Semi-Supervised Self-Training with Decision Trees: An Empirical Study

Tanha, J.; van Someren, M.; Afsarmanesh, H.

Publication date
2011
Document Version
Author accepted manuscript
Published in
3rd IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS 2011)

Link to publication

Citation for published version (APA):
Tanha, J., van Someren, M., & Afsarmanesh, H. (2011). Semi-Supervised Self-Training with Decision Trees: An Empirical Study. In *3rd IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS 2011): Guangzhou, China* (pp. 699-704) https://www.academia.edu/7170741/Semi-Supervised_Self-Training_with_Decision_Trees_An_Empirical_Study

Semi-Supervised Self-Training with Decision Trees: An Empirical Study

Jafar Tanha, Maarten van Someren, and Hamideh Afsarmanesh
Computer science Department,University of Amsterdam, The Netherlands
J.Tanha,M.W.vanSomeren,h.afsarmanesh@uva.nl

***Abstract*—We consider semi-supervised learning, using a pool of unlabeled data to augment performance of a supervised learning algorithm. In particular, we consider semi-supervised learning with decision tree learners. Experiments show that standard decision tree learners do not perform well when used as "base" classifier in semi-supervised learning by self-training. We argue that this is because they provide poor probability estimates with their classifications. Decision tree as the "base" classifier in self-training faces two obstacles to producing a good ranking of instances: the first is that the sample size on the leaves is almost always small, and the second is that all instances at a leaf get the same probability. This leads to poor selection of new labeled data during self-training. In this paper, we study the effect of four improvements to the standard decision tree learners: Grafting, Reduced pruning, Naive Bayes Tree, and Laplacian Correction. Experiments show that these improvements are helpful in the selection step of self-training where the selection of most reliable predictions is done for next iteration by self-training.**

***Index Terms*—Semi-Supervised Learning, Self-training, Grafted Decision Tree, Probability Estimation, Naive Bayes Tree**

I. Introduction

Supervised learning methods are effective when there are sufficient labeled instances to construct classifiers. Labeled instances however are often difficult, expensive, or time consuming to obtain, because they require empirical research or experienced human annotators. Meanwhile in many practical domains, such as medical domains, speech recognition, web-page classification and text mining, there is a large supply of unlabeled instances. Semi-supervised learning methods use both labeled and unlabeled instances. Often semi-supervised learning achieves a better accuracy than supervised learning which is only trained on the labeled data.

There are several different kinds of semi-supervised learning methods, for example Expectation Maximization, graph-based, mixture models, self-training, and co-training methods. In this paper we focus on self-training as one of the widely used semi-supervised learning method in many domains. In self-training the learning process employs its own predictions to teach itself. An advantage of self-training is that it can easily be combined with any supervised learning algorithm as base learner [1]. The self-training procedure "wraps" around the "base" learner without changing its inner workings.

Here we consider decision tree classifiers as the "base" learner in self-training. Decision trees are found to be the best classifier in many diverse domains such as medical diagnosis or speech recognition [2], [3]. An additional benefit is the comprehensibility of the resulting trees. However, as we shall see, standard decision tree learning algorithms are not suitable for self-training, see the results in Figure 3. We suspect that the reason is that decision trees provide poor probability estimates [4]. Decision trees as the "base" classifier in self-training faces two obstacles to producing a good ranking of instances: the first is that the sample size on the leaves is almost always small, and the second is that all instances at a leaf get the same probability. Therefore, selecting newly-labeled data in self-training is error-prone.

We consider several solutions for the above problems: reduced pruning, grafting, smoothing by the Laplacian correction and using a Naive Bayes classifier at the leaf of a decision tree, the NBTree algorithm. Experiments show that the performance of self-training is improved by such measures.

The rest of this paper is organized as follows. Section II outlines the related work on semi-supervised learning. In section III, decision tree classifiers as the supervised learner in self-training are presented. In section IV we address the four improvements for self-training. In section V, the setting of our experiments on the UCI datasets are addressed. The results of supervised learning on datasets are addressed in section V-A. Finally, in section VI, we address our conclusions.

