffer, Herbrich, & Wysotzki, 1996), we have focused on the transformation-based RELAGGS algorithm (Krogel & Wrobel, 2001). It follows pioneering ILP approaches to transformation-based learning (Lavrac, Džeroski, & Grobelnik, 1991) and succeeding work (Kramer, Lavrač, & Flach, 2001) and combines propositionalization with SQL aggregation functions (Krogel & Wrobel, 2001; Knobbe, de Haas, & Siebes, 2001). In contrast to traditional ILP algorithms, the propositionalization approach allows us to use the Support Vector Machine as final classifier. For a discussion and comparison of propositionalization algorithms, see Krogel et al. (2003b).

The problem of extracting information from MEDLINE abstracts—or even full papers—is receiving considerable attention. The first problem that one naturally encounters when extracting attributes of genes from text is the named entity recognition problem (we did in fact not encounter this problem because a table that related genes to corresponding abstracts was provided for KDD Cup 2002). Currently, a wide range of biological named entity recognition systems is being evaluated in the BioCreative competition (Hirschman et al., 2003). In order to extract attributes, we follow a dictionary-based approach (Fukuda et al., 1998). Many other approaches (e.g., Hirschmann et al., 2002; Hahn, Romacker, &

Schulz, 2002; Blaschke & Valencia, 2002), including hidden Markov models (Leek, 1997) and rule learning (Craven et al., 2000), have been explored. We see more sophisticated approaches for information extraction from MEDLINE as a promising research direction.

Many positive results on semi-supervised learning have been obtained—e.g., by Cooper and Friedman (1970) and Collins and Singer (1999) and, for co-training, by Kiritchenko and Matwin (2002) and Mladenic (2002). Other recent results (e.g., Nigam, Lafferty, & McCallum, 1999; Bruce, 2001; Kockelkorn, Lüneburg, & Scheffer, 2003) indicate that most often no benefit is observed when the labeled sample is large. This may be linked to invalid model assumptions (Cozman, Cohen, & Cirelo, 2003) and, for co-training, to dependent and incompatible views (Muslea, Kloblock, & Minton, 2002b). Our experiments show that transduction fails to improve the recognition rate for the studied functional genomics problems even when only few labeled data are available. We obtained negative results with co-training whenever the dependence Φ^2 of the initial decision function exceeds 60%.

From our experience in the KDD Cup competition and retrospective studies, we draw a number of lessons learned.

1. Our winning solution for the KDD Cup 2001 as well as all "honorably mentioned" solutions to the 2002 Cup aggregated attributes from the interaction relation. Our studies show that the performance of classifiers for all studied functional genomics problems deteriorate without these attributes. This supports our first lesson learned. *The interactions between genes play a crucial role for functional genomics problems. Propositionalizing the relational data by first computing joins of the tables, and then collapsing the joins by aggregation functions is an effective and scalable approach.*
2. All winning and "honorably mentioned" solutions for the KDD Cup 2002 generated additional attributes from MEDLINE abstracts. Our studies showed a decrease in classifier accuracy when these attributes are left out. This leads us to conclude that, *even in addition to available attributes in databases, MEDLINE papers contain information that is relevant for predicting properties of genes. Even fairly simple dictionary-based extractors generate attributes that improve such predictions substantially.* We believe that more sophisticated extractors will further improve biological models.
3. *A caveat for utilizing knowledge in document archives: background document archives may contain the class labels that a classifier is to predict. If this is the case, then the class labels of held-out instances will "leak" into the training process. The resulting performance estimates will be extremely optimistic. The resulting classifiers will perform substantially worse for new instances, the class labels of which are not yet disclosed in available abstracts.* An evaluation methodology is required to obtain useful performance estimates for knowledge discovery methods that utilize document archives.
4. Previously obtained empirical results have led to a faith of some researchers that transduction—or even semi-supervised learning in general—is not harmful for practically relevant problems and improves performance at least when the labeled sample is small. *Our negative results on all studied genomics problems refute this hypothesis.* Positive experiences with co-training have given rise to the hypothesis that co-training is generally beneficial in practice, when the labeled sample is small and the attributes can be split into subset with little dependencies. *Our results indicate that the dependence*

between the initial decision functions measured by the Φ^2 statistics is crucial to the benefit of co-training. Co-training is most beneficial when $\Phi^2 < 10\%$; it is detrimental when $\Phi^2 > 60\%$. When $\Phi^2 < 50\%$, co-training tends to improve the performance on average.

Acknowledgments

We would like to thank Mark Craven and David Page for their support, Marco Landwehr and Marcus Denecke for their contributions to the implementation, and Ulf Brefeld for his support of the experiments. Thanks to Nada Lavrac, Hiroshi Motoda, and the anonymous reviewers for helpful comments, and to Juffi Fürnkranz for inspiring discussions about co-training. Tobias Scheffer is supported by Emmy Noether Grant SCHE540/10-1 of the German Science Foundation DFG.

References

Blaschke, C., & Valencia, A. (2002). The frame-based module of the suiseki information extraction system. *IEEE Transactions on Intelligent Systems, 17*, 14–20.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Conference on Computational Learning Theory* (pp. 92–100).

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

Bruce, R. (2001). Semi-supervised learning using prior probabilities and EM. In *Proceedings of the IJCAI Workshop on Text Learning*.

