In: *Proc. 2nd Joint Symposium on Neural Computation, University of California, San Diego and California Institute of Technology*, June 15, 1995. Institute for Neural Computation, University of California, San Diego, U.S.A.

Using Temporal-Difference Reinforcement Learning to Improve Decision-Theoretic Utilities for Diagnosis

Magnus Stensmo **Terrence J. Sejnowski**

Computational Neurobiology Laboratory
The Salk Institute for Biological Studies
10010 North Torrey Pines Road
La Jolla, CA 92037, U.S.A.
{magnus,terry}@salk.edu

Abstract

Probability theory represents and manipulates uncertainties, but cannot tell us how to behave. For that we need utility theory which assigns values to the usefulness of different states, and decision theory which concerns optimal rational decisions. There are many methods for probability modeling, but few for learning utility and decision models. We use reinforcement learning to find the optimal sequence of questions in a diagnosis situation while maintaining a high accuracy. Automated diagnosis on a heart-disease domain is used to demonstrate that temporal-difference learning can improve diagnosis. On the Cleveland heart-disease database our results are better than those reported from all previous methods.

1 INTRODUCTION

Probability theory represents and manipulates uncertainties in a principled way, but it can not tell us how to behave. To be able to do that we need *utility theory*, which values the usefulness of different states so that they can be compared. Together these two form *decision theory* which deals with how to make rational decisions under uncertainty. The most important idea in decision theory is the principle of *maximum expected utility* [von Neuman & Morgenstern, 1947]. It is a general principle for rational behavior, and we will in the long run benefit the most if we make our choices following this principle with some utility function. By studying people's preferences for different kinds of lotteries it has been found that

our behavior is indeed guided by this (see [Russell & Norvig, 1995] for a general discussion). Even "irrational" behavior can in fact be seen as rational—it is the utilities that are wrong.

There have been a lot of work on how to learn probability models, but not much on how to learn utility models. We will in this paper show that it can be done through reinforcement learning. A complete automated system can be built where the probability model is found from a database of cases, while the utility model is elicited by observing how the system behaves. This is demonstrated with a system for automated medical or machine diagnosis.

Diagnosis is a sequential decision process that starts from little initial information. More information is then requested in order to improve the result, and this is repeated in an iterative fashion. Many automated diagnosis systems have been designed, but most of them have to be specified manually, *e.g.*, Expert Systems or Bayesian Networks[1]. Due to the complexity of the domains we are interested in, it is hard or impossible to do this specification in an accurate and consistent way.

In [Stensmo & Sejnowski, 1995] a diagnosis system that learns from a database of cases was presented. This system builds a joint probability density model of the data which is then used for statistical inference to estimate the values of currently unknown variables. It is important to use joint probability density modeling in this application since there are always many unknown input variables. They can be found from the joint density in a principled way. Decision theory was used to find the most informative question to ask to get more information. Utilities determine how bad a misclassification is, and the goal is to maximize the expected utility. In previous work these utilities had to be manually specified and we used an approximation. The same approximation has been reported to work well in other work [Ledley & Lusted, 1959; Heckerman *et al.*, 1992], but we found that the results on our database were not optimal. This made us look for a way to learn the utilities.

Utilities cannot be found directly, but after each diagnosis we can get a "grade" on the sequence of questions which can be used to elicit them. Reinforcement learning is designed for this situation, and several learning methods have been developed. One of them is the method of Temporal-Differences [Sutton, 1988]. Systems that learn by observing their own behavior have been successful in earlier work, starting with a system that learned to play checkers [Samuel, 1959]. Tesauro's TD-Gammon system [Tesauro, 1992; Tesauro, 1993] learned master level backgammon by playing itself.

This paper demonstrates that utilities can be improved through reinforcement learning. This also means that utility values can be elicited from an expert that gives feedback on the quality of the diagnosis the system just performed. The expert therefore does not have to know what the utility values are. The only requirement is that he is consistent in the kind of feedback that he gives. We believe that our method could be used to find utility models for other decision theoretic systems.

Results on a database of heart-disease cases are presented.

[1] Certain kinds of networks can be learned from data [Russell & Norvig, 1995].

2 THE MODEL

2.1 PROBABILITY DENSITY MODEL

The joint probability density of the data was modeled by a finite linear mixture model [McLachlan & Basford, 1988],

$$\mathrm{p}(\boldsymbol{x}) = \sum_{j=1}^{M} \mathrm{p}(\boldsymbol{x}, \omega_j; \boldsymbol{\theta}_j) = \sum_{j=1}^{M} \mathrm{p}(\omega_j)\mathrm{p}(\boldsymbol{x}|\omega_j; \boldsymbol{\theta}_j).$$

There are M mixture components each denoted by ω_j. For each component there are two parameters, the *a priori* probability, or mixing proportion, $\mathrm{p}(\omega_j)$, and $\boldsymbol{\theta}_j$.

