Adapting in Agent-Based Markets: A Study from TAC SCM

David Pardoe and Peter Stone
Department of Computer Sciences
The University of Texas at Austin
{dpardoe, pstone}@cs.utexas.edu

ABSTRACT

An agent attempting to model market conditions may benefit from considering how various combinations of competitor strategies would impact these conditions. We give an illustration using a prediction task faced by our agent for the Supply Chain Management scenario of the Trading Agent Competition (TAC SCM). We present the learning approach taken, evaluate its effectiveness, and then explore methods of improving predictions through combining multiple sources of data reflecting various combinations of competitor behaviors.

Categories and Subject Descriptors

I.2.6 [**Computing Methods**]: Artificial Intelligence—*Learning*

General Terms

Algorithms, Experimentation, Economics

Keywords

machine learning, adaptation, trading agents, markets

1. INTRODUCTION

In agent-based markets, adapting to the behavior of other agents is often necessary for success. Sometimes, however, it may not be possible to directly model, or even observe, the behavior of competitors. In such cases, agents may instead model and adapt to the market conditions that result from competitor behavior. An agent taking this approach can still benefit from reasoning about possible competitor strategies. In choosing the methods of adaptation it will use, an agent should consider how various combinations of competitor strategies would impact the market conditions being modeled and how effective the adaptation would be under different possible scenarios.

In this paper, we present an application of such an approach in the Supply Chain Management scenario of the Trading Agent Competition (TAC SCM). We describe a specific prediction problem faced by TacTex-06 (winner of the 2006 competition), present the learning approach taken, and evaluate the effectiveness of this approach through analysis of the competition results. We then explore methods of improving predictions through combining multiple sources of data reflecting various competitor behaviors.

2. PRICE PREDICTION IN TAC SCM

TAC SCM [1] provides a unique testbed for studying and prototyping automated supply chain management agents by providing a competitive environment in which independently created agents can be tested against each other over the course of many simulations in an open academic setting. In a TAC SCM game, six agents act as computer manufacturers in a simulated economy managed by a game server. Each agent must negotiate with suppliers to purchase components, manage a factory where computers are assembled, and sell the completed computers to customers through a series of first-price procurement auctions.

The focus of this paper is the method of predicting changes in computer prices used by TacTex-06. In order to maximize revenue from the computers sold, TacTex-06 needs to consider not only the prices it will offer on the current day, but also what computers it will wish to sell on future days. Computer prices often change significantly over a short period of time, and an agent that can predict changes in prices can increase its profits by attempting to sell when prices are highest. In the this paper, we describe the use of machine learning methods to predict the amount by which the average sales price of each type of computer will change in ten days, the period for which TacTex-06 typically plans ahead.

Making accurate predictions may depend on adapting to the behavior of the five competing agents. The structure of the TAC SCM competition encourages such adaptation: each round consists of a number of games against the same opponents, and after each game a log is provided that details the complete events of the game. Our approach to learning is thus to generate training data from the available game logs. Each training instance consists of 31 features representing data available to the agent during the game, such as the date, estimated levels of customer demand, and current and recent prices of a given type of computer. The label for each instance is the amount by which the average price of that computer changes in ten days. The question addressed in the rest of the paper is how to best make use of all available data when generating predictors. In the next section, we explain how this question was answered for the 2006 competition.

3. THE 2006 TAC SCM COMPETITION

We now address how TacTex-06 performed prediction in the 2006 competition. First we describe the choice of opposing agents used in simulations and of a learning approach, and then we present the results of the final round of compe-

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

978-81-904262-7-5 (RPS) ©2007 IFAAMAS

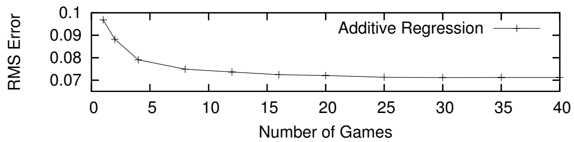

Figure 1: Results for Group 2

tition and additional experiments.

3.1 Agent Implementations

In order to develop a strategy for learning to make predictions, we ran a number of games using a variety of competing agents taken from the TAC Agent Repository,[1] a collection of agent binaries provided by the teams involved in the competition. At the time we designed our agent, only agents from the 2005 competition were available; however, in the experiments of this section, we make use of additional agents that have become available since then.