II. Related work

Often semi-supervised learning methods use a generative model for the classifier and employ Expectation Maximization (EM) [5] to estimate the labels, for examples by a mixture of Gaussian [6] and a mixture of experts [7]. Vapnik, [8] gives both theoretical and experimental evidence on Transductive Support Vector Machines (TSVM) as another useful method for semi-supervised learning. There are also graph-based models for semi-supervised data [9]. Unlike the other semi-supervised learning method, the self-training method can easily be used with any supervised learning algorithm [1]. Self-training, as a single-view semi-supervised learning method, has been widely used in diverse domains, such as, Natural Language Processing [10][11].

In [12] a semi-supervised approach to training object detection systems based on self-training shows that a model trained from a small number of labeled instances can achieve results comparable to a model trained in the supervised manner using a much larger set of labeled instances. A self-training semi-supervised support vector machine (SVM) algorithm proposed in [13] applies it to a dataset collected from a P300-based brain computer interface (BCI) speller. This significantly reduced training effort of the P300-based BCI speller.

In [11] a semi-supervised self-training approach using a hybrid of Naive Bayes and decision trees is used to classify sentences as subjective or objective. Provost and Domingos [4] presented a few techniques to modify the C4.5 decision tree learner for better probability estimation. First, they use Laplace correction at leaves, probability estimates are smoothed towards the prior probability distribution. Second, by turning off pruning in C4.5, decision trees generate larger trees to give more precise probability estimation. The resulting method is called C4.4.

Semi-supervised learning methods tend to find informative unlabeled instances such that with labeling them and adding to the original labeled data,it leads to improve the performance of the classifier. This point is indeed vital for self-training which only uses its own prediction for selecting the unlabeled instances. When the base learner of self-training are decision trees, the selection of unlabel data is more difficult, because in practice decision tree classifiers produce poor probability estimates. In this paper we propose the additional methods to improve the probability estimation and explore the effect in a wider range of domains.

III. Self-training with Decision tree learning

The best-known algorithm for building decision trees is C4.5 [14]. Decision Tree learning algorithms are among the most popular in practice and in many applications they are found to achieve the best accuracy, see for example [15]. A self-training algorithm uses its own predictions to obtain new labeled training data, see Figure 1. A “base” learner is first trained with a small number of labeled instances, the initial training set. The classifier is then used to predict the labels for the unlabeled instances (prediction step). In the next step, a subset S of the unlabeled instances, together with their predicted labels, is selected to augment the labeled instances (selection step). Typically, S consists of a few unlabeled instances with high confidence predictions. Bad selection in this step will reduce the performance. The classifier is then retrained on the new set of labeled instances, and the procedure is repeated (re-training step).

The selection function in Figure 1 tends to find a subset of the unlabeled instances based on confidence. The selected subset S of unlabeled instances includes the high-confidence predictions in each iteration of the training process. This is important because a misclassified prediction will propagate to further classification error. In each iteration the newly high-confidence labeled instances are added to the original labeled data. The number of iteration in Figure 1 depends on the threshold T. In the rest we study the selection strategy of self-training, particularly when the base learner is the decision tree.

```
Self-Training (L, U, F)
Input: L,U are labeled and unlabeled data;
       F is underlying classifier;
       T is the threshold for selection;
Initial:
       T=C //Threshold for confidence;
       L =Labeled data,U=Unlabeled data;
       While((U!=empty)or(maxIterations))
          Train F on L;
          S =S(U,T,F)//Selection function
            Where S is a set of the
            High-confidence Predictions;
          U = U - S; L = L U S;
Output: L //Original Labeled data+
          //Newly-Labeled Instances
```

Fig. 1: The self-training (ST) Algorithm

IV. Improving Self-Training by improving probability estimates

Our hypothesis is that decision tree learners do not provide good indications of the confidence in their classifications. The distribution at the leave of a decision tree gives the probability that the instance belongs to the majority class but these probabilities are based on very few data, due to the fragmentation of data over the decision tree. In semi-supervised learning this is aggravated by the fact that the sample size of initial training set is small from the beginning, which is characteristic for the semi-supervised learning tasks.

