On the Utility of Partially Labeled Data for Classification of Microarray Data

Ludwig Lausser*, Florian Schmid*, and Hans A. Kestler**

Research Group Bioinformatics and Systems Biology
Institute of Neural Information Processing, University of Ulm, Germany
{ludwig.lausser,florian-1.schmid,hans.kestler}@uni-ulm.de

Abstract. Microarrays are standard tools for measuring thousands of gene expression levels simultaneously. They are frequently used in the classification process of tumor tissues. In this setting a collected set of samples often consists only of a few dozen data points. Common approaches for classifying such data are supervised. They exclusively use categorized data for training a classification model. Restricted to a small number of samples, these algorithms are affected by overfitting and often lack a good generalization performance. An implicit assumption of supervised methods is that only labeled training samples exist. This assumption does not always hold. In medical studies often additional unlabeled samples are available that can not be categorized for some time (i.e., ”early relapse” vs. ”late relapse”). Alternative classification approaches, such as semi-supervised or transductive algorithms, are able to utilize this partially labeled data. Here, we empirically investigate five semi-supervised and transductive algorithms as ”early prediction tools” for incompletely labeled datasets of high dimensionality and low cardinality. Our experimental setup consists of cross-validation experiments under varying ratios of labeled to unlabeled examples. Most interestingly, the best cross-validation performance is not always achieved for completely labeled data, but rather for partially labeled datasets indicating the strong influence of label information on the classification process, even in the linearly separable case.

1 Introduction

In modern clinical studies the progress of a disease is monitored by DNA microarrays. These tools are high-throughput molecular biology devices for measuring thousands of gene expression levels simultaneously. The data collected within a clinical study usually does not exceed a few dozen gene expression profiles. These profiles can for example be used to discriminate the patients into clinical relevant groups (e.g. ”inflammation” vs. ”tumor”). In this setting a classifier performing this task has to deal with data of high dimensionality and low cardinality.

The standard learning scheme for training such a classifier is the supervised one. Here, a classifier is trained on a set of labeled samples. An implicite assumption of this scheme is that a training set of appropriate size exists.

* Contributed equally.

** Corresponding author.

F. Schwenker and E. Trentin (Eds.): PSL 2011, LNAI 7081, pp. 96–109, 2012.

Clinical relevant classification tasks do not always perfectly fit into this basic supervised scenario. In many cases the unlabeled data is available years before the corresponding diagnoses. For example, it could be of interest how a patient reacts to a certain treatment. It is important to know if a patient will suffer from an "early relapse" or have a "late relapse" of a disease. Applying the standard supervised scheme, the earliest moment to start the analysis of the collected dataset is after receiving the last label. Often it is desirable to receive preliminary predictions within an earlier stage.

Alternative learning schemes, like semi-supervised learning, are able to handle partially labeled datasets. They utilize the positional information of a data point during training. Although semi-supervised algorithms fit better into the setting described above, they are normally applied in fields with much more available observations. So far it is unclear how these algorithms perform on small sample sizes.

In our study we investigate the usability of semi-supervised algorithms as early predictors for small (microarray) datasets. Five of these classifiers were tested on seven public available microarray datasets under varying conditions. We utilize an experimental setup consisting of adapted $l \times k$ cross-validation experiments allowing to assess the performance of semi-supervised and transductive algorithms under varying ratios of labeled to unlabeled examples.

2 Methods

In general a classifier c can be seen as a function mapping $c : \mathscr{X} \rightarrow \mathscr{Y}$ from an input space $\mathscr{X}$ to the space of class labels $\mathscr{Y}$. In the following only binary classifiers will be considered and the space of class labels will be fixed to the Boolean space $\mathscr{Y} := \{0,1\}$. Normally it is assumed that $\mathscr{X} \times \mathscr{Y}$ is associated with a fixed but unknown probability distribution. A common objective for a classifier is to minimize its *generalization risk* according to this distribution

$$\mathscr{R} = Pr(c(X) \neq Y). \tag{1}$$

Here (X,Y) denotes a random example drawn *iid* from $\mathscr{X} \times \mathscr{Y}$. As the distribution of $\mathscr{X} \times \mathscr{Y}$ is unknown, the generalization risk of this classifier has to be estimated according to a finite set $\mathscr{S}_{te} = \{(x'_i, y'_i)\}_{i=1}^{m'}$ of test samples.

$$R_{emp} = \frac{1}{m'} \sum_{(x',y') \in \mathscr{S}_{te}} \mathbb{I}_{[c(x') \neq y']}. \tag{2}$$

Here $\mathbb{I}_{[]}$ denotes the indicator function, which is equal to 1, if the condition in $[]$ is fulfilled and 0 otherwise. R_{emp} is called the *empirical risk*.

During an initial training phase a classifier has to be adapted to the current classification task. This is done according to a finite set of training examples $\mathscr{S}_{tr} = \{(x_i, y_i)\}_{i=1}^{m}$ with $\mathscr{S}_{tr} \cap \mathscr{S}_{te} = \emptyset$. Different learning paradigms exist, varying in how the available samples are incorporated. We will use $\mathscr{X}_{tr} := \{x'_i\}_{i=1}^{m}$ and $\mathscr{X}_{te} := \{x'_i\}_{i=1}^{m'}$ as additional notation to denote the unlabeled training and test samples.

