Agent Teaching Agent Framework

Parijat Prosun Kar
Department of Mathematical & Computer Sciences
The University of Tulsa
karpa@ens.utulsa.edu

Sandip Sen
Department of Mathematical & Computer Sciences
The University of Tulsa
sandip@kolkata.mcs.utulsa.edu

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence; I.2.8 [**Artificial Intelligence**]: Learning—*concept learning*

General Terms

Algorithms, Experimentation, Performance

Keywords

Agents, training

1. INTRODUCTION

We address the problem of knowledge transfer between a trainer and a trainee agent. The knowledge being transferred is a concept description, a boolean-valued function that classifies input examples as members or non-members of the target concept. We assume that the trainer agent does not have access to the internal knowledge representation of the trainee agent, but can evaluate its concept recognition abilities by asking it to categorize selected exemplars and non-exemplars of the target concept. The trainer agent also do not have access to the original training set from which it learned its concept description, and hence has to rely only on that learned knowledge to train the trainee agent. We present an Agent Teaching Agent (ATA) framework which focuses on incremental selection of training examples by the trainer to expedite the learning of the trainee agent.

We envisage an iterative training procedure in which alternatively the trainer selects a set of training and testing exemplars, the trainee trains using the training set and then classifies the testing set, the trainer observes errors made by the trainee in classifying the instances in the last testing set and accordingly generates the next training and testing sets. This iterative process converges when error of the trainee falls below a given threshold. We now present these iterative training steps in an algorithmic form:

AAMAS'03, July 14–18, 2003, Melbourne, Australia.
ACM 1-58113-683-8/03/0007.

Procedure Train-Agent(Trainer,Trainee,Domain-Info){
Select initial training set N_0 and initial testing set T_0 from Domain-Info
$i \leftarrow 0$
repeat
  Train trainee agent on training set N_i
  Let M_i be the instances in T_i misclassified by trainee after training on N_i
  $T_{i+1} \leftarrow T_i \cup newTestInstances(M_i)$ and Domain-Info
  $N_{i+1} \leftarrow N_i \cup newTrainingInstaces(M_i)$
  $i \leftarrow i + 1$
until $|M_i| < threshold$
}

The above procedure needs to be further fleshed out to realize an actual implementation. In particular, we have to specify procedures for selection of the initial training and testing sets, N_0 and T_0, and the generation of the next test set T_{i+1} based on the mistakes, M_i, made by the trainee on the current test set. When selecting the initial training and testing instances, the goal is to select the most discriminating examples that help identify regions of the input space that do and do not belong to the target concept. For example, if a hyperplane separates instances of the target concept from non-instances, then points close to and on both sides of that hyperplane should be selected as initial training and testing set members. When selecting the next set of training and testing instances, the goal is to first isolate the mistakes made on the previous test set, and for each of these instances, find a few neighboring points, use some of them as part of the training data and the rest as part of the test data in the following iteration. Note that the true classification of these points will not be known in general, and only their estimated classification, based on the concept description knowledge previously acquired by the trainer, can be used.

The actual procedure for selecting the incremental training and testing sets depend on the internal representation used by the trainer agent. We have developed these procedures for instance based and decision tree learners to work on problems with real-valued attributes. More details about the architecture and the algorithms can be found at `http://www.mcs.utulsa.edu/~karpa/AAMAS03full.pdf`.

2. RESULTS

In this paper, we report on results from experiments using an instance-based learning algorithm, IB2 [1], and a decision-tree based learning algorithm, C4.5 [2], as trainer and trainee agents. For initial development and evaluation

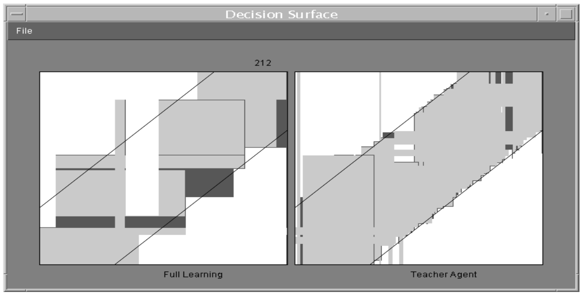

Figure 1: Decision surfaces produced by batch and incremental training of C4.5 by IB2 on 2/1 dataset.