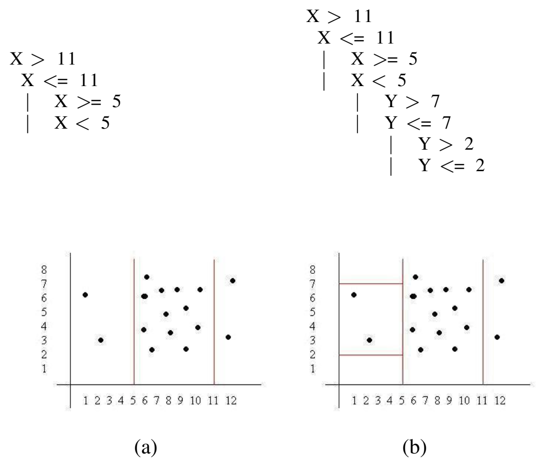
When a decision tree uses a numerical attribute then it splits the range of values. All values in an interval have the same probability of belonging to a class. For many domains this is not optimal. In most domains examples close to the boundary have a higher probability of belonging to a different class than examples that are in the middle of an interval. Also, some leaves may have too few instances to estimate the confidence and other data should be used. In this paper we consider four methods for improving the probability estimates at the leaves of decision trees: Naive Bayes Tree, Grafted Decision Tree, Laplacian Correction and Reduced Pruning.

A. NBTree

The naive Bayesian tree learner, NBTree [16], combines naive Bayesian classification and decision tree learning. In an NBTree, a local Naive Bayes Classifier is constructed at each leaf of decision tree that is built by a standard decision tree learning algorithm like C4.5. NBTree achieves in some domains higher accuracy than either a Naive Bayes Classifier or a decision tree learner. NBTree uses additional attributes and gives a posterior probability distribution that can be used to estimate the confidence of the classifier.

B. Grafted Decision Tree

A “grafted decision tree classifier” generates a grafted decision tree from a standard tree. The grafting technique [17] searches for regions of the multidimensional space of attributes that are labeled by the decision tree but that contain no or very sparse training data. These regions are then split by splitting the region that corresponds to a leaf and labeling the empty or sparse areas by the label of the majority above the



```
X > 11
 X <= 11
  |   X >= 5
  |   X < 5
```

```
X > 11
  X <= 11
  |   X >= 5
  |   X < 5
        |   Y > 7
        |   Y <= 7
             |   Y > 2
             |   Y <= 2
```

Fig. 2: Grafted Decision tree

previous leaf node. Consider the example in Figure 2. Figure 2(a) shows the resulting tree. As it can be seen, there are two cuts in the decision trees at nodes 11 and 5. After grafting, the branches have increased due to the grafting technique. Figure 2(b) shows the results of grafting and gives the resulting tree. Grafting performs a kind of local "unpruning" for low-density areas. This can improve the resulting model (see [17]). Grafted decision tree implies that grafting gives better decision trees in case of sparse data.

Inspired from [4] that uses Laplacian Correction and No-Pruning in C4.5 for improving probability estimates of decision tree we employ these two improvements in grafted decision tree. These two improvements affect on both the classification accuracy and probability estimates of grafted decision tree. We call the resulting tree C4.4graft. In fact C4.4graft gives better decision tree in case of sparse data and it also improves probability estimates because of using Laplacian correction and No-Pruning. Our experimental results verify these improvements as well, see Figure 5.

C. Laplacian Correction

The Laplacian correction (or Laplace estimator) is a way of smoothing probability values. In fact smoothing of probability estimates from small samples is a well-studied statistical problem [18]. Assume there are K instances of a class out of N instances at a leaf, and C classes. The Laplacian correction calculates the estimated probability P(class) as (K+1)/(N+C). Therefore, while the frequency estimate yields a probability of 1.0 from K=10, N=10 leaf, for a binary classification problem the Laplace estimate produces a probability of (10+1)/(10+2)=0.92. For sparse data, the Laplacian correction in the leaves of a tree yields a more reliable estimation that is crucial for the selection step in self-training.

D. Reduced Pruning

We include an additional methods in our experiments: a decision tree learner that does not do any pruning. Although this indeed introduces the risk of "overfitting", it may be a useful method because of the small amount of training data. In this case, pruning methods can easily produce *under*fitting and Reduced Pruning avoids this.

V. Experiments with UCI datasets

Eight UCI datasets [19] are used in our experiments. We selected these because:(i) they involve binary classification and (ii) these are used on several other studies in semi-supervised learning [20]. Information about these datasets is in Table I. All sets have two classes and Perc. represents the percentage of the largest class.