2.1 Supervised Learning

Supervised learning schemes only incorporate knowledge from labeled samples. A prediction of a supervised classifier will be denoted by $c_{\mathscr{S}_{tr}}(x)$. They can be distinguished by how they use the training data in categorization.

Inductive learning: In this scheme the classifier c is chosen from a concept class $\mathscr{C}$ and adapted according to $\mathscr{S}_{tr}$ within a learning procedure l. Once trained, the classifier can be abstracted from the original training data; it can be independently applied on $\mathscr{S}_{te}$.

$$l(\mathscr{C}, \mathscr{S}_{tr}) \rightarrow c \tag{3}$$

Model-free learning: Training and application of a classifier can not be separated in this setting. The label of a single test sample x' is directly predicted according to measurements on $\mathscr{S}_{tr}$.

$$\mathscr{S}_{tr} \times x' \rightarrow \hat{y}' \tag{4}$$

2.2 Semi-supervised Learning

The term semi-supervised learning will here be used for algorithms incorporating knowledge from both labeled and unlabeled samples during their training phase. A prediction of such a classifier will be denoted by $c_{\mathscr{S}_{tr}, \mathscr{X}_{te}}(x)$. This category will subsume the real semi-supervised algorithms and the transductive learning algorithms.

Semi-supervised learning: This term is normally used for inductive algorithms that can also incorporate knowledge of unlabeled samples within their training. The final classifier is again independent of the training data used to adapt it and can be applied without knowing it.

$$l(\mathscr{C}, \mathscr{S}_{tr}, \mathscr{X}_{te}) \rightarrow c \tag{5}$$

Transductive learning: This can be seen as the generalization of model-free learning. Here, the label of a single test sample x' is determined according to measurements on the labeled and unlabeled training data.

$$\mathscr{S}_{tr} \times \mathscr{X}_{te} \rightarrow \hat{\mathscr{Y}} \tag{6}$$

2.3 $l \times k$ Cross-Validation

In supervised classification one standard evaluation method for datasets of small sample size is the k-fold cross-validation experiment (see e.g. [3, 7, 9]). The benefit of this method is its guarantee that each sample is used as training as well as test sample. Subsampling effects resulting in misleading performance measures are minimized.

For this experiment the available data of n samples is split into k-folds ($2 \leq k \leq n$) of approximately equal size (Figure 1). A subset of $k-1$ folds is used as a labeled training set. The remaining fold is used as an independent test set. The procedure is repeated for each of the k possible splits into training sets $\mathscr{S}^i_{tr}$ and test sets $\mathscr{S}^i_{te}$, with $i \in \{1,\dots,k\}$. In this way each sample is used once as an test sample; the cross-validation results in one prediction per datapoint. These predictions are then used to estimate the risk of the classifier.

$$R_{CV} = \frac{1}{n}\sum_{i=1}^{k}\sum_{(x,y)\in\mathscr{S}^i_{te}} \mathbb{I}_{\left[c_{\mathscr{S}^i_{tr}}(x)\neq y\right]} \tag{7}$$

The estimate can be affected by the particular choice of splits. In order to minimize their influence, the k-fold cross-validation is repeated on l different permutations (runs) of the original dataset. The risk of the classifier is then estimated by the average over the l cross-validation errors. The final experiment is called a $l \times k$ cross-validation.

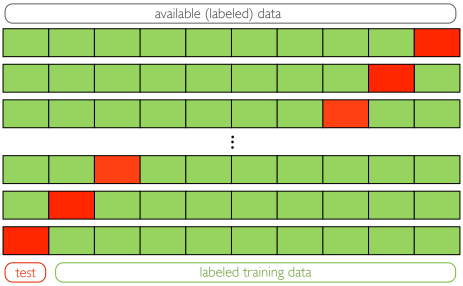

Fig. 1. Basic concept of a k-fold cross-validation. The available data is split into k folds of approximately equal size. The samples of $k-1$ folds are used as (labeled) training set for a classification model. The remaining fold is used as an independent set of test samples. The procedure is repeated for all k possible splits. The number of misclassifications over the whole dataset is used for estimating the risk of the classifier.

2.4 Cross-Validation Experiments for Semi-supervised Classifiers

In order to gain insight into the usability of semi-supervised algorithms as early prediction methods, two different cross-validation types were used (Figure 2).