We have in this paper used binomial mixture components. Other components, *e.g.*, multinomial or Gaussian could have been used [Stensmo & Sejnowski, 1994]. Each data point $\boldsymbol{x}_i$ out of a set of size N is a D-dimensional binary vector. There is a parameter vector μ_j, also of length D, for each mixture component. Binomial mixture components have the form

$$\mathrm{p}(\boldsymbol{x}|\omega_j) = \prod_{d=1}^{D} \mu_{jd}^{x_d}(1-\mu_{jd})^{(1-x_d)}. \tag{1}$$

The Expectation-Maximization (EM) algorithm [Dempster *et al.*, 1977; Ghahramani & Jordan, 1994] was used to find the maximum likelihood estimates of the parameters. The algorithm consists of two steps that are repeated: An Expectation (E) step that estimates the probability that mixture component ω_j generated the data point,

$$p_j(\boldsymbol{x}_i) = \frac{\mathrm{p}(\omega_j)\mathrm{p}(\boldsymbol{x}_i|\omega_j)}{\sum_{k=1}^{M} \mathrm{p}(\omega_k)\mathrm{p}(\boldsymbol{x}_k|\omega_j)},$$

and a Maximization (M) step that updates the parameters

$$\hat{\boldsymbol{\mu}}_j \longleftarrow \frac{\sum_{i=1}^{N} p_j(\boldsymbol{x}_i)\boldsymbol{x}_i}{\sum_{i=1}^{N} p_j(\boldsymbol{x}_i)},$$

$$\hat{\mathrm{p}}(\omega_j) \longleftarrow \frac{\sum_{i=1}^{N} p_j(\boldsymbol{x}_i)}{N}.$$

Regression was used to find the expected values of the unknown variables. It is the conditional expectations of the missing variables given the observed ones. In the binomial case this means that $p_j(\boldsymbol{x})$ is only evaluated at the observed dimensions, and that corresponding μ_j values are used where there are missing dimensions for $\boldsymbol{x}$ in (1) [Ghahramani & Jordan, 1994].

See [Stensmo & Sejnowski, 1994; Stensmo & Sejnowski, 1995] for a longer discussion on mixture models, the EM-algorithm and diagnosis.

2.2 DECISION MODEL

In the diagnosis application we are concerned with misclassifications, and this is reflected by the utility model. A linear utility model was used as is customary in decision theory. There is a utility matrix U, where entry U_{ij} is the utility of having misclassified outcome i for j.

It does not have to be symmetric. In a linear model, the expected utilities of the different outcomes are Up, where the predicted probabilities for the different outcomes are in p.

The *maximum-expected-utility (MEU) principle* [von Neuman & Morgenstern, 1947] from decision theory was used. The selection of the most informative question to ask was done according to the *value of information criteria*. First, the maximum expected utility value is noted before any additional observations. Then each possible new answer to the unknown variables are imagined to be tested. For each of them the probabilities of the outcomes and their expected utilities are calculated. The gain in utility is the maximum expected utility minus the new utility of the best outcome without the new observation. This is calculated for each new variable and is weighed by the predicted probability of occurrence of each answer. This is the *value of information* if we observe the variable. The cost of the hypothetical observation is then subtracted resulting in the *net value of information*. There are established procedures to convert costs (in *e.g.* dollars) to units of probability (of *e.g.* losing your life as the result of a treatment) [Howard, 1980; Heckerman *et al.*, 1992].

If any of the variables now has a positive net value of information, the one with the highest value is requested. Should this not be the case, nothing more can be gained and we are finished with our diagnosis. If several possible questions have the same value, one is picked randomly. This is perfectly fine since they have the same "values" to the user.

It should be noted that this scheme does not take into account different sequences of questions. The number of possible sequences grows exponentially. A common approximation is therefore to look ahead only one-step (often called *myopic* for this reason). It has been reported to work well in many instances [Gorry & Barnett, 1967; Heckerman *et al.*, 1992].

The main drawback with this theory is that we have to get the utility values from somewhere. They have to be assessed by an expert on the domain, which can require a considerable effort. Approximations are therefore often used here too, for example, [Heckerman *et al.*, 1992] used utility values 1 for the misdiagnosis between either two malign or two benign diseases, and the value 0 otherwise. However, our experiments have shown that this is not optimal as will be seen in Section 3.

2.3 REINFORCEMENT LEARNING

The Temporal Difference (TD(λ)) algorithm [Sutton, 1988] was used for the reinforcement learning of the utility values. Input data are sequences of values for each time step $x_1, \ldots, x_m$. For each time step t a prediction P_t of the final reinforcement z is estimated. The actual reinforcement z is defined to be data point P_{m+1}. P_t is a function of inputs x and weights w, in the linear case just $P_t = wx$, which corresponds to the expected utilities Up.