Dataset	**Attributes**	**Size**	**Perc.**
Bupa	6	345	58
Colic	22	368	63
Diabetes	6	768	65
Heart	13	270	55
Hepatitis	19	155	21
Ionosphere	34	351	36
Tic-tac-toe	9	958	65
Vote	16	435	61

TABLE I: Overview of Datasets

For each dataset, about 30 percent of the data are kept as test set, and the rest are used as the pool of training instances with a small amount of labeled and the rest unlabeled data. In this paper we only address the results of the experiments that include only 10% labeled data. We use decision tree classifies as "base" classifier in self-training, which is an inductive semi-supervised learning [1]. J48 (which is Java implementation of C4.5 in WEKA), C4.4, NBTree, C4.4graft, and J48graf are used as the "base" learners in self-training. For our experiments we use the WEKA tool [21] in Java.

A. Decision tree learning algorithms

First we assess whether decision trees are good classifiers for these domains or not. Decision tree learning gives the highest accuracy in five of the eight domains. We include also the domains for which decision tree learning does not give the best results to see the effect of semi-supervised learning. Our approach is to find solution that self-training benefits from decision tree classifiers, but based on our experiment it does not work well with C4.5 decision tree. we propose four improvements for solving the problem.

B. Self-training with C4.5 decision tree learner

In the first experiment, we use the J48 decision tree learner as "base" learner and compare this with running it only on the labeled data. Figure 3 shows the classification accuracy of decision tree learning (DT) and decision tree as "base" learner in self-training (ST-DT) respectively. As can be found, there is basically no improvement for self-training. The average improvement over these eight domains is very small, 0.2%.

Based on the self-training process, in each iteration, the selection procedure have to find the high-confidence predictions, but in practice it cannot recognize the correct predictions, because decision tree provides poor probability estimates. As mentioned earlier, two main problems leads to having poor

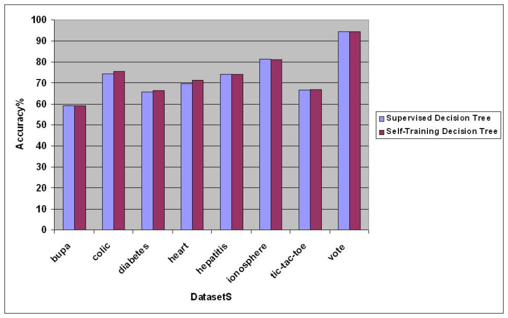


Fig. 3: Self-training with basic decision tree learner

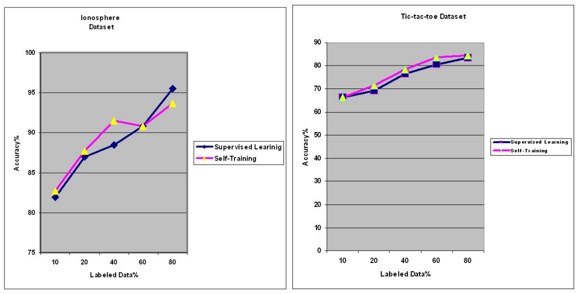


Fig. 4: Performance of No-Pruning Self-Training with increasing proportions of labeled data on Ionosphere and Tic-tac-toe datasets

probability estimates in decision tree learners: the first is that the sample size on the leaves is almost always small, and the second is that all instances at a leaf get the same probability. Therefore, finding S in the selection step of self-training regarding the above two problems is challenging and this may well restrict the accuracy of self-training with C4.5 decision tree. In the following sections we introduce some improvements.

C. Self-Training with Laplacian, Grafting, NBTree, and Reduced Pruning

In this section our goal is to find method that can solve two problems, which mentioned earlier. For achieving this, set of experiments are performed on the self-training with different settings.

Dataset	GDT	ST-GDT	DTL	ST-DTL
Bupa	56.7	57	56.9	70.9
Colic	74.7	77.8	78.5	79.7
Diabetes	66.5	70.2	68.1	69.0
Heart	70.2	72.3	56.7	70.9
Hepatitis	74.6	79.8	71.2	71.6
Ionosphere	81.9	82.8	80.0	82.1
Tic-tac-toe	66.5	69.4	67.1	69.1
Vote	92.9	94.3	94.7	95.2