Standard cross-validation: In this setting a classifier $c_{\mathscr{S}^i_{tr},\mathscr{X}^i_{te}}(x)$ is adapted to all available samples. The labeled samples come from $\mathscr{S}^i_{tr}$ while the unlabeled samples come from $\mathscr{X}^i_{te}$. The tests are performed on $\mathscr{S}^i_{te}$.

$$R_{CV} = \frac{1}{n}\sum_{i=1}^{k}\sum_{(x,y)\in\mathscr{S}^i_{te}} \mathbb{I}_{\left[c_{\mathscr{S}^i_{tr},\mathscr{X}^i_{te}}(x)\neq y\right]} \tag{8}$$

Inverted cross-validation: The algorithms were also compared in a setting we call inverted cross-validation. Here for each fold i the $\mathcal{S}_{te}^{i}$ was used as labeled training set and the samples of $\mathcal{X}_{tr}^{i}$ were used unlabeled. The algorithm receives more unlabeled than labeled training examples. The error of a classifier is estimated according to

$$R_{CV^{-1}} = \frac{1}{n(k-1)} \sum_{i=1}^{k} \sum_{(x,y)\in\mathcal{S}_{tr}^{i}} \mathbb{I}_{\left[c_{\mathcal{S}_{te}^{i},\mathcal{X}_{tr}^{i}}(x)\neq y\right]}, \tag{9}$$

The results of the inverted cross-valdiation will be indexed by $l \times -k$. The learning task given by the inverted cross-validation setting better fits to the typical proportion of labeled and unlabeled training samples of known semi-supervised applications. Its benefit is the more systematic evaluation of the performance of a classifier than for example the evaluation done by experiments on randomly drawn splits.

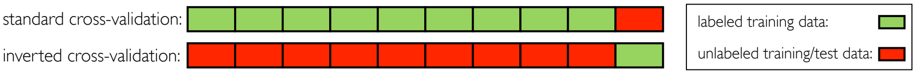
standard cross-validation:
inverted cross-validation:
labeled training data: 🟩
unlabeled training/test data: 🟥

Fig. 2. Differences between the standard cross-validation and the inverted cross-validation: The figure shows the splits of the available data for the two kinds of experiments in the semi-supervised scenario. While the standard cross-validation setting utilizes $k-1$ folds as labeled training data and 1 fold as unlabeled training (test) data, the inverted cross-validation uses one fold as labeled training data and $k-1$ as unlabeled training (test) data.

2.5 Algorithms

We have tested following five algorithms on their usability as ”early prediction tools”:

Transductive support vector machines (tsvm) [10]: The algorithm applied here is a version of the standard (linear) inductive svm [16]. The basic strategy of both classification methods is to find a linear hyperplane ω maximizing the margin to the given samples. If it is not possible to separate the data correctly, a tradeoff between the misclassified datapoints (distance to margin) and the diminished margin has to be found.

In contrast to the inductive version, the class labels $\mathcal{Y}'$ of the test samples are directly included in the optimization process of the tsvm algorithm. They become estimated by solving an optimization task, described by the following system of linear equations:

$$\begin{aligned} \min_{\omega,\theta,\xi,\xi',\mathcal{Y}'} \quad & \|\omega\|_2^2 + C\sum_{i=1}^{m}\xi_i + C'\sum_{i=1}^{m'}\xi_i' \\ \text{s.t.} \quad & \forall_{i=1}^{m} : y_i(\omega^T x_i) - \theta \geq 1 - \xi_i, \xi_i \geq 0 \\ & \forall_{i=1}^{m'} : y_i'(\omega^T x_i') - \theta \geq 1 - \xi_i', \xi_i' \geq 0 \end{aligned}$$

The available labeled samples x_i and the unlabeled samples x_i' are separately treated within this optimization problem. For each kind of data there is a combination of cost parameter and distance measure, called C and ξ_i for the labeled samples and C' and ξ_i' for the unlabeled ones. The binary variables y_i' are chosen within the algorithm according to the solution of the optimization task. The cost parameters of this algorithm were fixed to a value of 1 in our experiments.

Penalized likelihood based pattern classification algorithm (plc) [2]: This algorithm estimates the likelihood $P_l = P(Y = 1|X = x_l)$ for each given sample. As the algorithm does not estimate a likelihood function, it belongs to the category of transductive algorithms. The estimates are determined in a penalized optimization task.

$$\min \qquad J = \log(L) - \lambda S, \tag{10}$$

where L is the likelihood for the labeled samples $\mathscr{S}_{tr}$

$$L = \prod_{l=1}^{m} P_l^{y_l}(1-P_l)^{1-y_l} \tag{11}$$

and S (smoothness) is a penalty on the roughness of the estimations

$$S = \frac{1}{K}\sum_{l=1}^{m+m'}\sum_{l' \in \mathbb{K}(x_l)} (P_l - P_{l'})^2. \tag{12}$$

The smoothness of each prediction is calculated according to the neighbourhood $\mathbb{K}$. The size K of this neighbourhood is determined within the algorithm. As proposed in [2], we set parameter $\lambda = 0.4$.