We chose four different agent groupings, and ran 50 games with each group. The first three groups each contain TacTex-06 and five unique additional agents. The fourth group includes what appear to be the strongest agents from the first three groups: TacTex-06, the 2005 version of TacTex, and the four other agents from the 2006 final round for which binaries are available. We included TacTex-06 in each group because we are only interested in making predictions for games in which our agent plays, and we therefore would like to capture the effect of TacTex-06 on the economy in the predictive models learned.

3.2 Learning Algorithm

After experimenting with algorithms from the WEKA machine learning package [2], we determined that the most effective was additive regression with decision stumps (an iterative method in which a decision stump is repeatedly fit to the residual from the previous step). A learning curve for Group 2 is shown in Figure 1, and is representative of the results for the other groups. In this and all other experiments in this paper except those involving data from the actual competition (for which a limited number of games are available), results are presented for four runs of five-fold cross validation. Root mean squared error is used as the measure of accuracy, and the values reported are fractions of the *base price* (a reference price based on maximum component costs) for each computer.

3.3 Results for Different Groups of Agents

From Figure 1, it appears that about 30 games are needed for training before prediction error reaches its minimum level, and about eight games before the error comes somewhat close to this level. Since a typical round of the TAC SCM competition involves 16 games, these results are somewhat concerning, as it might not be possible to learn sufficiently accurate predictors in time for them to be useful if only data from the current round is used.

We now consider the possibility of training predictors on games involving a different group (or groups) of agents. For each group of agents, each predictor trained was evaluated on test data from each of the four groups. In addition, for each group a predictor was trained on all games from the other three groups combined and evaluated for each fold of

	Test Data			
Model	*1*	*2*	*3*	*4*
heuristic	0.1173	0.1220	0.1074	0.1107
1	*0.0606*	0.0740	0.0657	0.0647
2	0.0636	*0.0711*	0.0676	0.0656
3	0.0641	0.0763	*0.0615*	0.0634
4	0.0640	0.0766	0.0637	*0.0597*
other 3	0.0620	0.0743	0.0641	0.0632

Table 1: RMS error when predictive models are learned using games from one group and tested on games from another group

that group. Table 1 shows the average results of evaluating each model on each group. For reference, we also determined the results of using a heuristic that performs linear regression on what TacTex-06 believes to be the average price of each computer over the past 10 days and predicts that the observed trend will continue.

The most important observation from these results is that while the predictive models that give the best results for each group are those trained on that group (statistically significant in each case with 99% confidence according to paired t-tests), the difference is fairly small. Also, for each group the predictor trained on all games from the other three groups does about as well as the best of the three predictors trained on only one of these groups, if not better, suggesting that training a predictor on games from all available groups is an effective strategy when it is not known which group will give the best results. In fact, after making this observation during our experimentation prior to the competition, we chose to use this strategy to learn the predictor that TacTex-06 used throughout the competition. We learned a single predictor before the start of the competition and did not adapt this predictor during the competition. The predictor was trained on all games that we ran between different groups of agent binaries available at the start of the 2006 competition.

3.4 Agent Performance

TaxTex-06 was the winner of the final round of the 2006 TAC SCM competition. Although it is difficult to assign credit for an agent's performance to particular components, an analysis of the game logs shows that TacTex-06 generally sold computers at higher prices than other agents, which would suggest that the attempt to predict changes in computer prices paid off. In fact, during the first third of each game, TacTex-06 had a higher average sales price than any opponent for every type of computer.

Figure 2 shows a comparison between the results of using a fixed predictive model (here we used the model from Section 3.3 that was trained on all games from Groups 1, 2, and 3, as Group 4 is very similar to the actual agents competing in the finals) and the results that would have been obtained by learning only from completed games. The results show that the fixed predictor performed as well as or better than the alternative for at least the first 8 games, and somewhat worse afterwards.

In order to better measure the effect of learning to predict changes in computer prices on the performance of TacTex-06, we performed two additional experiments using variations of TacTex-06 in which this ability was weakened or removed. In each experiment, 30 games were run using the agents of Group 4, except that TacTex-05 was replaced with an altered version of TacTex-06. In Experiment 1, the al-

[1] `http://www.sics.se/tac/showagents.php`

Exp. #	*Description*	*Score*	*Revenue*
1	no price change prediction	-4.27M	-3.05M
2	heuristic price change prediction	-1.79M	-1.21M

Table 2: Experiments comparing the performance of one altered version of TacTex-06 and one unaltered version. Numbers represent the difference between the altered version and the unaltered version.

tered version predicted no changes in computer prices, and in Experiment 2, the altered version used the heuristic from Section 3.2 in place of the learned predictor. Table 2 shows the differences between the scores and revenues of the normal and altered versions. Differences are statistically significant with 99% confidence according to paired t-tests. These results suggest that learning to predict the changes in computer prices had a significant impact on the performance of TacTex-06 in the 2006 competition.