TABLE II: Performance of Self-Training with Grafting, Laplacian and both

Dataset	DT-NP	ST-NP	NBTree	ST-NBTree
Bupa	59.2	59.3	58.7.	59.2
Colic	74.4	76.1	69.9	72.3
Diabetes	65.7	67.6	70.4	72.0
Heart	69.8	70.8	71.7	75.7
Hepatitis	74.6	70.8	79.3	82.5
Ionosphere	82	83.1	83.1	86.8
Tic-tac-toe	66.2	66.3	64.4	68.3
Vote	92.9	94.3	89.2	90.2

TABLE III: Performance of Self-Training with No-Pruning and with NBTree

The results show that both Grafting and the Laplace correction enable the decision tree learner to benefit from unlabeled data. Table II shows the performance of grafted decision tree (GDT), self-training grafted decision tree (ST-GDT), decision tree with Lapacaian (DTL) correction, and self-training decision tree with Laplacian (ST-DTL) correction . Note that in each experiment we use supervised learning methods on original labeled data and then apply self-training with the base learners on both labeled and unlabeled data.The results are promising, because the grafting leads to good decision trees.

Another setting for self-training with decision tree is to use No-Pruning tree and to employ the NBTree. Table III depicts the performance of decision tree with No-Pruning (DT-NP), self-training with No-Pruning (ST-NP) decision tree, NBtree and self-training with NBTree (ST-NBTree). The experiment shows that No-Pruning gives a small improvement in accuracy, but the Naive Bayes Tree (NBTree) learner can improve the performance of self-training well and also achieves the highest accuracy for four of the eight domains. For the $Ionosphere$ dataset the self-training with Naive Bayes tree learner reaches the highest accuracy.

NBTree is a different approach to estimating the probability distribution at the node, using all remaining attributes. As a result, it is useful for self-training in the selection step, which is aimed to find a set of high-confidence predictions. Our results verify this approach as well.

Consistent with our hypothesis we see a small but consistent improvement for the self-training versions for decision tree learners with Laplace correction and with grafting. The reason for why switching off pruning improves the effect of self-training is that the decision tree learner indeed seems to “underfit” the data which prevents it from giving good predictions on the unlabeled data because of small size of the tree. As it can be seen in table III, No-Pruning helps self-training performance. To see if the effect of “No-Pruning” is weakened with increasing numbers of labeled data, we repeat the experiments with larger proportions of labeled data, see Figure 4.

D. Combining improvements

As an important question for our experiment is: do the effects of grafting, Laplacian correction, No-Pruning, and NBTree add up or not? To answer this question we ran another series of experiments in which all improvements are combined together. The results depict that the new settings improve the performance of self-training much more than the others. We suspect that this is because the effects are not the same

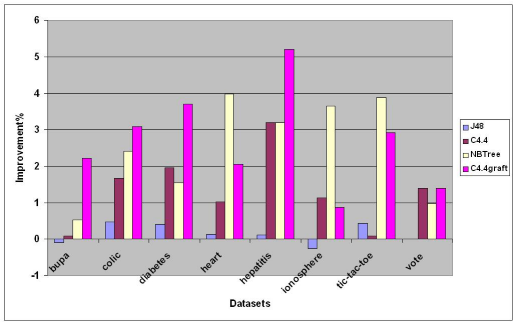


Fig. 5: Improvements of self-training in terms of datasets and different settings

and combining them gives the effect of improving on both probability estimates and classification accuracy. Both grafting and the Laplacian correction improve the decision trees and the ranking of predictions for self-training. Combining both gives an even better result because the Lapacian correction improves the probability estimates in the grafted leaf nodes. The same applies to the combination of No-pruning and Laplacian correction.

Dataset	**DT**	**C4.4**	**NBTree**	C4.4Graft
Bupa	59.1(±1.8)	59.2(±2.2)	59.2(±0.8)	61.4(±1.1)
Colic	75.6(±0.7)	76.8(±1.9)	72.3(±0.5)	77.7(±0.9)
Diabetes	66.4(±2.5)	67.6(±1.9)	71.9(±1.5)	70.9(±2.6)
Heart Statlog	71.4(±1.3)	70.8(±2.7)	75.7(±0.6)	72.3(±1.2)
Hepatitis	74.1(±1.5)	77.8(±1.9)	82.5(±1.5)	79.8(±0.8)
Ionosphere	81.2(±2.6)	83.1(±3.92)	86.8(±1.6)	82.8(±1.8)
Tic-tac-toe	67.1(±1.8)	66.2(±2.3)	68.3(±0.6)	69.3(±0.9)
Vote	94.3(±1.9)	94.3(±3.2)	90.2(±1.7)	94.3(±3.1)

TABLE IV: Average Classification Accuracy and Standard Deviation of combined effect of self-training with No-Pruning, Laplacian correction, NBTree, and grafting

The average improvement (over all datasets) for C4.4, NBTree, and C4.4graft are 1.32%, 2.53%, and 2.68% respectively. Table IV shows the performance of self-training with J48 (standard DT), C4.4, NBTree, and C4.4graft. Figure 5 shows the improvements in terms of datasets and the proposed methods separately.