Transductive k-nearest neighbors classifier (tknn) [14]: This version of the k-nearest neighbor classifier (e.g. [8]) determines the label of a single sample according to measurements on its k_l labeled and k_u unlabeled neighbors. The influence of the single samples on the classification of a datapoint x_i is thereby regulated according to a weight vector w_i.

$$w_{ij} = \begin{cases} K(x_i,x_j), & \text{if } x_j \in \mathscr{X}_{tr} \wedge x_j \in \mathbb{K}_l(x_i,k_l) \\ aK(x_i,x_j), & \text{if } x_j \in \mathscr{X}_{te} \wedge x_j \in \mathbb{K}_u(x_i,k_u) \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Here $K(x_i,x_j)$ denotes following distance kernel function

$$K(x_i,x_j) = \frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{||x_i - x_j||^2}{2h^2}\right). \tag{14}$$

The label of a sample x_i is determined within a label propagation process iteratively calculating the class membership probabilities p_{ir}, $r \in \{0,1\}$.

$$p_{ir}^{[t+1]} \leftarrow \sum_{j=1}^{m+m'} v_{ij} p_{jr}^{[t]} \tag{15}$$

Here v_{ij} corresponds to the row normalized value of w_{ij}. The initial class membership probabilities is initialized by 0.5 for unlabeled samples and fixed to 0 and 1 for labeled ones. The propagation process is repeated until the last class membership probability of an unlabeled test sample has converged. We have fixed the number of labeled (unlabeled) neighbors to $k_l = 1$ ($k_u = 3$). The influence of the unlabeled neighbors was regularized by $a \in \{0.3, 0.7, 1.0\}$.

Yarowsky's algorithm (yar) [18]: This algorithm is a general iterative procedure for modifying an inductive classifier $c_{\mathscr{S}_{tr}}$ into a semi-supervised one. The inductive algorithm must therefore be able to give confidence values $p_{\mathscr{S}_{tr}}$ for its predictions. We used a svm, which returns class probabilities, as a base classifier [12] . Yarowsky's algorithm iteratively includes unlabeled samples into the (labeled) training set, if they allow a prediction above a fixed confidence level d.

$$\mathscr{S}_{tr}^{[t+1]} = \mathscr{S}_{tr}^{[0]} \cup \{(x', \hat{y}) \mid x' \in \mathscr{X}_{te},\ \hat{y} = c_{\mathscr{S}_{tr}^{[t]}}(x'),\ p_{\mathscr{S}_{tr}^{[t]}}(x') \geq d\} \tag{16}$$

The classifier is retrained on the modified training sets until $\mathscr{S}_{tr}^{[t+1]} = \mathscr{S}_{tr}^{[t]}$. In our experiments the confidence level is chosen from $d \in \{0.6, 0.7, 0.8, 0.9\}$.

Mincut algorithm (mc) [5]: This algorithm is based on a weighted graph connecting the samples of the dataset. The graph is extended by a node for each class label of the dataset. These nodes are connected with all samples of the corresponding class. The weights of these edges are set to infinity. During the training process the graph is pruned according to a max-flow algorithm. The remaining paths to one of the label nodes determine the labels of the samples. The graph we have chosen in our experiments is based on the dataset's distribution of pairwise (Euclidian) distances. An edge between two datapoints is drawn, if the corresponding distance is smaller than the q-quantile of this distribution ($q \in \{0.1, 0.2, 0.3\}$).

3 Experimental Setup

We compared the five semi-supervised algorithms mentioned before in a series of cross-validation experiments on seven microarray datasets (see Table 1). The series include $10 \times k$ cross-validations for k = 10,...,2 and inverted $10 \times -k$ cross-validations for $k = 2, \ldots, 10$. The single experiments differ in their fold number and the number of available training and test samples; while the number of (unlabeled) training samples decreases with k, the number of (labeled) test samples increases. An overview on the available positive and negative samples in the $10 \times k$ setting can be found in Figure 3. Over the seven datasets the mean number of labeled training samples per fold varies from 25.6 to 91.8 within the 10×10 experiment and 14.0 to 51.0 in the 10×2 experiment; the corresponding mean number of unlabeled test samples per fold vary from 2.8 to 10.2 (10×10) and 14.0 to 51.0 (10×2). In the inverted cross-validation the numbers of training and test samples are reversed.

Table 1. Key properties of the utilized datasets

Dataset	Features	Positive samples	Negative samples
Armstrong (*AR*) [1]	12582	24	48
Bittner (*BI*) [4]	8067	19	19
Nutt (*NU*) [11]	12625	14	14
Pancreas (*PA*) [6]	169	37	25
Shipp (*SH*) [13]	7129	58	19
Singh (*SI*) [15]	12600	52	50
West (*WE*) [17]	7129	25	24