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.

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Cooper, D., & Friedman, J. (1970). On the asymptotic improvement in the outcome of supervised learning provided by additional nonsupervised learning. *IEEE Transactions on Computers, C-19*, 1055–1063.

Cozman, F., Cohen, I., & Cirelo, M. (2003). Semi-supervised learning of mixture models. In *Proceedings of the International Conference on Machine Learning* (pp. 99–106).

Craven, M. (2002). The genomics of a signalling pathway: A KDD Cup challenge task. *SIGKDD Explorations, 4:2*, 97–98.

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (2000). Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence, 118*, 69–113.

Džeroski, S., & Lavrač, N. (eds.). (2001). *Relational Data Mining*. Springer.

Forman, G. (2002). Feature engineering for a gene regulation prediction task. *SIGKDD Explorations, 4:2*, 106–107.

Fukuda, K., Tsunoda, T., Tamura, A., & Takagi, T. (1998). Toward information extraction: Identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing*.

Hahn, U., Romacker, M., & Schulz, S. (2002). Creating knowledge repositories from biomedical reports: The MEDSYNDICATE text mining system. In *Proceedings of the Pacific Symposium on Biocomputing*.

Hirschman, L., Valencia, A., Blaschke, C., Light, M., & Yeh, A. (2003). Critical assessment of information extraction systems in biology. www.pdg.cnb.uam.es/BioLINK.

Hirschmann, L., Park, J., Tsujii, J., Wong, L., & Wu, C. (2002). Accomplishments and challenges in literature data mining for biology. *Bioinformatics, 18*, 1553–1561.

Joachims, T. (1999a). Making large-scale SVM learning practical. *Advances in Kernel Methods—Support Vector Learning*. MIT Press.

Joachims, T. (1999b). Transductive inference for text classification using support vector machines. In *Proceedings of the International Conference on Machine Learning* (pp. 200–209).

Kiritchenko, S., & Matwin, S. (2002). Email classification with co-training. Technical Report, University of Ottawa.

Knobbe, A., de Haas, M., & Siebes, A. (2001). Propositionalisation and aggregates. In *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 277–288).

Kockelkorn, M., Lüneburg, A., & Scheffer, T. (2003). Using transduction and multi-view learning to answer emails. In *Proceedings of the European Conference on Principle and Practice of Knowledge Discovery in Databases*.

Kowalczyk, A., & Raskutti, B. (2002). One class SVM for yeast regulation prediction. *SIGKDD Explorations, 4:2*, 99–100.

Kramer, S., Lavrač, N., & Flach, P. A. (2001). Propositionalization approaches to relational data mining. In N. Lavrač and S. Džeroski (eds.), *Relational data mining*. Springer.

Krogel, M.-A., Landwehr, M., Denecke, M., & Scheffer, T. (2003a). Combining data and text mining techniques for yeast gene regulation prediction: A case study. *SIGKDD Explorations, 4:2*, 104–105.

Krogel, M.-A., Rawles, S., Zelezny, F., Flach, P., Lavrac, N., & Wrobel, S. (2003b). Comparative evaluation of approaches to propositionalization. In *Proceedings of the International Conference on Inductive Logic Programming* (pp. 197–214).

Krogel, M.-A., & Wrobel, S. (2001). Transformation-based learning using multirelational aggregation. In *Proceedings of the International Conference on Inductive Logic Programming* (pp. 142–155).

Lavrac, N., Džeroski, S., & Grobelnik, M. (1991). Learning nonrecursive definitions of relations with LINUS. In *Proceedings of the European Working Session on Learning* (pp. 265–281).

Leek, T. (1997). Information extraction using hidden Markov models, Master's Thesis. University of California at San Diago.

Mladenić, D. (2002). Learning word normalization using word suffix and context from unlabeled data. In *Proceedings of the International Conference on Machine Learning* (pp. 427–434).

Muslea, I., Kloblock, C., & Minton, S. (2002a). Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the International Conference on Machine Learning* (pp. 435–442).

Muslea, I., Kloblock, C., & Minton, S. (2002b). Adaptive view validation: A first step towards automatic view detection. In *Proceedings of the International Conference on Machine Learning* (pp. 443–450).

Nigam, K., & Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. In *Proceedings of Information and Knowledge Management*.

Nigam, K., Lafferty, J., & McCallum, A. (1999). Using maximum entropy for text classification. *IJCAI-99 Workshop on Machine Learning for Information Filtering*.

Perera, A., Denton, A., Kotola, P., Jockheck, W., Granda, W., & Perrizo, W. (2002). P-tree classification of yeast gene deletion data. *SIGKDD Explorations, 4*, 108–109.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*, 130–137.

Provost, F., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing inductive algorithms. In *Proceedings of the International Conference on Machine Learning* (pp. 445–453).

Scheffer, T., Herbrich, R., & Wysotzki, F. (1996). Efficient θ-subsumption based on graph algorithms. In *Proceedings of the International Workshop on Inductive Logic Programming* (pp. 212–218).

Vogel, D., & Axelrod, R. (2002). Predicting the effects of gene deletion. *SIGKDD Explorations* (pp. 101–103).

Witten, I., & Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

Received April 7, 2003
Accepted April 8, 2004
Final manuscript April 20, 2004