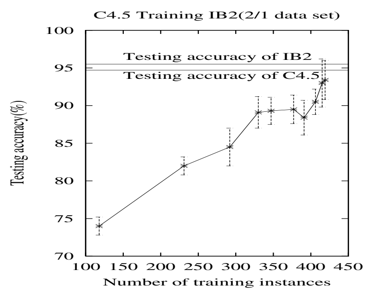

Figure 2: Performance of IB2 in the 2/1 problem with increasing training instances given by C4.5.

we used a set of artificial concept descriptions. All of the artificial problems we have experimented with are defined on 2 continuous attributes, where the domain of each attribute is $[0, 1]$. Due to space limitations, here we discuss results from only one of the artificial data sets, the 2/1 problem, and one real-life data set from the UCI machine learning repository (`http://www1.ics.uci.edu/~mlearn/MLSummary.html`).

We compare the effectiveness of this incremental training framework with the baseline case of the trainee having access to the entire original training set used by the trainer for its learning. Figure 1 presents results when IB2 is the trainer and C4.5 is the trainee on the 2/1 data set. The figure on the left shows the concept description (shown as the white region) formed when all points stored by IB2 is given to C4.5 in one shot. The right one shows the decision surface after 14 iterations in the iterative method. The straight lines correspond to the actual concept boundaries: the region in between the two lines is not part of the target concept. From these two figures, it is clear that the incremental method produces a better fit of the concept learned by the trainee to the true concept, and hence produces greater test-set classification accuracy.

We next used C4.5 as the trainer agent and IB2 as the trainee agent and the corresponding results are presented in Figure 2. When C4.5 is trained with 800 training instances it learns 27-31 rules for different training sets of 5-fold cross validation. While using C4.5 as the trainer, the initial training set for training IB2 (we assume that the original training set is no longer available to the teacher) is generated by selecting points on all sides of the boundaries at each vertex of every hyperrectangle corresponding to leaf nodes of the decision tree. The initial training set for the 2/1 problem typically contains about 115-125 examples over different runs.

C4.5 (full training set, 245 instances)	68.72%
IB2 (full training set, 245 instances)	56.08%
C4.5 (incremental training, 65 instances)	63.5%
IB2 (incremental training, 154 instances)	63.52%

Table 1: Average test set performance on Haberman's Survival Data Set.

In all of the 5-fold cross validations, we got a testing accuracy of about 75% after the first training iteration. After 12-17 iterations final testing accuracy in the range of 92.5% to 94.5% is reached. The final training set size varies in the range of 415 to 460 whereas IB2 has an accuracy of 94% when trained with the original 800 training examples.

As in the case of IB2 teaching C4.5, here also we find that incremental training results in rapid improvement in classification performance of the trainee agent over the entire test set. The final accuracy is comparable to the accuracy of the trainer's knowledge or the trainee's knowledge if it had access to the original training set. Also the incremental version requires much less training examples, between 240 and 275 instances, than the entire training set size of 800 points.

Particularly interesting results (see Table 1) were obtained with the Haberman data set obtained from the UCI repository: IB2 acting as a trainer can train C4.5 to have better testing accuracy than itself. The trainer can, via the iterative training process, produce a more competent trainee! The final concept identification capability appears to depend on the learning biases of the trainee more than the learning bias of the trainer.

3. RELATED WORK AND CONCLUSIONS

Related work on active learning methods allow learners to query omniscient trainers. For example, pool based active learning has been used in support-vector machines (SVMs) instead of randomly selected training set [3].

We are currently running experiments on a wider set of problem instances that contain both artificial and real-life data and plan to include the results in the next update of this paper. We plan to use additional learning algorithms, e.g., SVMs, neural nets, genetic algorithm based classifiers, etc. as trainer and trainee agents. We also plan to explore possible combinations of active learning and the ATA approach.

Acknowledgments: This work has been supported in part by an NSF CAREER award IIS-9702672.

4. REFERENCES

[1] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1), 1991.

[2] Ross J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, California, 1993.

[3] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. In *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 999–1006. Morgan Kaufmann, 2000.