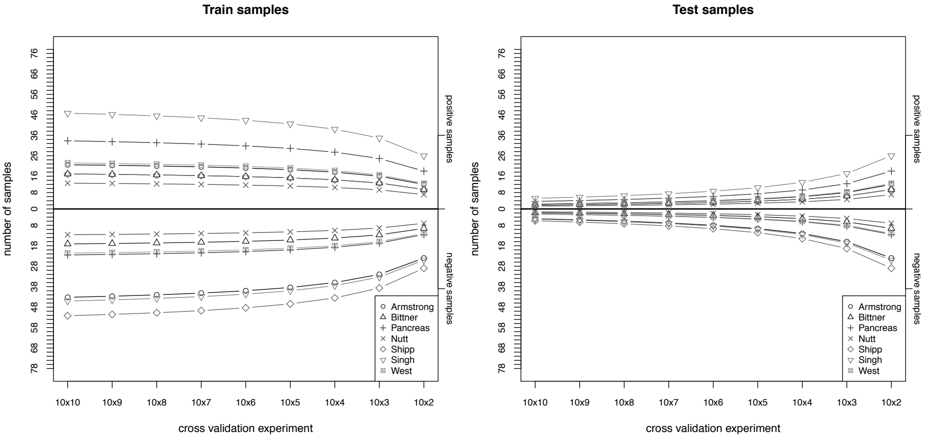

Fig. 3. Number of samples (microarray datasets): The figure shows the influence of the chosen k within a standard $l \times k$ cross-validation experiment on the number of available (labeled) training and (unlabeled) test examples per fold. While the number of (labeled) training examples increases with the number of folds, the number of (unlabeled) test examples decreases. The number of (labeled) training samples within a standard $l \times k$ cross-validation is corresponding to the number of (unlabeled) test samples within a inverted $l \times -k$ cross-validation and vice versa.

4 Results

The results of the cross-validation experiments can be found in Figures 4 and 5. The accuracies were additionally compared to the results of a ”prevalence” classifier always predicting the class label of the larger class. The accuracy of the constant classifier can be seen as a lower bound for a beneficial (”meaningful”) classification performance. In the case of imbalanced data, this bound is tighter than the 50% bound. For the semi-supervised classifiers following results could be observed:

Two of the algortihms, *tsvm* and *plc*, showed a better performance than the ”prevalence” classifier on all datasets. Despite of its performance in the 10×-9 and 10×-10 cross-validation experiment on *SH*, the same is true for the *tknn* algorithm. For some datasets, the performance of *yar* and *mc* did not cross the minimal accuracy level; while

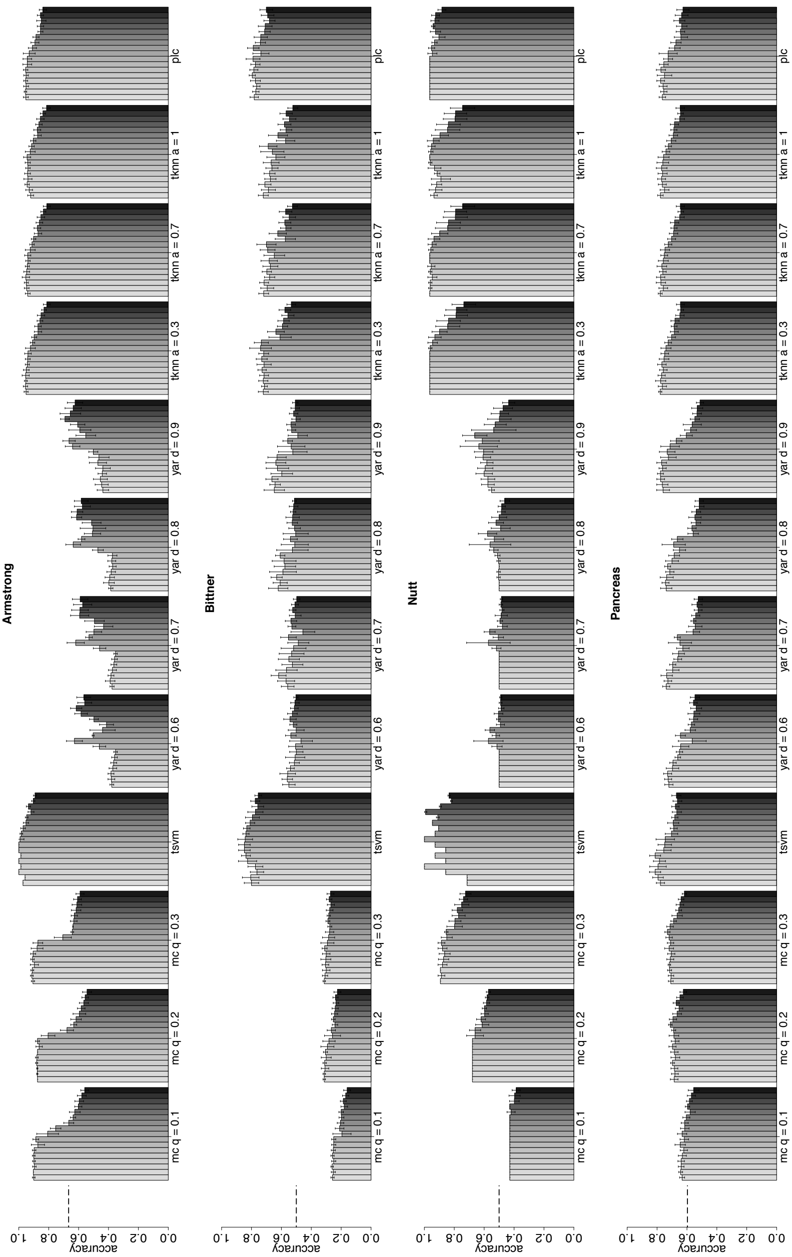

Fig. 4. Results of the $10 \times k$ cross-validation experiments ($k \in 10, \ldots, \pm 2, \ldots, -10$) for datasets Armstrong, Bittner, Nutt, Pancreas. A legend for the different experimental setups can be found in Figure 5.