TD(λ) updates the weights by

$$\Delta w_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k, \tag{2}$$

where α is the learning rate, and λ is how past experience is weighed in. [Sutton, 1988] derives and explains this and proves convergence for $\lambda = 0$. [Dayan, 1992] proves convergence for general λ.

In an on-line intra-sequence version of the algorithm [Sutton, 1988], the sum in (2) is performed separately in a variable e. The following update equations were used, with $U = w_t$,

Table 1: The Cleveland heart-disease database.

	Observation	Description	Values
1	age	Age in years	continuous
2	sex	Sex of subject	male/female
3	cp	Chest pain	four types
4	trestbps	Resting blood pressure	continuous
5	chol	Serum cholesterol	continuous
6	fbs	Fasting blood sugar	lt or gt 120 mg/dl
7	restecg	Resting electrocardiogr.	five values
8	thalach	Max heart-rate achieved	continuous
9	exang	Exercise induced angina	yes/no
10	oldpeak	ST depr. induced by exercise relative to rest	continuous
11	slope	Slope of peak exercise ST segment	up/flat/down
12	ca	# major vess. col. flourosc.	0-3
13	thal	Defect type	normal/fixed/reversible
	Disorder	**Description**	**Values**
14	num	Heart-disease	Not present/4 types

$p = x_t$ and $p' = x_{t-1}$. After each question (*i.e.* time step), the utility matrix U was updated by

$$\Delta U = \alpha U(p - p')e,$$

and e by

$$\Delta e = p + \lambda e. \tag{3}$$

At the end of the diagnosis sequence the reinforcement z is known. U is updated by

$$\Delta U = \alpha U(z - p)e.$$

and e as in (3).

The reinforcement given in our system were the correct diagnosis for each case, with a 1 for the correct position and a 0 otherwise. Other reinforcement values could be used (see Section 4). The patterns were presented randomly to remove possible ordering bias in the database. A complete randomized presentation of the entire data set is called an iteration. We reset the diagonal values to one after each of the iterations. Experiments without this operation still worked but resulted in a larger average number of questions, as well as a higher error rate.

3 RESULTS

The publically available Cleveland heart-disease database was used to test the method. It consists of 303 cases where the disorder is one of four types of heart-disease or its absence. There are thirteen variables as shown in Table 1. Continuous variables were converted into a *1-of-N* binary code. Nominal and categorical variables were coded with one unit per value. In total 55 binary variables coded the 14 original variables.

Iteration	Average	St. dev.	Errors
initial	5.31	1.77	78
1	6.32013	2.23455	47
2	6.25413	2.25188	45
3	6.37294	1.92135	51
4	6.48845	2.13122	47
5	6.47855	2.16658	47
6	6.39274	2.25102	46
7	6.16172	2.30969	49
8	5.9538	2.38882	46

Table 2: Results of TD-learning of the utilities with $\lambda = 0.1$, $\alpha = 0.01$.

1	0	0	0	0	1	0.3322	0.3419	0.3369	0.3090
0	1	1	1	1	0.1158	1	0.0988	0.1387	0.2872
0	1	1	1	1	0.0826	0.0819	1	0.0833	0.2396
0	1	1	1	1	0.0911	0.0649	0.0880	1	0.2479
0	1	1	1	1	0.0360	0.0281	0.0300	0.0826	1

Figure 1: Initial utility matrix (left). After 8 iterations with $\lambda = 0.1$, $\alpha = 0.01$ (right).

To find the parameter values, the EM-algorithm was run until convergence. A mixture model with 98 binomial mixture components gave the best classification results using just the mixture model to predict the probabilities of the outcomes from the observations of each case. We tried from 40 to 120 mixture components. The classification error was 46 or 15.2% with 98 components.[2]

Using the full utility/decision model and the 0/1-approximation for the utility matrix (left part of Figure 1), there were 78 errors. Over the whole data set an average of 5.31 questions were used with a standard deviation of 1.77. This is without TD-learning, *i.e.*, $\alpha = 0$.

With TD-learning, we varied λ from 0 to 1 in increments of 0.1, and found that $\lambda = 0.1$ gave us the best result. The learning rate α was also varied, $\alpha = 0.01$ was best for $\lambda = 0.1$. A higher learning rate gave rise to large fluctuations and instability, a lower one gave similar results but took longer time. Simulation results are shown in Table 2 for the best parameter values. The initial utility matrix is shown in the left side of Figure 1. The utility matrix after 8 iterations is shown on the right.

As can be seen in Table 2, the price paid for increased robustness is an increase in the average number of questions, but note that we can in fact get the same accuracy as we have when we use all of the input variables by using only less than half of them on average.