4. ADDITIONAL APPROACHES

In the previous section, we chose between using a fixed predictor trained on a variety of games from our own simulations and the alternative of learning a predictor using only the games from the current round of competition. In this section, we explore the use of more sophisticated learning approaches that make use of both sources of data.

One way to make use of all available game data is to train on some combination of data from the current round (which we will call "new data") and other sources (which we will call "old data" and could include games from past rounds or the simulated competition of the previous section). The primary difficulty with this approach is deciding what the ratio of new data to old data should be. As more games are played, it likely makes sense to decrease the weight of the old data until at some point only new data is used.

We address this issue by using leave-one-out cross validation to choose the fraction of old data to be added to the complete set of new data. To test a particular choice of fraction when N games are available from the current round, we use each game once as the testing set while training a predictor on the combination of that fraction of old data and the remaining $N-1$ games. The fraction that produces the highest average accuracy over all N trials is then chosen, and a the predictor to be used is trained on all N games plus that fraction of the old data.

In the experiments of this section, we apply this approach of mixing data to the 2006 final round using all games from Groups 1, 2, and 3 of the previous section as the old data. To choose the fraction of old data to use at each step, we test each of 0, 1, 2, 3, 4, and 5 percent as described and choose the best. As the old data consists of 150 games, each percent is 1.5 games worth of data. Results are shown in Figure 2. The fraction of old data determined to be best decreased from 5% when two games were available to 1% when 15 games were available.

Instead of combining the old and new data, another possible approach is to combine the predictors themselves into an ensemble. We present here a method that is somewhat analogous to the data combination approach – instead of finding weights for the old and new data, we can find weights to be used in combining an "old predictor" and a "new predictor" through weighted averaging of their predictions. Given two predictors and a set of training data, we determine the weights of each predictor by evaluating both predictors on

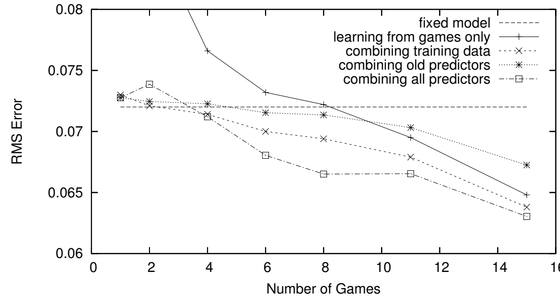

Figure 2: Predictor accuracy

each training instance and performing linear regression to find the weights that best combine these outputs to match the correct labels.

We are now left with the question of which predictor to use as the old predictor. Rather than using a single predictor, we will in fact use all of them: the predictors trained on each of the three groups alone along with the predictor trained on all three. The regression step described above can be performed using any number of predictors, and so we choose to perform linear regression on five variables: a weight for each of the four old predictors and a weight for the new predictor. For comparison, we also present the results of performing regression using only the four old predictors without learning a new predictor. The results of both approaches are shown in Figure 2.

We can see from the results that none of the approaches described in this section significantly outperform the fixed model for the first four games, but that both the method of combining data and the method of combining new and old predictors outperform the fixed and learned predictors when six or more games are available for training. The method of combining new and old predictors results in the lowest error, and this result is statistically significant with at least 95% confidence after at least six games have been played.

5. CONCLUSIONS AND FUTURE WORK

In this paper we described a number of approaches to learning to predict computer sales prices in the TAC SCM domain. There are many ways in which this work could be extended. The effects of a wider variety of opponent behavior could be explored by designing our own agents to behave in particular ways. Many ensemble methods other than weighted averaging of predictors could be tried. It is not clear how adaptation would be affected if other agents are themselves adapting.

Acknowledgments

This research was supported in part by NSF CAREER award IIS-0237699 and NSF grant EIA-0303609.

6. REFERENCES

[1] J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne, and S. Janson. The supply chain management game for the 2006 trading agent competition. Technical report, 2005. Available from http://www.sics.se/tac/tac06scmspec_v16.pdf.

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