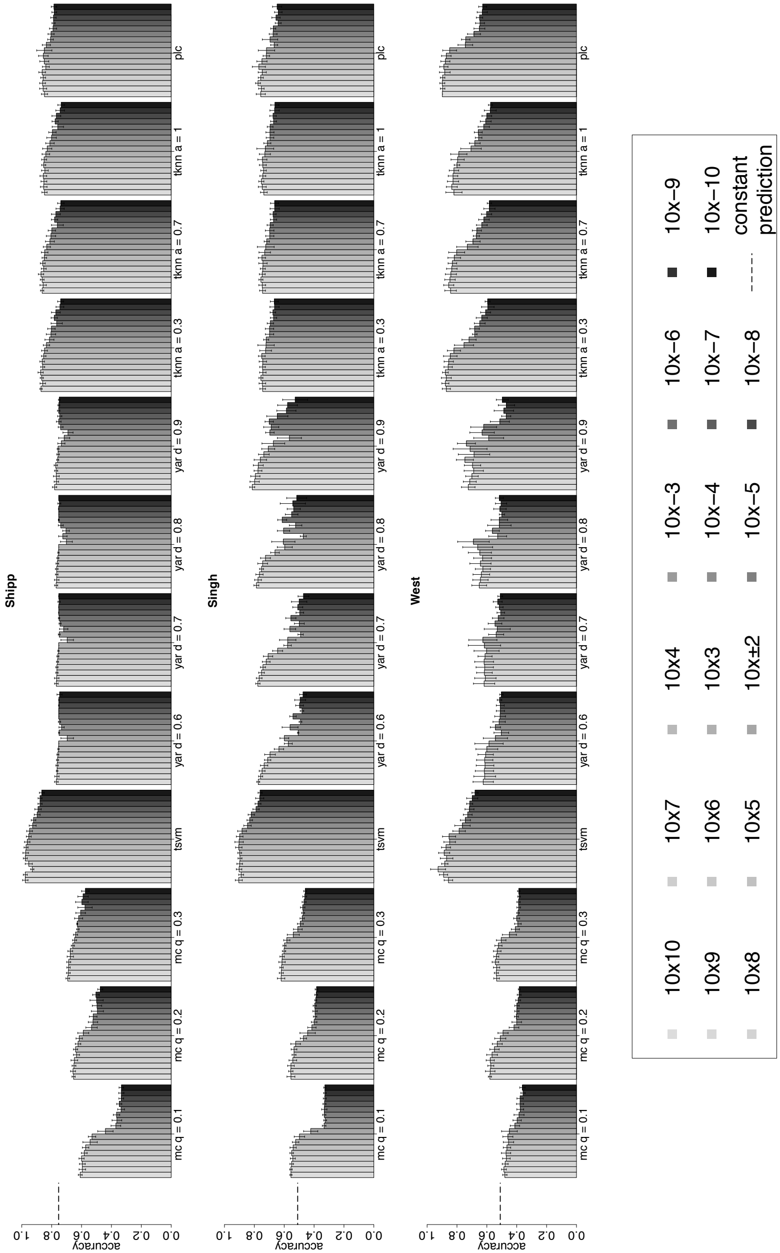

Fig. 5. Results of the $10 \times k$ cross-validation experiments ($k \in 10, \ldots, \pm 2, \ldots, -10$) for datasets Shipp, Singh, West

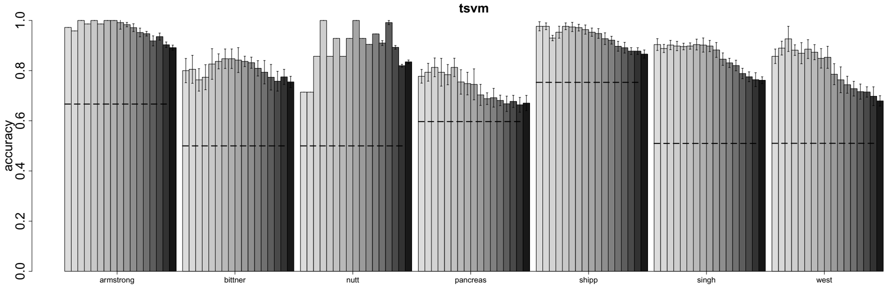

Fig. 6. Results of the $10 \times k$ cross-validation experiments ($k \in 10,\ldots,\pm 2,\ldots,-10$): The figure summarizes the results of the tsvm. A legend for the different experimental setups can be found in Figure 6. The dotted line corresponds to the results of a "prevalence" classifier.