VI. Conclusion

Although decision tree learning is the best method for many domains, we observed that self-training with a standard decision tree learner does not work well. We find that variations that are aimed at improving estimates of the posterior probabilities of classifications of the unlabeled data give better results. Decision Tree learners normally maximize the accuracy but not the margin. They do not try to optimize estimates of posterior probabilities. In self-training these probabilities are used to select unlabeled data with their prediction that should improve the performance of current decision tree. It is therefore important that the probability estimates are as precise as possible given the sparseness of data. Modifications to the decision tree learner aimed at improving these estimates do result in better estimates and this in turn gives better self-training performance.

Our experimental results showed that four modifications effectively improve the performance of self-training, even when there is only limited labeled instances. The best result based on experiments with a small amount of labeled instances (10%), which is the most relevant for semi-supervised settings, was obtained by a combination of grafting, no pruning and Laplacian correction, called C4.4graft. As it can be seen in the experiments, better probability-based ranking selects the high-confidence predictions in the selection step of self-training. Therefore, these variations improve the performance of self-training.

References

[1] X. Zhu and A. B. Goldberg, *Introduction to Semi-Supervised Learning*, ser. Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.

[2] M. Pal and P. M. Mather, "An assessment of the effectiveness of decision tree methods for land cover classification," *Remote Sensing of Environment*, vol. 86, no. 4, pp. 554 – 565, 2003.

[3] T.-S. Lim, W.-Y. Loh, and Y.-S. Shih, "A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms," *Machine Learning*, vol. 40, pp. 203–228, 2000.

[4] F. J. Provost and P. Domingos, "Tree induction for probability-based ranking," *Machine Learning*, vol. 52, no. 3, pp. 199–215, 2003.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. pp. 1–38, 1977.

[6] B. Shahshahani and D. Landgrebe, "The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon," *IEEE Transactions on*, vol. 32, no. 5, pp. 1087 –1095, Sep. 1994.

[7] D. Miller and H. Uyar, "A mixture of experts classifier with learning based on both labelled and unlabelled data," in *Advances in NLP*, M. Mozer, M. Jordan, and T. Petsche, Eds. Boston: MIT Press, 1997, pp. 57–?577.

[8] V. Vapnik, *Statistical learning theory*. Berlin: Springer, 1998.

[9] X. Zhu, "Semi-Supervised Learning Literature Survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[10] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *ACL*, 1995, pp. 189–196.

[11] B. Wang, B. Spencer, C. X. Ling, and H. Zhang, "Semi-supervised self-training for sentence subjectivity classification," in *Proceedings of 21st conference on Advances in artificial intelligence*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 344–355.

[12] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models," in *WACV/MOTION*. IEEE Computer Society, 2005, pp. 29–36.

[13] Y. Li, C. Guan, H. Li, and Z. Chin, "A self-training semi-supervised svm algorithm and its application in an eeg-based brain computer interface speller system," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1285 – 1294, 2008.

[14] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[15] M. van Someren and T. Urbancic, "Applications of machine learning: matching problems to tasks and methods," *Knowledge Eng. Review*, vol. 20, no. 4, pp. 363–402, 2005.

[16] R. Kohavi, "Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid," in *KDD*, 1996, pp. 202–207.

[17] G. I. Webb, "Decision tree grafting from the all tests but one partition," in *IJCAI*, T. Dean, Ed. Morgan Kaufmann, 1999, pp. 702–707.

[18] J. Simonoff, *Smoothing Methods in Statistics*. New York: Springer, 1996.

[19] A. Frank and A. Asuncion, "UCI machine learning repository," 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[20] Z.-H. Zhou and M. Li, "Tri-training: exploiting unlabeled data using three classifiers," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 11, pp. 1529 – 1541, 2005.

[21] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, November 2009.