We have sometimes noted that the results get worse after a number of iterations. This is probably due to over-training and the stochasticity of the presentations. It has also been noted by others that have used TD-learning [Tesauro, 1992]. We can of course use early

[2]This result is considerably better than the 101 errors (33.3%) reported in [Stensmo & Sejnowski, 1994]. This is due to a better coding of the continuous variables and the use of binomial instead of Gaussian mixture components.

stopping to prevent this, and do not consider it to be a problem.

4 CONCLUSIONS AND FURTHER WORK

We have demonstrated that Temporal-Difference learning can be used to improve the accuracy of an automated diagnosis system based on probability and decision theory. The reinforcement that we used were chosen to make the system more robust. Other kinds of reinforcement can be used to bring down the number of questions needed, and to keep the total cost of the diagnosis to a minimum. The complicated and potentially error-prone process of eliciting utility values from an expert can thus be simplified. We believe that this technique could also be useful for other decision theoretic systems with linear or non-linear utility models.

This method can be extended in several directions. We intend to explore as many as possible of the following ideas within the next six months:

- The technique should be tested on more databases. Medical databases are not easy to find [Stensmo & Sejnowski, 1994; Stensmo & Sejnowski, 1995].
- Explore how different question costs will affect the behavior. This is already in the model and should be explored.
- We attempt to explore methods to automatically find good values for the α and λ parameters, perhaps along the lines of [Sutton & Singh, 1994].
- Utilities are considered to be non-linear [Grimmett & Stirzaker, 1992; Russell & Norvig, 1995] but are, probably by convenience, modeled linearly in decision theory, since manual specification and interpretation of the utility values then is easy. We believe that non-linear utilities could be useful. This eliciting can also be done through TD-learning with a multilayer utility network [Sutton, 1988; Tesauro, 1992].
- An alternative to learning the utility or value function is to directly learn the optimal actions to take in each state. A method for this is Q-learning [Watkins, 1989]. In our case this would be to learn which question to ask in each situation instead of the utility values. The myopic approximation would then not be necessary, but it would lead us away from the normal decision theoretic formulation.

Acknowledgements

The heart-disease database is from the University of California, Irvine Repository of Machine Learning Databases and originates from R. Detrano, Cleveland Clinic Foundation.

References

Dayan, P. (1992). The convergence of TD(λ) for general λ. *Machine Learning*, **8**, 341–362.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series, B.*, **39**, 1–38.

Feigenbaum, E. A. & Feldman, J., editors (1963). *Computers and thought: a collection of articles*. McGraw-Hill, New York, NY.

Ghahramani, Z. & Jordan, M. I. (1994). Supervised learning from incomplete data via an EM approach. In *Advances in Neural Information Processing Systems*, vol. 6, pp 120–127. Morgan Kaufmann, San Mateo, CA.

Gorry, G. A. & Barnett, G. O. (1967). Experience with a model of sequential diagnosis. *Computers and Biomedical Research*, **1**, 490–507.

Grimmett, G. R. & Stirzaker, D. R. (1992). *Probability and Random Processes*. Oxford Science Publications, Oxford, UK, second edition.

Heckerman, D. E., Horvitz, E. J. & Nathwani, B. N. (1992). Toward normative expert systems: Part I. The Pathfinder project. *Methods of Information in Medicine*, **31**, 90–105.

Howard, R. (1980). On making life and death decisions. In Schwing, R. C. & Albers, Jr., W. A., editors, *Societal risk assessment: How safe is safe enough?* Plenum Press, New York.

Ledley, R. S. & Lusted, L. B. (1959). Reasoning foundations of medical diagnosis. *Science*, **130(3366)**, 9–21.

McLachlan, G. J. & Basford, K. E. (1988). *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, Inc., New York, NY.

Russell, S. J. & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, New Jersey.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal on Research and Development*, **3**, 210–229. Reprinted in [Feigenbaum & Feldman, 1963].

Stensmo, M. & Sejnowski, T. J. (1994). A mixture model diagnosis system. Tech. Rep. INC-9401, Institute for Neural Computation, University of California, San Diego. URL: ftp://salk.edu/pub/magnus/INC-9401.ps.Z.

Stensmo, M. & Sejnowski, T. J. (1995). A mixture model system for medical and machine diagnosis. In *Advances in Neural Information Processing Systems*, vol. 7. MIT Press, Cambridge, MA. URL: ftp://salk.edu/pub/magnus/nips94.ps.Z.

Sutton, R. S. (1988). Learning to predict by the method of temporal differences. *Machine Learning*, **3**, 9–44.

Sutton, R. S. & Singh, S. P. (1994). On step-size and bias in temporal-difference learning. In *Eight Yale Workshop on Adaptive and Learning Systems*, pp 91–96. Center for Systems Science, Yale University.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, **8**, 257–277.

Tesauro, G. (1993). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, **6**, 215–219.

von Neuman, J. & Morgenstern, O. (1947). *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ.

Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, England.