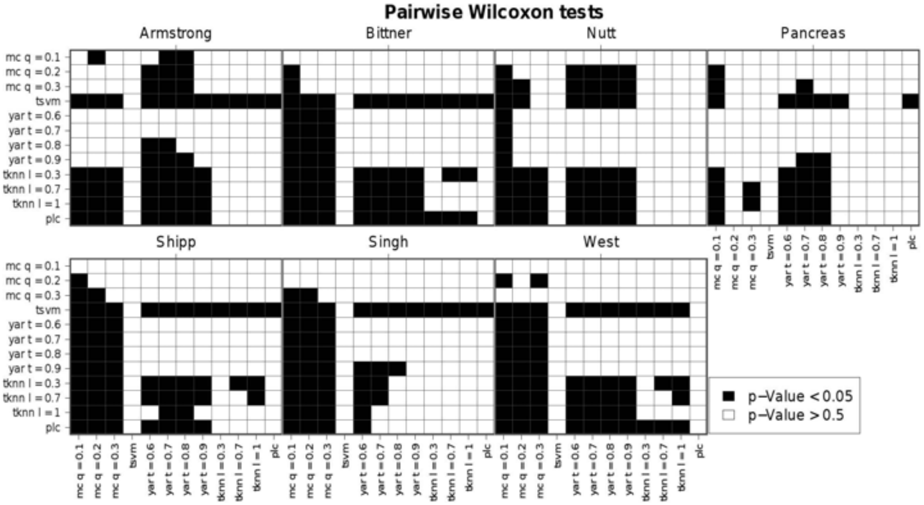

Fig. 7. Results of paired one-sided Wilcoxon rank tests done for the cross-validation accuracies (over all k) of every pair of algorithms on each dataset. The black color in cell ij denotes that the median cross-validation accuracy of algorithm i is significantly higher than the median cross-validation accuracy of algorithm j. For each dataset, the tests were corrected for multiple-testing (Bonferroni $n = 132$).

mc did not attain good results on *BI* and *SH*, *yar* did not excel on *AR*. On the other datasets these algorithms showed a low performance in the inverted cross-validation setting and again did not achieve the minimal accuracy level. In general lower accuracies were achieved within the inverted cross-validations than in the standard cross-validations. An exception is *yar* for the datasets *AR* and *NU*.

The *plc* is with a mean range (max accuracy − min accuracy) of about 13.5% the steadiest of the analysed algorithms. The mean range of *yar* (over all t) is with about

18.5% the largest. The other algorithms *tsvm*, *tknn* and *mc* achieved mean ranges of 16.2%, 17.0% (over all l), and 16.5%.

We applied paired one-sided Wilcoxon rank sum tests on the cross-validation accuracies (over all k) of every pair of algorithms on each dataset (Figure 7). The null hypothesis was that the first classifier has an equal or lower median cross-validation accuracy than the second. For each dataset, the tests were corrected for multiple-testing (Bonferroni $n = 132$). According to these tests, there was no algorithm which was significantly better than *tsvm*. The *tsvm* itself was significantly better than all other tested algorithms on four datasets (*AR*, *BI*, *SI*, *SH*). The *plc* was only outperformed by the *tsvm* on five datasets (*AR*, *BI*, *PA*, *SH*, *SI*). The *tknn* was not outperformed on the datasets *NU* and *PA*. *tknn* completely outperformed *yar* on the datasets *AR*, *BI*, *NU* and *WE*. The *mc* algorithm was outperformed by all algorithms on the datasets *BI*, *SH*, *WE* and *SI*.

5 Discussion

The major challenge in our settings is the low cardinality of the datasets ($\lesssim 100$) limiting the number of available labeled and unlabeled training samples. Although this is an unusual constraint for semi-supervised learning, some of the algorithms achieved good classification results in our experimental setting. The results on the standard cross-validation experiments were mostly better than those of the inverted ones. Coupled to a smaller number of labeled training samples, the results gained on the inverted cross-validation indicate that the lack of labeled training samples can often not be compensated by an increasing number of samples (which do not have a label). Nevertheless, and most interestingly the best cross-validation performance is not always achieved for completely labeled data, but rather for partially labeled datasets indicating the strong influence of label information on the classification process.

The lower performance of Yarowsky's algorithm and the mincut strategy can may be related to the small number of available samples. The iterative process of Yarowsky's algorithm is controlled by confidence predictions for the single data points. Related to the distance between the samples and the decision boundary these confidence predictions get less distinguishable and less informative in high dimensional settings.

The initial graph constructed by the mincut strategy was also effected by the small sample sizes. Here the majority of unlabeled test samples built separate subgraphs which were not connected to one of the label nodes. In this case the algorithm is not able to determine the class label of these samples.

The classifiers *tsvm* and *plc* showed better accuracies than a constant classifier throughout all experiments. The same holds true for *tknn* except for two single experiments. These three algorithms can therefore be used as early predictors. Finally, the *tsvm* achieved the best classification performance in our study followed by *plc* and *tknn*.

The accuracy of this algorithm is summarized in in Figure 6. Besides the overall trend of receiving higher accuracies for higher values of k an additional behavior of this algorithm can be seen. In some of the settings the best classification results is not directly achieved for $k = 10$. Better results can be found for slightly smaller values of k. We believe that this is not a direct effect of the tradeoff between labeled and unlabeled samples. Disturbances such as measurement and label noise seem to be related to this behavior with different severity. The benefit of this diminished label information even in the linearly separable case can serve as a starting point for future work.

Acknowledgments. This work was funded in part by a Karl-Steinbuch grant to FS, the German federal ministry of education and research (BMBF) within the framework of the program of medical genome research (PaCa-Net; project ID PKB-01GS08) and the framework GERONTOSYS (Forschungskern SyStaR, project ID 0315894A) to HAK. The responsibility for the content lies exclusively with the authors.

References

1. Armstrong, S.A., Staunton, J.E., Silverman, L.B., Pieters, R., den Boer, M.L., Minden, M.D., Sallan, S.E., Lander, E.S., Golub, T.R., Korsmeyer, S.J.: Mll translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nature Genetics 30(1), 41–47 (2002)
2. Atiya, A.F., Al-Ani, A.: A penalized likelihood based pattern classification algorithm. Pattern Recognition 42, 2684–2694 (2009)
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, Secaucus (2006)
4. Bittner, M., Meltzer, P., Chen, Y., Jiang, Y., Seftor, E., Hendrix, M., Radmacher, M., Simon, R., Yakhini, Z., Ben-Dor, A., Sampas, N., Dougherty, E., Wang, E., Marincola, F., Gooden, C., Lueders, J., Glatfelter, A., Pollock, P., Carpten, J., Gillanders, E., Leja, D., Dietrich, K., Beaudry, C., Berens, M., Alberts, D., Sondak, V.: Molecular classification of cutaneous malignant melanoma by gene expression profiling. Nature 406(6795), 536–540 (2000)
5. Blum, A., Chawla, S.: Learning from labeled and unlabeled data using graph mincuts. In: Brodley, C.E., Danyluk, A.P. (eds.) ICML 2001 Proceedings of the Eighteenth International Conference on Machine Learning, pp. 19–26. Morgan Kaufmann, San Francisco (2001)
6. Buchholz, M., Kestler, H.A., Bauer, A., Böck, W., Rau, B., Leder, G., Kratzer, W., Bommer, M., Scarpa, A., Schilling, M.K., Adler, G., Hoheisel, J.D., Gress, T.M.: Specialized DNA arrays for the differentiation of pancreatic tumors. Clinical Cancer Research 11(22), 8048–8054 (2005); HAK and MB contributed equally
7. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley, New York (2001)
8. Fix, E., Hodges Jr., J.L.: Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties. Technical Report Project 21-49-004, Report Number 4, USAF School of Aviation Medicine, Randolf Field, Texas (1951)
9. Hastie, T., Tibshirani, R., Friedman, J.H.: The Elements of Statistical Learning, corrected edn. Springer, Heidelberg (2003)
10. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machines. In: Bratko, I., Dzeroski, S. (eds.) Proceedings of ICML 1999, 16th International Conference on Machine Learning, pp. 200–209. Morgan Kaufmann Publishers, San Francisco (1999)
11. Nutt, C.L., Mani, D.R., Betensky, R.A., Tamayo, P., Cairncross, J.G., Ladd, C., Pohl, U., Hartmann, C., McLaughlin, M.E., Batchelor, T.T., Black, P.M., von Deimling, A., Pomeroy, S.L., Golub, T.R., Louis, D.N.: Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. Cancer Research 63(7), 1602–1607 (2003)
12. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In: Bartlett, P.J., Schölkopf, B., Schuurmans, D., Smola, A.J. (eds.) Advances in Large Margin Classifiers. MIT Press (2000)

13. Shipp, M.A., Ross, K.N., Tamayo, P., Weng, A.P., Kutok, J.L., Aguiar, R.C.T., Gaasenbeek, M., Angelo, M., Reich, M., Pinkus, G.S., Ray, T.S., Koval, M.A., Last, K.W., Norton, A., Lister, T.A., Mesirov, J., Neuberg, D.S., Lander, E.S., Aster, J.C., Golub, T.R.: Diffuse large b-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. Nature Medicine 8(1), 68–74 (2002)
14. Shu, L., Wu, J., Yu, L., Meng, W.: Kernel-Based Transductive Learning with Nearest Neighbors. In: Li, Q., Feng, L., Pei, J., Wang, S.X., Zhou, X., Zhu, Q.-M. (eds.) APWeb/WAIM 2009. LNCS, vol. 5446, pp. 345–356. Springer, Heidelberg (2009)
15. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., Lander, E.S., Loda, M., Kantoff, P.W., Golub, T.R., Sellers, W.R.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1(2), 203–209 (2002)
16. Vapnik, V.N.: Statistical Learning Theory. Wiley, New York (1998)
17. West, M., Blanchette, C., Dressman, H., Huang, E., Ishida, S., Spang, R., Zuzan, H., Olson, J.A., Marks, J.R., Nevins, J.R.: Predicting the clinical status of human breast cancer by using gene expression profiles. Proceedings of the National Academy of Science of the United States of America 98(20), 11462–11467 (2001)
18. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: Uszkoreit, H. (ed.) ACL 1995 Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, pp. 189–196. Association for Computational Linguistics, Stroudsburg (1995)